

Rozhraní pro tvorbu 3D vizualizací bytů a menších rodinných domů určené k prezentaci zákazníkům prostřednictvím virtuální reality

Bc. Alexandr Hrach



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

Ústav elektroniky a měření

Akademický rok: 2024/2025

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	Bc. Alexandr Hrach
Osobní číslo:	A23372
Studijní program:	N1032A020003 Bezpečnostní technologie, systémy a management
Specializace:	Bezpečnostní technologie
Forma studia:	Prezenční
Téma práce:	Rozhraní pro tvorbu 3D vizualizací bytů a menších rodinných domů určené k prezentaci zákazníkům prostřednictvím virtuální reality
Téma práce anglicky:	Interface for Creating 3D Visualizations of Apartments and Smaller Houses for Presentation to Customers via Virtual Reality

Zásady pro vypracování

1. Popište problematiku 3D vizualizace a pojednejte o současných metodách.
2. Analyzujte nástroje, které jsou využívány k vizualizaci objektů pro bydlení.
3. Navrhněte a porovnejte alespoň dvě metody tvorby rozhraní 3D vizualizace objektů pro bydlení.
4. Vytvořte rozhraní pro tvorbu 3D vizualizace objektů pro bydlení.
5. Zkompletujte dokumentaci využití rozhraní pro tvorbu 3D vizualizace objektů pro bydlení.

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

Seznam doporučené literatury:

1. ROMERO, Marcos a SEWELL, Brenden. Blueprints Visual Scripting for Unreal Engine 5. 3. Packt Publishing, 2022. ISBN 9781801811583.
2. OGTEROP, Wilhelm a VENTER, Henk. Unreal Engine 5 Character Creation, Animation, and Cinematics. Packt Publishing, 2022. ISBN 9781801812443.
3. RACE, Steve. BIM Demystified. 2. RIBA Publishing, 2019. ISBN 9780429347627.
4. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management I. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBuM, 2011. ISBN 978-80-87500-05-7.
5. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management II. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBuM, 2012. ISBN 978-80-87500-19-4.
6. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management III. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBuM, 2013. ISBN 978-80-87500-35-4.
7. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management IV. Zlín: Radim Bačuvčík – VeRBuM, 2014. ISBN 978-80-87500-57-6.
8. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management V. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBuM, 2015. ISBN 978-80-87500-67-5.

Vedoucí diplomové práce: **Ing. Jiří Ševčík**
Ústav bezpečnostního inženýrství

Datum zadání diplomové práce: **29. listopadu 2024**

Termín odevzdání diplomové práce: **30. května 2025**

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan



Ing. Milan Navrátil, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 29. listopadu 2024

PROHLÁŠENÍ AUTORA DIPLOMOVÉ PRÁCE

Beru na vědomí, že

- odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení diplomové práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – diplomovou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na diplomové práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jakospoluautor;
- odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Předkládaná diplomová práce se zabývá návrhem a implementací rozhraní pro tvorbu 3D vizualizací bytů a menších rodinných domů, primárně určených k prezentaci zákazníkům prostřednictvím virtuální reality. Cílem práce je zpřístupnit 3D modelování a vizualizaci širšímu okruhu uživatelů, kteří dosud neměli s touto disciplínou zkušenosti. Tradiční modelovací programy jsou často složité a finančně náročné, což je činí nedostupnými pro běžné uživatele. Práce analyzuje stávající nástroje a metody v oblasti 3D vizualizace a virtuální reality, na základě čehož je navrženo a realizováno vlastní softwarové řešení v Unreal Engine 5. Zvláštní důraz je kladen na intuitivní ovládání a integraci prvků poplachových zabezpečovacích a tísňových systémů (PZTS), které aplikaci propůjčují unikátní specializaci v rámci vizualizace chytrých domácností. Součástí práce je detailní popis implementace klíčových funkcionalit, jako je systém stavění, ukládání a načítání dat, a práce s uživatelským rozhraním. Výsledkem je funkční a uživatelsky přívětivá aplikace, která umožňuje efektivní tvorbu a pohlcující prezentaci architektonických návrhů ve VR.

Klíčová slova: 3D vizualizace, virtuální realita, Unreal Engine, PZTS

ABSTRACT

This diploma thesis focuses on the design and implementation of an interface for creating 3D visualizations of apartments and smaller family houses, primarily intended for customer presentations through virtual reality. The aim of the thesis is to make 3D modeling and visualization accessible to a wider range of users who have no prior experience in this field. Traditional modeling software is often complex and financially demanding, rendering it inaccessible to average users. The thesis analyzes existing tools and methods in the field of 3D visualization and virtual reality, based on which a custom software solution is designed and implemented in Unreal Engine 5. Special emphasis is placed on intuitive control and the integration of alarm, security, and emergency systems (PZTS) elements, which gives the application a unique specialization within smart home visualization. The thesis includes a detailed description of the implementation of key functionalities, such as the building system, data saving and loading, and user interface management. The result is a functional and user-friendly application that enables efficient creation and immersive presentation of architectural designs in VR.

Keywords: 3D visualization, virtual reality, Unreal Engine, PZTS, user interface

„You miss 100% of the shots you don't take“

OBSAH

ÚVOD.....	13
I TEORETICKÁ ČÁST	14
1 PROBLEMATIKA 3D VIZUALIZACE A JEJÍ SOUČASNÉ METODY.....	15
1.1 ZÁKLADNÍ PRINCIPY TVORBY 3D SCÉNY	15
1.1.1 Tvorba modelu	15
1.1.2 Aplikace textur a materiálů	17
1.1.3 Osvětlení	18
1.1.4 Propojení	19
1.2 METODY RENDEROVÁNÍ	19
1.2.1 Dělení statické a dynamické.....	20
1.2.2 Metody renderování	20
1.2.3 Renderování ve VR	22
1.3 SPECIFICKÉ POŽADAVKY A METODY PRO VR VIZUALIZACI	22
1.3.1 Požadavky na VR.....	23
1.3.2 Techniky optimalizace	24
1.3.3 Specifické požadavky.....	24
2 NÁSTROJE VYUŽÍVÁNÉ K VIZUALIZACI OBJEKTŮ PRO BYDLENÍ.....	26
2.1 PŘEHLED KATEGORIÍ SOFTWAREVÝCH NÁSTROJŮ	26
2.2 ROLE CAD A BIM SYSTÉMŮ VE VIZUALIZAČNÍM WORKFLOW	27
2.3 VYUŽITÍ SPECIALIZOVANÝCH ARCHITEKTONICKÝCH VIZUALIZAČNÍCH NÁSTROJŮ	28
2.4 POTENCIÁL HERNÍCH ENGINŮ PRO ARCHITEKTONICKOU VIZUALIZACI A VR	29
3 EXISTUJÍCÍ SOFTWARE PRO VIZUALIZACI A TVORBU VR MODELŮ OBJEKTŮ.....	31
3.1 REVIT	31
3.2 SKETCHUP	31
3.3 SIMLAB VR VIEWER.....	31
II PRAKTICKÁ ČÁST	33
4 KOMPARACE PRO TVORBU ROZHRANÍ PROSTOROVÉ VIZUALIZACE.....	34
4.1 METODY TVORBY ROZHRANÍ 3D VIZUALIZACE	34
4.2 NÁVRH METODY A: MANUÁLNÍ / POLOMANUÁLNÍ MODELOVÁNÍ S PODPOROU PŮDORYSU	35
4.3 NÁVRH METODY B: POLOAUTOMATICKÉ GENEROVÁNÍ ZE STRUKTUROVANÝCH DAT	36

4.4	INTEGRACE PRVKŮ PZTS V RÁMCI NAVRŽENÝCH METOD	37
5	VÝBĚR VR PRO VÝVOJ	39
5.1	VARJO AERO	39
5.2	HTC VIVE PRO 2	39
5.3	META QUEST 3.....	39
5.4	ZDŮVODNĚNÍ.....	40
6	PŘENESENÍ PŮDORYSU DO 3D MODELU	41
6.1	BĚŽNĚ UŽÍVANÝ POSTUP.....	41
6.2	METODA POUŽÍVANÁ V PROGRAMU	41
7	VÝBĚR VHODNÉ PLATFORMY PRO VÝVOJ.....	43
7.1	UNITY.....	43
7.2	UN	43
7.3	ANDROID STUDIO	43
7.4	ZDŮVODNĚNÍ VÝBĚRU	44
8	TVORBA ASSETŮ.....	45
8.1	PROČ VYTVÁŘÍM ASSETY	45
8.2	BLENDER.....	45
8.3	AUTODESK MAYA	45
8.4	ZBRUSH.....	46
8.5	MOJE VOLBA + UKÁZKY	46
8.6	CO VŠE SE MODELOVALO.....	48
9	TVORBA APLIKACE – ZÁKLADNÍ NASTAVENÍ A VLOŽENÍ DAT	49
9.1	INSTALACE A NASTAVENÍ UNREALENGINE 5	49
9.1.1	Zvolená komprese	49
9.2	VÝBĚR ZPŮSOBU TVORBY	50
9.3	TVORBA LEVELŮ	51
9.3.1	Editor levely	52
9.3.2	VR levely.....	54
9.4	HERNÍ MÓDY	56
9.5	VYTVOŘENÍ PAWNCLASS	57
9.6	PŘIDÁNÍ VSTUPŮ.....	59
9.7	VLOŽENÍ ASSETŮ	60
9.8	TVORBA BAREV	61
9.9	TVORBA KOLIZÍ	63
9.10	TVORBA AKTORŮ.....	64

9.11	VYTVOŘENÍ DATA TABLE	65
10	TVORBA APLIKACE PROSTŘEDNICTVÍM BLUEPRINTU.....	67
10.1	ČASTO POUŽÍVANÉ BLOKY	67
10.2	ZÁKLADNÍ POHYB	71
10.3	SYSTÉM STAVĚNÍ.....	74
10.3.1	Detekce kolize	74
10.3.2	Výběr bloků.....	77
10.3.3	Rotace.....	78
10.3.4	Mazání bloků.....	79
10.3.5	Nastavitelný blok	80
10.4	IMPORT BUDOVY	82
10.4.1	Real Time Import/Export Plugin.....	83
10.4.2	Implementace	83
10.5	UKLÁDÁNÍ.....	86
10.5.1	Ukládání a načítání do slotů	89
10.5.2	Ukládání a načítání vrstev	89
10.6	WIDGETY	90
10.6.1	Tvorba menu	90
10.6.2	Tvorba UI	92
11	DOKUMENTACE APLIKACE	94
11.1	KLÍČOVÉ PRVKY	94
11.2	ZÁKLADNÍ OVLÁDÁNÍ.....	95
12	MOŽNÝ BUDOUCÍ VÝVOJ	96
	ZÁVĚR	97
	SEZNAM POUŽITÉ LITERATURY.....	99
	SEZNAM OBRÁZKŮ	102
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	104
	SEZNAM PŘÍLOH.....	105

ÚVOD

Diplomová práce se zabývá návrhem a implementací rozhraní pro tvorbu 3D vizualizací bytů a menších rodinných domů, primárně určených k prezentaci zákazníkům prostřednictvím virtuální reality. V dnešní době, kdy se 3D vizualizace stává nepostradatelnou součástí procesů navrhování a prodeje nemovitostí, je snahou vytvořit nástroj, který by byl dostupný a intuitivní i pro uživatele bez hlubokých technických znalostí v oblasti 3D modelování. Tradiční modelovací programy jsou často složité a finančně náročné, což je činí nedostupnými pro běžné uživatele nebo malé podniky. Práce si klade za cíl analyzovat stávající nástroje a metody v oblasti 3D vizualizace a virtuální reality, a na základě této analýzy navrhnout a realizovat vlastní softwarové řešení. Zvláštní důraz je kladen na integraci prvků poplachových zabezpečovacích a tísňových systémů (PZTS), což aplikaci propůjčuje unikátní specializaci v rámci vizualizace chytrých domácností. Projekt se soustředí na praktickou implementaci v Unreal Engine 5, který byl vybrán pro své pokročilé schopnosti v oblasti real-time renderování a robustní podporu virtuální reality. V práci budou detailně popsány klíčové fáze vývoje, včetně nastavení projektu, tvorby levelů, implementace uživatelských vstupů a správy assetů, až po komplexní systém stavění a ukládání dat.

I TEORETICKÁ ČÁST

1 PROBLEMATIKA 3D VIZUALIZACE A JEJÍ SOUČASNÉ METODY

V této kapitole bude popsána problematika 3D vizualizace, jejíž význam je v současné době široce využívána napříč mnoha obory. Bude pojednáno o základních principech, na nichž je 3D vizualizace založena, a budou vysvětleny procesy modelování, texturování, osvětlení a renderingu. Dále budou diskutovány současné metody renderingu, včetně porovnání real-time a offline přístupů. Rovněž budou identifikovány a popsány hlavní výzvy a problémy, s nimiž se v oblasti 3D vizualizace potýkáme, s cílem poskytnout ucelený úvod do tématu. [1]

1.1 Základní principy tvorby 3D scény

V této kapitole bude věnována pozornost fundamentálním principům, které tvoří základ pro vytvoření jakékoli trojrozměrné scény. Bude objasněno, jaké esenciální komponenty definují virtuální scénu, přičemž hlavní důraz bude kladen na tři klíčové aspekty, které jsou nezbytné pro její digitální ztvárnění. Tyto aspekty zahrnují proces modelování objektů a prostředí, aplikaci materiálů a textur pro definici vzhledu povrchů a nastavení osvětlení, které scéně dodává hloubku, atmosféru a věrohodnost.

1.1.1 Tvorba modelu

V rámci procesu tvorby trojrozměrné scény, který představuje fundament pro jakoukoli vizualizaci, ať už statickou, animovanou, interaktivní nebo určenou pro virtuální realitu, je prvním a naprosto nezbytným krokem samotná tvorba 3D modelu. 3D model je digitální reprezentací reálného nebo navrženého objektu či prostoru ve trojrozměrném digitálním prostředí. Jeho podstatou je geometrická definice tvaru a rozměrů, která tvoří základ, na který jsou následně aplikovány další vizuální atributy jako jsou materiály, textury a osvětlení, jež dodávají modelu jeho finální vzhled. Většina 3D modelů je v digitálním světě reprezentována pomocí tzv. polygonové sítě, která je složena z vrcholů bodů v prostoru definovaných souřadnicemi X, Y, Z hran jako spojnic mezi těmito vrcholy a stěn či polygony, které tvoří plochy ohraničené hranami, nejčastěji trojúhelníků nebo čtyřúhelníků. Vytvoření přesné, optimalizované a vhodné geometrie je kriticky důležité pro kvalitu celé navazující vizualizace, neboť ovlivňuje nejen vizuální dojem, ale také výpočetní náročnost scény, což je zvláště patrné v real-time aplikacích a VR. K tvorbě těchto modelů se využívají různé softwarové nástroje a metody, které se liší svým přístupem, složitostí, mírou automatizace a vhodností pro různé typy objektů a účely použití. [1]

Jednou z nejrozšířenějších, nejuniverzálnějších a v mnoha ohledech základních metod je polygonální modelování. Princip spočívá v přímé manipulaci s komponenty polygonové sítě – s jednotlivými vrcholy, hranami a stěnami. Modely jsou vytvářeny a upravovány pomocí široké škály nástrojů a technik, jako je extrudování (vytahování plochy nebo hrany do prostoru, čímž vzniká nová geometrie), řezání a dělení polygonů pro přidání detailů, spojování a sjednocování vrcholů a hran, nebo posouvání, otáčení a škálování vybraných částí sítě. Předností této metody je její velká flexibilita a přesná, granulární kontrola nad geometrií v každém bodě modelu. Je vhodná pro tvorbu široké škály objektů od jednoduchých geometrických tvarů až po složité detailní modely s komplexní topologií. V kontextu architektonické vizualizace je polygonální modelování velmi často využíváno pro tvorbu stavebních prvků, jako jsou stěny, okna, dveře, schodiště, ale i pro detailní modelování fasád, interiérových prvků a zejména pro tvorbu a úpravu modelů nábytku a dalšího vybavení interiéru a exteriéru, které je často potřeba přizpůsobit konkrétní scéně nebo designu. Nevýhodou této metody může být její relativní časová náročnost při tvorbě složitých organických tvarů, kde je třeba pečlivě hlídat plynulost povrchu a management polygonové sítě pro dosažení hladkého vzhledu bez viditelných hran, a nutnost manuálního úsilí i při tvorbě jednodušších opakujících se prvků.

Další významnou kategorií metod je modelování založené na křivkách a plochách, často reprezentovaných matematickým popisem, jako jsou NURBS (Non-Uniform Rational B-Splines) nebo Bézierovy křivky. Princip je založen na modelu, který není přímo vytvářen manipulací s body polygonové sítě, ale definováním řídicích bodů a křivek, které určují tvar povrchu. Z těchto křivek jsou následně generovány hladké plochy. Předností tohoto přístupu je schopnost vytvářet matematicky přesné a dokonale hladké povrchy, které jsou ideální pro organické tvary, aerodynamické plochy, design produktů s plynulými přechody, ale i pro přesné komponenty v inženýrství a architektuře, zejména tam, kde jsou vyžadovány hladké křivky a přesná geometrie. Modely vytvořené pomocí NURBS jsou původně nezávislé na rozlišení a mohou být převedeny do polygonové sítě s různou úrovní detailu podle potřeby finálního výstupu, což umožňuje dobrou optimalizaci. Tato metoda je často využívána v CAD systémech a programování pro architektonický návrh a bývá z těchto systémů importována do vizualizačních programů pro další zpracování. Nevýhodou může být pro některé uživatele méně intuitivní práce se složitější topologií scény jako celku a nutnost převodu na polygonovou síť pro většinu real-time renderovacích enginů a VR prostředí,

jelikož ty primárně pracují s polygony. Převod může někdy vést ke ztrátě původní přesnosti křivek.

Pro tvorbu trojrozměrných modelů lze využít metody vycházející z dat reálného světa. Jedná se o techniky založené na skenování, při němž je tvar měřen senzory, a fotogrammetrii, která rekonstruuje geometrii a texturu z analýzy mnoha překrývajících se fotografií. Obě metody jsou primárně určeny k digitalizaci a dokumentaci existujících reálných objektů či prostředí. Výsledkem bývá detailní trojrozměrný model, často s vysoce realistickou texturou odvozenou z pořízených dat. Jsou vynikající pro zachycení složitých tvarů, které by bylo obtížné modelovat ručně. Nejsou však využívány pro tvorbu modelů na základě projektové dokumentace, jako jsou půdorysy. Jejich použití vyžaduje specifické podmínky pro sběr dat, například stabilní osvětlení či nepohyblivost scény. Výstupem bývají často modely s velkým množstvím geometrie, což vyžaduje následnou optimalizaci pro real-time aplikace a VR. Navzdory těmto omezením představují cenný doplňkový nástroj umožňující obohatit virtuální scény o realistické digitální kopie prvků z reálného světa. [1]

Existuje několik hlavních metod pro tvorbu 3D modelů, přičemž často jsou v praxi kombinovány v závislosti na typu objektu, požadovaném detailu a účelu, pro který je model vytvářen. Například, základní konstrukce budovy může být vytvořena z CAD/BIM dat (související s NURBS nebo procedurálním přístupem), zatímco nábytek a dekorace mohou být modelovány polygonálně. Volba metody modelování představuje první a zásadní rozhodnutí v procesu přípravy 3D scény, jelikož kvalita a struktura geometrie přímo ovlivňuje možnosti aplikace materiálů, nastavení osvětlení a celkový výkon scény, což je zvláště kritické pro real-time vizualizaci a virtuální realitu, které vyžadují optimalizované modely s rozumným počtem polygonů. Tyto modely tvoří nezbytný základ pro všechny navazující kroky vedoucí k finální vizualizaci.

1.1.2 Aplikace textur a materiálů

Dalším klíčovým principem, který dodává 3D modelům vizuální bohatost a realismus, je aplikace materiálů a textur. Model sám o sobě, vytvořený pouze geometrií, je pouhou sadou abstraktních bodů a ploch bez určení jejich povrchových vlastností. Materiál je to, co definuje, jak bude povrch modelu interagovat s světlem, a určuje jeho vizuální vzhled. Bude vysvětleno, že materiály určují vlastnosti jako barvu, lesk, odrazivost, průhlednost, propustnost světla a další. Materiály jsou vytvářeny, často pomocí sady parametrů, které simulují fyzické vlastnosti reálných povrchů – tento přístup je znám jako PBR. Cílem PBR

materiálů je zajistit, aby model vypadal realisticky za různých světelných podmínek. Kromě základních materiálových vlastností se na povrchy aplikují textury. Bude popsáno, že textury jsou v podstatě obrázky, které se obalí na 3D model a dodávají povrchu detailní vzory, barvy a struktury, které by bylo složité nebo nemožné vytvořit čistou geometrií. Příkladem může být kresba dřeva na podlaze, struktura betonu na stěně, vzor tapety nebo detailní povrchová nerovnost zdiva simulovaná tzv. normalovou mapou. V kontextu vizualizace interiérů a exteriérů budov je pečlivý výběr, tvorba a aplikace materiálů a textur zásadní pro dosažení požadované atmosféry a realismu. Umožňuje věrně simulovat vzhled různých typů dřevěných podlah, lesklou plochu kuchyňské linky, drsnou texturu fasády, průhlednost skla v oknech nebo měkkost čalounění nábytku. [1]

1.1.3 Osvětlení

Třetím a neméně důležitým principem, který oživuje 3D scénu a dává jí hloubku a atmosféru, je nastavení osvětlení. Osvětlení má zásadní vliv nejen na pouhou viditelnost objektů ve scéně, ale především na celkovou náladu, vnímání tvarů, textur a realismu vizualizace. Bude popsáno, že bez správného osvětlení by i dokonale vymodelované objekty s detailními materiály působily ploše, nepřírozeně a nezajímavě.

Mezi základní osvětlení patří:

bodová světla, vyzařující světlo rovnoměrně do všech směrů.

- Směrová světla, která simulují světlo přicházející z nekonečné dálky, typicky slunce.
- Kuželová světla, vyzařující světlo ve specifikovaném kuželu, podobně jako reflektor
- Plošná světla, která vyzařují světlo z definované plochy, vhodná např. pro simulaci oken nebo studiového osvětlení.

Globální osvětlení, které představuje pokročilé techniky simulující, jak se světlo odráží od povrchů scény a nepřímo osvětluje ostatní objekty a prostory. Tento jev, kdy například světlo z okna dopadá na podlahu a odtud se odráží a osvětluje spodní část stěny, je klíčový pro realistické zobrazení interiérů, kde je přímého světla často málo a většina osvětlení pochází z odrazů. Je důležité pečlivé umístění a nastavení virtuálních zdrojů světla, jejich intenzity, barvy a parametrů globálního osvětlení je možné věrně simulovat denní dobu, atmosféru slunečného dne s ostrými stíny, zamračeného dne s měkkým rozptýleným světlem, nebo intimní atmosféru večerního interiéru osvětleného lampami. V kontextu

vizualizace bydlení se simuluje jak přirozené světlo procházející okny a dveřmi, tak i umělé osvětlení z různých typů svítidel umístěných v interiéru i exteriéru. Správné osvětlení zásadně ovlivňuje, jak jsou vnímány barvy a materiály, a dodává scéně potřebnou hloubku a realističnost.

1.1.4 Propojení

Teprve kombinací a vzájemným sladěním těchto tří základních principů – precizního 3D modelování pro vytvoření geometrické struktury, aplikace realistických materiálů a textur pro definování povrchových vlastností a detailů a pečlivého nastavení osvětlení pro vytvoření atmosféry a vizuální věrohodnosti, je vytvořena kompletní a vizuálně přesvědčivá 3D scéna. Bude zdůrazněno, že zvládnutí a cit pro tyto principy je nezbytným předpokladem pro tvorbu kvalitních vizualizací pro jakékoli účely a představuje základ, na kterém je postaven celý navazující proces renderování a optimalizace scény pro specifické výstupy, jako je právě interaktivní průzkum ve virtuální realitě, který je hlavním tématem této práce. Tyto základy budou dále rozpracovány a aplikovány v dalších částech práce při analýze nástrojů a návrhu metod pro tvorbu VR vizualizací objektů pro bydlení z půdorysů

1.2 Metody renderování

Po popsání základních principů tvorby trojrozměrné scény, tedy modelování, aplikace materiálů a textur a nastavení osvětlení, je nezbytné přejít k závěrečnému kroku, kterým je renderování. Bude popsán proces renderování jako technická fáze, během níž jsou veškerá data 3D scény – geometrie objektů, definice materiálů, nastavení osvětlení, pozice a parametry virtuální kamery – převedena do finálního dvourozměrného obrazu, který může být následně zobrazen na fyzickém výstupním zařízení, jako je monitor počítače nebo displej VR headsetu. [1]

Podstatou renderování je simulace šíření světla ve virtuální scéně a jeho interakce s povrchy objektů, a následné zachycení tohoto světla virtuální kamerou, která se chová podobně jako reálný fotoaparát či lidské oko. Hlavní výzva spočívá ve věrné simulaci komplexního chování světla v reálném světě, včetně jeho odrazů od povrchů, průchodů průhlednými materiály (refrakce), rozptylu a tvorby stínů. Pro dosažení realistického vzhledu je třeba v renderovacím procesu přesně řešit, jak je světlo vyzařováno virtuálními zdroji, jak se odráží od různých typů povrchů (jak přímé světlo dopadající přímo ze zdroje, tak nepřímé světlo vznikající odrazy od jiných objektů – globální osvětlení), jak prochází

průhlednými nebo průsvitnými objekty, jak je blokováno neprůhlednými objekty (tvorba stínů) a jak tyto světelné paprsky nakonec dopadají na senzor virtuální kamery, čímž je určena barva každého pixelu výsledného obrazu.

1.2.1 Dělení statické a dynamické

Real-timeRendering je renderování v reálném čase, probíhá dynamicky a za chodu, přičemž obrazy jsou generovány v okamžiku, kdy jsou potřeba pro zobrazení na výstupním zařízení. Hlavním cílem tohoto typu renderování je dosažení dostatečně vysoké snímkové frekvence, typicky 60 snímků za sekundu nebo více, a udržení minimální latence, tedy zpoždění mezi uživatelskou akcí a vizuální odezvou. Tento typ renderování je nezbytný pro interaktivní aplikace, kde je vyžadována okamžitá reakce na pohyb kamery či uživatelské vstupy. Uplatnění nachází především ve videohrách, simulátorech a klíčově také ve virtuální realitě, kde je nízká latence a vysoká snímková frekvence kritická pro komfortní zážitek. Využívány jsou především metody založené na rasterizaci, ačkoliv s rozvojem hardwaru jsou implementovány i některé prvky real-timeraytracingu. Kvalita bývala tradičně nižší než u offline renderování, ale s vývojem hardwaru a technik se vizuální rozdíly neustále zmenšují, ačkoliv je vždy kladen velký důraz na optimalizaci výkonu scény pro dosažení požadované plynulosti.

OfflineRendering je renderování, které probíhá předem, tedy před samotným zobrazením finálního výstupu. Každý jednotlivý snímek výsledné vizualizace je počítán nezávisle, přičemž na výpočet každého snímku může být vyhrazen značný čas, od sekund až po hodiny či déle, v závislosti na složitosti scény a požadované kvalitě. Hlavním cílem je dosažení maximální možné vizuální kvality, realističnosti a věrnosti. Důraz je kladen na přesnou simulaci fyzikálního chování světla ve scéně, včetně komplexních jevů jako je globální osvětlení, realistické odrazy, lomy světla či kaustiky. Využívány jsou pokročilé metody jako pathtracing či pokročilé varianty raytracingu. Tento typ renderování je využíván pro tvorbu finálních, nereálných médií, jako jsou statické fotorealistické obrázky pro marketing, předrenderované animace pro filmy, reklamy či architektonické prezentace. Neumožňuje interaktivní procházení nebo manipulaci s renderovanou scénou v reálném čase, jelikož každý pohled by musel být vypočítán zvlášť.

1.2.2 Metody renderování

Rasterizace je v současnosti základní a nejrozšířenější metodou renderování, která je primárně využívána pro grafiku v reálném čase. Její princip spočívá v promítání

trojrozměrné geometrie scény na dvourozměrnou rovinu obrazovky a následném vyplňování jednotlivých pixelů tvořících výsledný obraz. Barva každého pixelu je určována na základě vlastností materiálu objektu, textur aplikovaných na jeho povrch a zjednodušeného modelu osvětlení, který počítá interakci světla pouze v daném bodě. Hlavní předností rasterizace je její mimořádná rychlost a výpočetní efektivita, která je zajištěna rozsáhlou hardwarovou akcelerací pomocí moderních grafických procesorů. Je nezbytná pro dosažení vysoké snímkové frekvence v interaktivních aplikacích, jako jsou videohry a virtuální realita. Omezením rasterizace je její nižší schopnost věrně simulovat složité světelné jevy, jako jsou realistické odrazy, lomy světla a globální osvětlení bez použití dodatečných, často výpočetně náročných nebo přibližných technik. Přesto tvoří páteř moderní real-time grafiky díky svému bezkonkurenčnímu výkonu pro rychlé generování obrazu.

Raytracing je metoda renderování založená na simulaci dráhy světelných paprsků ve virtuální scéně. Typicky proces začíná vystřelením paprsků z virtuální kamery do scény pro každý pixel výsledného obrazu. Je sledována jejich první interakce s povrchy objektů. Na rozdíl od základní rasterizace však raytracing dokáže sledovat i následné odrazy (reflexní paprsky) od lesklých povrchů a lomy (refrakční paprsky) při průchodu průhlednými materiály. Tento přístup umožňuje věrně simulovat realistické odrazy a průhlednost s lomem světla. Představuje pokročilejší a výpočetně náročnější metodu než rasterizace. Tato metoda byla historicky primárně využívána pro offline renderování scén ve filmovém průmyslu či pro statické vizualizace, kde nebyl kladen důraz na kvalitu. S rozvojem moderního hardwaru, který poskytuje specializovaná jádra, je umožněno omezené využití prvků real-time Raytracingu i v interaktivních aplikacích pro dosažení vyšší vizuální kvality u specifických efektů, ačkoliv v plném rozsahu stále zůstává výpočetně náročnou metodou.

Pathtracing je pokročilá metoda renderování, která představuje rozšíření a zjemnění metody Raytracingu a je považována za jednu z nejpřesnějších metod pro simulaci fyzikálního chování světla. Namísto sledování jen několika málo odrazů simuluje cestu velkého množství náhodných světelných paprsků scénou, přičemž dokáže modelovat opakované interakce světla s různými typy povrchů, včetně rozptýlených odrazů od matných povrchů. Tento přístup umožňuje věrnou simulaci komplexních světelných jevů, které zásadně přispívají k fotorealismu, jako je globální osvětlení (nepřímé osvětlení odraženým světlem), realistické měkké stíny, kaustiky nebo rozptyl světla v materiálech (subsurfacescattering). Hlavní předností Pathtracingu je schopnost dosáhnout mimořádně vysoké míry fotorealismu a vizuální věrnosti scény. Je však výpočetně velmi náročná a k

dosažení čistého obrazu bez šumu, způsobeného simulací náhodných cest paprsků, vyžaduje výpočet velkého množství vzorků. Proto je primárně využívána pro vysoce kvalitní offline renderování statických obrázků a animací, kde je cílem dosáhnout maximálního realismu bez ohledu na čas výpočtu. Pro reálný čas je obvykle příliš pomalá, ačkoliv probíhá vývoj v oblasti optimalizací a využití strojového učení ke zrychlení tohoto procesu pro budoucí aplikace.

1.2.3 Renderování ve VR

V kontextu virtuální reality je nezbytnost real-time renderování zcela zásadní a tvoří technologický základ pro pohlcující zážitek. Bude zdůrazněno, že pro komfortní a věrohodný VR zážitek je vyžadována velmi vysoká snímková frekvence, typicky 60 snímků za sekundu nebo dokonce více, pro každé oko zvlášť, což efektivně zdvojnásobuje požadovaný výpočetní výkon oproti tradičnímu zobrazení na monitoru. Klíčová je rovněž extrémně nízká latence, tedy minimální zpoždění mezi pohybem hlavy uživatele a aktualizací obrazu ve VR headsetu, aby nedocházelo k nevolnosti a dezorientaci. Tyto přísné požadavky na výkon mohou být splněny pouze metodami real-time renderování. V současné VR je nejčastěji využívána právě pokročilá rasterizace, doplněná o různé optimalizační techniky a metody pro simulaci složitějších světelných jevů v reálném čase např. screen-space efekty nebopre-kalkulované osvětlení. S nástupem nového hardwaru se však otevírají dveře i pro omezené využití prvků real-time sledování paprsků pro dosažení vyššího realismu ve VR, ačkoliv to stále představuje značnou výpočetní výzvu. Neustále probíhá snaha o kompromis, který musí být mezi vizuální kvalitou renderované scény a nutným výkonem pro dosažení požadované snímkové frekvence a nízké latence ve VR, a z toho plynoucí nutnost použití efektivních optimalizačních technik při tvorbě VR obsahu.

1.3 Specifické požadavky a metody pro VR vizualizaci

Poté, co byly popsány základní principy tvorby 3D scény a metody renderování, je nezbytné přejít k analýze specifických požadavků, které klade technologie virtuální reality na 3D vizualizaci a z toho plynoucích metod a technik, které musí být pro úspěšnou realizaci VR zážitků aplikovány. Tvorba vizualizace určené pro zobrazení ve VR headsetu se totiž významně liší od tvorby statických obrázků nebo tradičních interaktivních aplikací zobrazovaných na monitoru. Jsou zde kladeny mnohem přísnější nároky, především na výkon a plynulost celého systému. [1]

1.3.1 Požadavky na VR

Jedním z klíčových specifík VR je potřeba zajištění stereoskopického vidění. Pro vytvoření věrohodného pocitu hloubky a reálného prostorového vnímání scény je nutné uživateli prezentovat mírně odlišný obraz pro každé oko. Každý z těchto obrazů je renderován z pozice virtuální kamery odpovídající poloze jednoho oka, přičemž tyto dvě pozice jsou od sebe vzdáleny přibližně jako lidské oči. Z toho přímo vyplývá, že scéna musí být renderována dvakrát pro každý snímek – jednou pro levé a jednou pro pravé oko. Tento požadavek bez použití speciálních optimalizací zdvojnásobuje celkový výpočetní nárok na grafický subsystém ve srovnání s renderováním pro jedno oko.

Dalšími kriticky důležitými aspekty pro komfortní, uvěřitelný a věrohodný VR zážitek jsou nízká latence a vysoká snímková frekvence. Pro zamezení tzv. kinetózy jinak známe jako nevolnosti z pohybu, která může být ve VR snadno vyvolána při nesouladu mezi pohybem uživatele a zobrazeným obrazem a pro udržení pocitu přítomnosti ve virtuálním světě je vyžadována extrémně nízká latence mezi fyzickým pohybem hlavy uživatele a okamžikem, kdy je v headsetu zobrazen odpovídající aktualizovaný obraz. Tato latence by měla být ideálně pod 20 milisekund. Zároveň je nutné dosáhnout konzistentně vysoké snímkové frekvence, která je pro většinu moderních VR systémů stanovena na 60 snímků za sekundu nebo dokonce více pro každé oko. Vysoká snímková frekvence zajišťuje plynulost pohybu a interakce ve virtuálním světě a redukuje cukání obrazu při rychlých pohybech hlavy, což rovněž přispívá k pohodlí a věrohodnosti zážitku.

Kombinace nutnosti renderovat scénu dvakrát s extrémně nízkou latencí a při velmi vysoké snímkové frekvenci klade na hardware a software enormní výpočetní nároky, které jsou řádově vyšší než u tradiční 3D grafiky zobrazované na monitoru. Aby bylo možné tyto přísné požadavky splnit a zajistit plynulý a komfortní VR zážitek i na spotřebitelském hardwaru, je prvořadá důkladná optimalizace výkonu 3D scény a celého renderovacího procesu. [1]

1.3.2 Techniky optimalizace

Splnění vysokých výkonnostních požadavků real-time renderování a virtuální reality klade na tvorbu trojrozměrných scén značné nároky. Aby bylo dosaženo nutné plynulosti zobrazení a nízké latence, je nezbytné využít specifické optimalizační techniky. Tyto metody slouží ke snížení výpočetní zátěže při zachování co nejvyšší vizuální kvality scény. V následujícím textu budou tyto klíčové metody optimalizace pro real-time a VR vizualizaci podrobněji popsány.

Optimalizace geometrie, spočívající v redukci počtu polygonů u složitých modelů tam, kde je to možné např. využitím LOD - Level of Detail, kdy se zobrazují zjednodušené verze modelů při pohledu z dálky, nebo odstraněním skrytých částí modelů. Odstraňování geometrie, která není viditelná z aktuálního pohledu culling, dále se optimalizují materiály a textury, například redukcí rozlišení velkých textur tam, kde vysoký detail není nutný, sdružováním menších textur do větších celků pro efektivnější práci grafické karty s pamětí nebo optimalizací složitosti tzv. shaderů, které určují vzhled materiálů a mohou být výpočetně náročné. V oblasti osvětlení se pro snížení real-time zátěže často využívá tzv. baked lighting, tedy předpečené (předpočítané) osvětlení a globální osvětlení, které sice neumožňuje dynamické změny světla v reálném čase, ale výrazně snižuje výpočetní zátěž během renderování, nebo se využívají zjednodušené modely real-time osvětlení. Kromě toho jsou využívány specifické renderovací techniky pro VR, jako je Single Pass Stereo Rendering nebo Multi-view rendering, které umožňují renderovat pohled pro levé a pravé oko efektivněji v rámci jednoho průchodu renderovací pipeline GPU.

1.3.3 Specifické požadavky

Kromě pouhé vizualizace je od moderních VR aplikací často očekávána i určitá míra interakce uživatele s virtuálním prostředím, což dále zvyšuje složitost tvorby a implementace. V rámci VR vizualizace bytů a domů se může jednat o různé typy interakce: základní navigace ve virtuálním prostoru např. plynulý pohyb pomocí ovladačů nebo teleportace na vybrané místo pro snížení rizika kinetózy, manipulace s virtuálními objekty jako je uchopení a posunutí kusu nábytku, otevírání dveří skříní, spouštění událostí, rozsvícení světla kliknutím na vypínač, změna materiálu na stěně výběrem z menu nebo interakce s virtuálním uživatelským rozhraním pro nastavení možností vizualizace. Tato interakce je typicky implementována pomocí skriptování a využití specifických softwarových sad (SDK - Software Development Kits) poskytovaných platformami pro VR

(např. Oculus SDK, SteamVR SDK), které umožňují získávat data o pohybu uživatele a interakci s ovladači a převádět je na akce ve virtuálním světě.

Tyto specifické požadavky a metody pro VR vizualizaci se přímo vztahují k tvorbě programu pro vizualizaci prostředí bytů a domů z půdorysů, který je předmětem této práce. Při tvorbě virtuálního modelu bytu či domu z půdorysu je nutné neustále myslet na výkonnostní optimalizace popsané výše, aby bylo dosaženo požadované vysoké snímkové frekvence a nízké latence pro plynulý a komfortní zážitek ve VR. Každá stěna, každý kus nábytku, každý materiál a zdroj světla musí být optimalizován s ohledem na zátěž pro grafickou kartu. Implementace interakcí, která umožní uživateli se nejen virtuálním prostorem procházet, ale také například otevírat dveře, okna, případně interagovat s prvky PZTS, které budou do scény přidány. Integrace interaktivních prvků PZTS do VR prostředí vyžaduje specifické programování a zajištění, aby jejich chování ve virtuálním světě odpovídalo zamýšlené funkcionalitě (např. aktivace alarmu při narušení zóny), a to vše při zachování požadované plynulosti a výkonu VR aplikace. Návrh a implementace těchto interaktivních prvků tvoří důležitou součást praktické realizace programu.

2 NÁSTROJE VYUŽÍVÁNÉ K VIZUALIZACI OBJEKTŮ PRO BYDLENÍ.

Vizualizace se stala nepostradatelnou součástí procesu navrhování, prezentace a prodeje objektů určených k bydlení. Umožňuje klientům, developerům i dalším zúčastněným stranám získat realistickou představu o budoucí podobě nemovitosti, ať už se jedná o rodinný dům, byt či developerský projekt. Dynamický rozvoj technologií přinesl na trh širokou škálu nástrojů, které se liší svou komplexností, účelem a výstupy – od jednoduchých 2D a 3D půdorysů, přes fotorealistické vizualizace interiérů a exteriérů, až po pohlcující virtuální prohlídky. Tato kapitola si klade za cíl analyzovat tyto rozmanité nástroje, prozkoumat jejich klíčové vlastnosti.

2.1 Přehled kategorií softwarových nástrojů

Po objasnění teoretických základů 3D vizualizace, principů tvorby scény a metod renderování, včetně specifik pro virtuální realitu, bude nyní pozornost přesunuta k praktičtějším aspektům, konkrétně k nástrojům, které jsou pro tyto účely reálně využívány. Obecný přehled různých typů softwaru, jež nacházejí uplatnění v celém procesu vizualizace objektů pro bydlení, počínaje prvotním návrhem a konče finální vizuální prezentací, včetně té ve virtuální realitě. Různé fáze vizualizačního workflow vyžadují různé nástroje s odlišnými funkcemi a specializacemi, které se často vzájemně doplňují prostřednictvím importu a exportu dat.

V závislosti na fázi projektu a požadovaném výstupu jsou využívány různé softwarové platformy. Základní kameny pro vizualizaci staveb bývají často položeny v systémech určených pro technické projektování. Dále se uplatňují programy zaměřené na tvorbu a úpravu trojrozměrných modelů s důrazem na uměleckou stránku či detail. Neméně významnou roli hrají specializované nástroje, které jsou přímo navrženy pro rychlou a kvalitní architektonickou vizualizaci. V posledních letech pak nezastupitelné místo získávají platformy původně vyvinuté pro tvorbu videoher, které se ukázaly jako mimořádně vhodné pro real-time interaktivní vizualizace, včetně těch pro virtuální realitu. [2]

První významnou skupinou jsou systémy CAD (Computer-Aided Design) a BIM (BuildingInformation Modeling). Tyto programy slouží primárně k přesnému technickému návrhu a dokumentaci staveb a představují často výchozí zdroj dat ve formě půdorysů, řezů a základních 3D modelů. Druhou skupinu tvoří univerzální 3D modelovací programy, které

nabízejí pokročilé nástroje pro detailní tvorbu a úpravu složitých trojrozměrných objektů, jež nemusí být nutně přímo součástí stavební konstrukce, ale jsou nezbytné pro vybavení interiéru či exteriéru. Třetí kategorií jsou specializované architektonické vizualizační nástroje, které jsou navrženy tak, aby zjednodušily proces vytváření realistických renderů budov a interiérů a často obsahují rozsáhlé knihovny předpřipravených modelů a materiálů. Poslední, avšak pro téma této práce mimořádně důležitou kategorií, jsou herní enginey, které původně sloužily k vývoji videoher, ale díky svým schopnostem real-time renderování a podpoře interaktivity a VR se staly klíčovými nástroji pro tvorbu pohlcujících vizualizací.

Tyto kategorie softwaru se v reálném vizualizačním workflow často prolínají a data jsou mezi nimi přenášena pomocí různých souborových formátů. Efektivita celého procesu závisí na kompatibilitě nástrojů a hladkém přenosu informací mezi nimi.

2.2 Role CAD a BIM systémů ve vizualizačním workflow

Po představení obecných kategorií softwarových nástrojů využívaných v oblasti vizualizace bude nyní detailněji popsána role, kterou v tomto procesu sehrávají systémy CAD a BIM. Tyto platformy představují často nezbytný výchozí bod pro jakoukoli vizualizaci stavebních objektů, neboť právě v nich vzniká primární technická dokumentace a geometrická data budoucí stavby. [2]

CAD systémy jsou tradičně používány pro tvorbu přesných dvourozměrných technických výkresů, jako jsou půdorysy, řezy a pohledy. V kontextu vizualizace objektů pro bydlení slouží půdorys vytvořený v CAD programu jako základní grafický podklad, ze kterého lze následně odvozovat trojrozměrnou geometrii stěn, místností a dalších architektonických prvků. Některé pokročilejší CAD programy umožňují i tvorbu jednoduchých trojrozměrných modelů. Hlavním přínosem CAD dat je jejich přesnost a standardizace, což umožňuje jejich další zpracování v navazujících vizualizačních softwarech.

Význam BIM systémů ve vizualizaci je v posledních letech stále rostoucí. BIM není pouze o 3D geometrii, ale jedná se o komplexní proces vytváření a správy informací o stavbě během celého jejího životního cyklu. BIM model obsahuje inteligentní objekty např. stěny, okna, dveře, které mají kromě geometrie přiřazeny i různé parametry a informace jako typ materiálu, tepelné vlastnosti, výrobce okna. V kontextu vizualizace to znamená, že BIM model již v sobě nese podstatně bohatší datovou základnu než pouhý 3D model z CADu. Z BIM modelu lze snadno exportovat nejen přesnou 3D geometrii celé budovy s definovanými

architektonickými prvky, ale často i informace o použitých materiálech nebo dokonce o typech objektů, což může významně zjednodušit a zrychlit proces přípravy vizualizace.

Ve standardních formátech dat, ve kterých jsou data z CAD a BIM systémů typicky exportována pro účely vizualizace nejčastější patří formáty jako DWG, primárně pro 2D půdorysy a jednoduché 3D modely, DXF, ale především formáty pro výměnu 3D modelů jako OBJ, FBX nebo specializovaný formát IFC (IndustryFoundationClasses), který je standardem pro výměnu dat v BIM procesech a nese s sebou nejvíce informací o stavebních prvcích.

Důležitost kvalitně připravených vstupních dat z CAD a BIM systémů pro plynulost navazujícího vizualizačního procesu. Nepřesnosti v půdorysu, chybějící nebo nesprávné informace v BIM modelu mohou vést k nutnosti složitých úprav ve vizualizačním softwaru, což celý proces prodlužuje a prodražuje. Role CAD a BIM systémů je tedy v podstatě role dodavatele základních, přesných dat o geometrii a struktuře objektu pro bydlení, na jejichž základě je následně ve specializovaných vizualizačních nástrojích vytvářena vizuálně bohatá a realistická 3D scéna. Pro tvorbu VR vizualizace z půdorysů je schopnost vizualizačního softwaru efektivně importovat a interpretovat data z těchto systémů zcela klíčová.[2]

2.3 Využití specializovaných architektonických vizualizačních nástrojů

Po prozkoumání role systémů CAD a BIM jakožto primárních zdrojů architektonických dat se nyní pozornost přesune ke skupině softwarových nástrojů, které jsou navrženy specificky pro účely architektonické vizualizace. Tyto nástroje se od obecných 3D modelovacích programů liší především svým zaměřením na potřeby architektů, interiérových designérů a developerů, s cílem zjednodušit a zrychlit proces tvorby realistických vizualizací budov a interiérů.

Klíčovou vlastností těchto specializovaných nástrojů je jejich optimalizace pro práci s architektonickými daty a často nabízejí přímou podporu pro import souborů z CAD a BIM systémů, přičemž dokáží lépe interpretovat stavební prvky, jako jsou stěny, okna, dveře, a převést je do podoby optimalizované pro 3D vizualizaci. Tyto programy bývají vybaveny rozsáhlými knihovnami předpřipravených 3D modelů nábytku, doplňků, rostlin a lidí, stejně jako knihovnami materiálů jako dřevo, beton, sklo, textilie, omítky, což výrazně zrychluje proces vybavování interiéru a exteriéru scény.

Renderovací schopnosti těchto nástrojů. Mnoho z nich využívá moderní a rychlé renderovacíenginy, které dokáží v relativně krátkém čase vygenerovat fotorealistické snímky nebo dokonce plynulé animace. Některé z těchto nástrojů pracují na principu real-time renderování, což znamená, že uživatel vidí výsledek svých úprav např. změna materiálu, nastavení světla okamžitě, což významně zefektivňuje pracovní postup. Příkladem takových nástrojů mohou být Lumion, Enscape, Twinmotion nebo V-Ray, které jsou často používané jako renderovací plugin pro jiné 3D programy. Každý z nich má své specifické silné stránky, například v rychlosti renderování, kvalitě výstupu, rozsahu knihoven nebo integraci s jinými architektonickými softwaremi.

Specializované architektonické vizualizační nástroje se zaměřují především na produkci vizuálně atraktivních výstupů pro marketing a prezentaci. Jejich primárním cílem je vytvořit realistické obrázky a animace, které prodají návrh. Ačkoliv některé z těchto nástrojů v současné době nabízejí i určitou míru podpory pro tvorbu VR výstupů, tato podpora nemusí být vždy tak robustní nebo flexibilní jako u herních enginů. Možnosti pokročilé interaktivity, jako je složitější manipulace s objekty nebo implementace vlastních funkcionalit např. specifické chování prvků PZTS, mohou být v těchto specializovaných nástrojích omezené nebo vyžadovat složitější postupy.

2.4 Potenciál herních enginů pro architektonickou vizualizaci a VR

Závěrečnou a pro účely této práce mimořádně relevantní kategorií softwarových nástrojů, které budou analyzovány, jsou herní enginey. Platformy jako Unity a UnrealEngine, původně vyvinuté primárně pro tvorbu videoher, v posledních letech získaly významné postavení i v oblasti architektonické vizualizace a interaktivních aplikací, zejména těch určených pro virtuální realitu. Tento přesun je umožněn díky jejich pokročilým schopnostem v oblasti real-time renderování a robustní podpoře interaktivity a VR hardwaru.

Herní enginey excelují v real-time renderování, které je, jak bylo popsáno v předchozí kapitole, nezbytné pro zajištění plynulého a komfortního zážitku ve virtuální realitě.[3] Dokáží zpracovat složité 3D scény a generovat vysoký počet snímků za sekundu, což je klíčové pro dosažení nízké latence a zamezení kinetózy. Jsou vybaveny výkonnými renderovacípipelinami (kroky při, kterých je 3D obraz převeden na 2D), které podporují moderní grafické techniky a často již integrují i možnosti real-time simulace pokročilých světelných jevů, jako je globální osvětlení nebo odrazy.

Kromě renderovacích schopností je hlavním přínosem herních engineů pro architektonickou vizualizaci jejich robustní podpora interaktivity. Bude popsáno, že poskytují komplexní nástroje pro skriptování a programování chování objektů a reakcí na uživatelské vstupy např. pohyb ovladačů, stisk tlačítek. To umožňuje vytvářet vysoce interaktivní vizualizace, ve kterých se uživatel může nejen volně pohybovat, ale také manipulovat s objekty, otevírat dveře, měnit materiály, zapínat světla nebo spouštět jiné události. Právě tato flexibilita v tvorbě interakce je zásadní pro integraci specifických funkcionalit, jako je vizualizace a interakce s prvky PZTS.

Herní enginey mají nativní podporu pro většinu hlavních VR headsetů a platform a poskytují vývojářům potřebné nástroje (SDK) pro práci se stereoskopickým zobrazením, sledováním pohybu hlavy a interakci s VR ovladači. To umožňuje vytvářet plně pohlcující VR průchody architektonickými návrhy s vysokou mírou realismu a interaktivity.

Práce v herních enginech může být ale pro architektky bez zkušeností s herním vývojem zpočátku složitější. Import architektonických dat z CAD/BIM systémů může vyžadovat určité úpravy a optimalizace 3D modelů pro real-time renderování. Nicméně, díky jejich výkonu, flexibilitě a podpoře pro VR a interaktivitu představují herní enginey mimořádně silný nástroj pro realizaci cílového programu popsaného v zadání této práce, tedy pro tvorbu VR vizualizace objektů pro bydlení z půdorysů s možností interakce s prvky PZTS. Bude provedeno zhodnocení, proč jsou herní enginey pro tento specifický účel vhodné a jaké výhody nabízejí ve srovnání s tradičními architektonickými vizualizačními nástroji, zejména pokud jde o implementaci pokročilé interaktivity a optimalizaci pro VR. Tím bude uzavřena kapitola o analýze nástrojů a připraven půda pro kapitolu následující, která se bude věnovat konkrétním existujícím softwarovým řešením.[4]

3 EXISTUJÍCÍ SOFTWARE PRO VIZUALIZACI A TVORBU VR MODELŮ OBJEKTŮ

Ve světě digitálního designu a 3D modelování existuje celá řada nástrojů, které uživatelům umožňují přenést jejich kreativní vize do virtuálního prostoru. Ačkoli tyto programy často disponují ohromujícím množstvím funkcí a relativně uživatelsky přívětivým rozhraním, často se skrývá háček. Jejich cena bývá pro běžné uživatele nebo malé podniky neúměrně vysoká. Navíc, i přes snahu o zjednodušení, mohou být tyto sofistikované nástroje s velkým množstvím funkcí příliš komplexní pro ty, kteří nemají hlubší technické znalosti. Pro mnoho lidí se tak 3D modelování stává nedostupnou disciplínou, vyhrazenou pouze pro profesionály nebo velké společnosti.

3.1 Revit

Revit je software pro informační modelování budov (BIM) od společnosti Autodesk, určený pro architekty, stavební inženýry a dodavatele. Umožňuje vytvářet komplexní 3D modely budov s inteligentními informacemi o prvcích a nabízí pokročilé BIM funkce pro koordinaci a spolupráci, parametrické modelování a rozsáhlou knihovnu prvků a materiálů. Jeho nevýhodou je strmější křivka učení, vyšší cena a vysoké požadavky na hardware. Revit se prodává formou předplatného a ceny se liší podle varianty a délky předplatného.[5]

3.2 SketchUp

SketchUp je 3D modelovací software od společnosti Trimble, známý svou snadnou použitelností a intuitivním rozhraním. Používá se pro architektonické návrhy, interiérový design a 3D tisk a je dostupný v několika verzích, od bezplatné SketchUp Free až po profesionální SketchUp Studio. Mezi jeho výhody patří snadné a rychlé modelování, velká komunita a rozsáhlá knihovna 3D modelů (3D Warehouse). Nevýhodou jsou omezené funkce pro pokročilé BIM a menší vhodnost pro komplexní stavební projekty. SketchUp nabízí bezplatnou verzi a placené verze se prodávají formou předplatného.[6]

3.3 SimLab VR Viewer

SimLab VR Viewer je software pro prohlížení 3D modelů a VR scén, který podporuje širokou škálu formátů souborů a VR zařízení. Umožňuje interaktivní prohlížení modelů ve VR a vytváření a sdílení interaktivních VR prezentací. Je dostupný i pro mobilní zařízení.

Jeho hlavní nevýhodou je, že je primárně určen pro prohlížení, nikoli pro tvorbu modelů.
Cena SimLab VR Viewer se liší podle licence a existuje bezplatná zkušební verze.[7]

II PRAKTICKÁ ČÁST

4 KOMPARACE PRO TVORBU ROZHRAŇÍ PROSTOROVÉ VIZUALIZACE.

Pro efektivní interakci s 3D vizualizacemi objektů je klíčové kvalitní a intuitivní uživatelské rozhraní. Právě rozhraní zprostředkovává uživateli možnost prohlížet si prostor, provádět úpravy a získat tak ucelenou představu. Návrh rozhraní pro práci s komplexními 3D daty představuje specifickou výzvu. Tato kapitola se proto zaměří na návrh a detailní srovnání metod tvorby rozhraní pro 3D.

4.1 Metody tvorby rozhraní 3D Vizualizace

Po analýze současných nástrojů využívaných v oblasti architektonické vizualizace a VR bude nyní pozornost přesunuta posouzení konkrétních metod, kterými by bylo možné realizovat program pro VR vizualizaci prostředí bytů a domů z půdorysu. V rámci této kapitoly bude nejprve nezbytné upřesnit a definovat pojem metody tvorby rozhraní 3D vizualizace tak, jak je chápán v kontextu tohoto projektu. Tento pojem se primárně nevztahuje k návrhu grafického uživatelského rozhraní samotné aplikace ve smyslu tlačítek, menu a ovládacích prvků, ačkoliv i ty jsou důležité pro interakci uživatele s programem. Místo toho se jedná o technický a pracovní postup, kterým je z dodaného vstupního půdorysu získána, generována či vytvořena plnohodnotná a průchozí trojrozměrná scéna. Tato scéna musí být následně optimalizována a připravena pro zobrazení ve virtuální realitě a musí umožňovat vkládání a interakci s doplňujícími prvky, zejména s prvky PZTS.

Nyní, po úvodním přiblížení významu uživatelského rozhraní pro 3D vizualizace, se v této podkapitole detailně zaměříme na definici metod jejich tvorby. Metodou zde rozumíme komplexní přístup k designu, který zahrnuje jak principy vizuálního rozvržení, tak především modely interakce uživatele s 3D prostorem a obsahem. Nejde jen o estetiku, ale o základní filozofii, jak uživatel s vizualizací pracuje.

Navržené metody se budou zabývat celým procesem transformace z 2D dat do 3D VR prostředí. To zahrnuje nejen samotné vytvoření geometrie stěn, podlah a stropů na základě půdorysných informací, ale také zohlednění výšek, umístění otvorů pro okna a dveře, přidání základních architektonických detailů a přípravu scény pro další vizualizační kroky, jako je aplikace materiálů, osvětlení a přidání vybavení. Důležitou součástí definice metod je rovněž zohlednění, jakým způsobem mohou být do takto vytvořené scény integrovány specifické

prvky, jako jsou komponenty PZTS, a jak je zajištěna možnost interakce s těmito prvky v rámci VR prostředí.

Cílem této části práce je navrhnout alespoň dva odlišné přístupy k tomuto procesu, které se budou lišit v závislosti na míře automatizace, požadavcích na vstupní data a nárocích na uživatele či software. Tyto metody budou následně podrobeny analýze z hlediska různých kritérií, jako je efektivita, přesnost, flexibilita, technická složitost a vhodnost pro daný účel, tedy tvorbu VR vizualizace z půdorysu s integrací PZTS. Navržené metody budou představovat teoretické základy pro realizaci programu, který je předmětem této práce.

4.2 Návrh Metody A: Manuální / Polomanuální modelování s podporou půdorysu

První navrhovaná metoda tvorby 3D vizualizace objektů vychází z půdorysu. Je založena na přístupu, který využívá manuální nebo polomanuální modelování trojrozměrné geometrie s přímou podporou dodaného půdorysu jako vizuálního nebo referenčního podkladu. Tento přístup představuje do jisté míry tradiční cestu v 3D modelování architektonických scén.

Prvním krokem je import půdorysu do 3D modelovacího softwaru nebo herního enginu. Půdorys může být dodán ve formě obrázku např. JPG, PNG nebo jako vektorová data např. DWG, DXF. Pokud je použit obrázek, je nutné jej nejprve správně zkalibrovat a umístit v 3D prostoru tak, aby jeho měřítko odpovídalo reálným rozměrům. V případě vektorových dat bývá proces importu přesnější, jelikož software dokáže často přímo pracovat s čarami a křivkami definujícími stěny a další prvky. Následně je na základě importovaného půdorysu prováděno manuální modelování. Zde jsou trasovány obvodové a vnitřní stěny, definováním jejich tloušťky a výšky, čímž vzniká základní hrubá stavba. Dále je prováděno modelování podlah a stropů. Následuje fáze vytváření otvorů pro okna a dveře na místech určených půdorysem a jejich vyplnění modely oken a dveří. V této fázi jsou také přidávány další architektonické prvky, jako jsou schodiště, zábradlí, výklenky nebo římsy, opět s využitím půdorysu a případně dalších dostupných řezů či pohledů jako reference.

Součástí této metody je i následné detailní modelování interiéru a exteriéru, pokud je to pro vizualizaci nutné. Do scény je také možno vložit modely nábytku, doplňků, kamer, detektorů nebo dalších prvků PZTS, EPS a dalšího vybavení, přičemž jejich umístění je často určeno opět půdorysem např. pozice kuchyňské linky, vany, postele nebo jiné značení, pokud je podporováno programem. Aplikace materiálů a textur a nastavení osvětlení scény

probíhá standardními postupy pro 3D vizualizaci, jak byly popsány v předchozích kapitolách.[8]

Hlavní předností manuálního či polomanuálního modelování je vysoká flexibilita. Umožňuje pracovat s jakýmkoli typem půdorysu, i s těmi, které nejsou dokonale zpracované nebo obsahují netradiční architektonické prvky. Je možné vytvářet velmi detailní a umělecky pojaté modely. Nicméně hlavní nevýhodou je její časová náročnost a vysoká závislost na zručnosti a zkušenostech uživatele, který modelování provádí. Pro rozsáhlé objekty nebo velký počet vizualizací může být tento proces velmi zdoluhavý. Přesnost výsledného modelu je rovněž silně závislá na pečlivosti modeláře. Tato metoda nicméně poskytuje plnou kontrolu nad každým detailem scény, což může být výhodné pro specifické požadavky nebo vysoce individualizované návrhy. [8]

4.3 Návrh Metody B: Poloautomatické generování ze strukturovaných dat

Jako alternativa k manuálnímu modelování je metoda, která je založena na principu poloautomatického generování 3D geometrie a základního uspořádání scény na základě zpracování strukturovaných vstupních dat. Tento přístup se snaží využít a automatizovat části procesu tam, kde jsou k dispozici data ve formátu, který obsahuje více informací než jen pouhé čáry půdorysu.

Klíčovým předpokladem je dostupnost strukturovanějších vstupních dat, ideálně pocházejících ze systémů BIM nebo pokročilejších CAD programů, které ukládají stavební prvky jako objekty s definovanými vlastnostmi, nikoli jen jako grafické entity. Data by měla obsahovat informace o typech objektů např. stěna, okno, dveře, jejich geometrii v 2D nebo 3D např. tloušťka, výška, umístění a případně informace o materiálech. Typickými vstupními formáty pro tuto metodu by mohly být DWG se správně strukturovanými vrstvami a bloky, nebo ideálně formát IFC. V rámci této metody by byl vyvinut nebo využit software či skript, který by dokázal tato strukturovaná data načíst a interpretovat. Využívá se zdeparsování dat, kdy jsou identifikovány jednotlivé stavební prvky a jejich parametry. Na základě těchto informací by následně byla programově generována odpovídající trojrozměrná geometrie – například by z definice stěny na půdorysu s informací o výšce a tloušťce byl automaticky vytvořen 3D model stěny, do kterého by byly na základě informací o oknech a dveřích vytvořeny odpovídající otvory a případně do nich vloženy standardní 3D modely oken a dveří z knihovny.

Po automatickém vygenerování základní geometrie stavby by následovala fáze poloautomatického nebo manuálního dokončení scény. Zde je prostor pro nutné přidání detailů, které nebyly obsaženy ve strukturovaných datech nebo je nelze snadno automaticky generovat např. lišty nebo jiné specifické prvky. Rovněž by bylo nutné manuálně vložit a uspořádat nábytek, dekorace a další vybavení, pokud by tyto prvky nebyly součástí vstupních dat v použitelné formě. Aplikace materiálů a nastavení osvětlení by probíhala buď automaticky na základě informací z BIM modelu pokud jsou k dispozici nebo manuálně, podobně jako u První metody.

Hlavní předností poloautomatického generování je potenciální úspora času a zvýšení rychlosti procesu tvorby základní 3D scény, zejména u standardních nebo opakujících se typů staveb. Pokud jsou vstupní data správná je závislost na zručnosti modeláře pro vytvoření základní geometrie snižována a je zajišťována vyšší míra přesnosti. Hlavní nevýhodou je však závislost na kvalitě a struktuře vstupních dat. Pokud data nejsou dobře připravena, je automatizace obtížná nebo nemožná a vyžaduje značné předběžné úpravy dat. Tato metoda je méně flexibilní pro netradiční nebo velmi složité architektonické návrhy, které se vymykají standardním stavebním prvkům, jež lze snadno programově generovat. Vyžaduje také větší technickou expertízu pro vývoj a implementaci samotného generovacího mechanismu.

4.4 Integrace prvků PZTS v rámci navržených metod

V této podsekci bude popsáno, jakým způsobem je v rámci obou navržených metod tvorby 3D vizualizace objektů pro bydlení řešena integrace prvků poplachových zabezpečovacích a tísňových systémů. Cílem není pouze vizuálně zobrazit tyto prvky ve virtuální scéně, ale ideálně umožnit s nimi i základní interakci a simulovat jejich funkci v rámci VR prostředí. V nejjednodušším případě se jedná o vkládání standardních 3D modelů těchto zařízení do virtuálního prostoru bytu či domu.

V případě první metody, manuální / polomanuální modelování, by vkládání prvků PZTS probíhalo převážně manuálně. Po vytvoření 3D modelu stavby a interiéru by uživatel z knihovny vybral příslušné 3D modely komponent PZTS a umístil je na požadovaná místa ve scéně. Tento přístup poskytuje plnou flexibilitu v umístění, ale je časově náročný a vyžaduje od uživatele znalost, kde mají být jednotlivé prvky umístěny podle návrhu zabezpečení. Využívají se zde značení na 2D výkresu jako vodítka pro manuální umístění 3D modelů ve virtuální scéně.

Pro druhou Metodu poloautomatické generování ze strukturovaných dat by mohla být zvažena určitá míra automatizace i při vkládání prvků PZTS. Pokud by vstupní strukturovaná data obsahovala informace o typech a přesných pozicích prvků PZTS, bylo by možné tyto prvky do 3D scény vkládat poloautomaticky během procesu generování stavby. Software by na základě informací o umístění detektoru v datovém souboru vložil odpovídající 3D model detektoru na příslušnou pozici ve virtuálním prostoru. Tento přístup by výrazně urychlil osazení scény prvky PZTS, ale vyžadoval by specifický formát a strukturu vstupních dat obsahujících informace o systému PZTS.

Pouhé zobrazení ale nestačí, a pro lepší pochopení funkce systému by měla být umožněna základní interakce. To může zahrnovat například zobrazení informací o daném prvku, typ detektoru, zóna nebo stav po namíření virtuálním ukazovátkem nebo kliknutí na něj nebo simulaci základní funkce jako rozsvícení virtuální kontrolky při detekci pohybu virtuálním pohybem uživatele. Implementace této interakce vyžaduje skriptování v prostředí, ve kterém je VR vizualizace provozována typicky herní engine, a propojení 3D modelů prvků PZTS s logikou jejich chování. Implementace interakce by probíhala nezávisle na zvolené metodě tvorby základní 3D scény, a představovala by dodatečnou programovací fázi, která by byla součástí obou postů.

5 VÝBĚR VR PRO VÝVOJ

Při vývoji mé aplikace pro prezentaci návrhů prvků zastínění a techniky stínění (PZTS) je klíčový výběr správného VR headsetu. Různé modely nabízejí odlišné vlastnosti, které ovlivňují uživatelský zážitek i efektivitu vývoje. V této kapitole se zaměříme na srovnání klíčových parametrů a určení ideálního VR zařízení pro dosažení optimálního výsledku.

5.1 Varjo Aero

Varjo Aero je špičkový VR headset určený pro profesionální uživatele, kteří vyžadují nejvyšší možnou vizuální kvalitu. Poskytuje extrémně vysoké rozlišení a ostrý obraz, s důrazem na věrnost barev a detailů, a je určen pro použití v náročných aplikacích, jako je design, architektura a simulace. Mezi jeho výhody patří absolutně špičková vizuální kvalita, extrémně ostrý obraz a věrné barvy, vysoký komfort při nošení a profesionální přesnost sledování. Nevýhodou je velmi vysoká cena, nutnost výkonného počítače, drátové připojení a zaměření na úzkou skupinu profesionálů. Cena se pohybuje v řádech stovek tisíc korun.

5.2 HTC Vive Pro 2

HTC Vive Pro 2 je profesionální VR headset určený pro náročné uživatele a profesionály. Nabízí extrémně vysoké rozlišení a široké zorné pole pro pohlcující VR zážitek a vyžaduje připojení k výkonnému počítači. Mezi jeho výhody patří extrémně vysoké rozlišení a ostrý obraz, široké zorné pole, přesné sledování pohybu a profesionální kvalita zobrazení. Nevýhodou je vysoká cena, nutnost výkonného počítače a drátové připojení, které omezuje volnost pohybu. Cena se pohybuje v řádech desítek tisíc korun.[13]

5.3 Meta quest 3

Meta Quest 3 je nejnovější bezdrátový VR headset od společnosti Meta, který nabízí vylepšené rozlišení, výkon a možnosti smíšené reality. Je určen pro širokou škálu uživatelů, od hráčů po profesionály, kteří hledají flexibilní a vysoce kvalitní VR zážitek. Mezi jeho výhody patří vysoké rozlišení a ostrý obraz, bezdrátové provedení pro maximální svobodu pohybu, pokročilé možnosti smíšené reality a velká knihovna her a aplikací. Nevýhodou může být pro některé uživatele vyšší cena a omezená výdrž baterie. Cena se pohybuje v řádech desetitisíců korun, v závislosti na kapacitě úložiště.[14]

5.4 Zdůvodnění

Meta Quest 3 se pro mou aplikaci jeví jako ideální volba z několika klíčových důvodů. Jedním z nejpřesvědčivějších je jeho vlastní operační systém založený na Androidu, což eliminuje nutnost připojení k výkonnému PC a umožňuje spouštět aplikaci přímo na headsetu. Tato samostatnost je zásadní pro flexibilitu a mobilitu, což je při prezentacích návrhů prvků zastínění a technik stínění (PZTS) přímo u klienta velkou výhodou.

Dalším významným faktorem je možnost plynulého přepínání mezi čistě virtuální realitou a augmentovanou realitou. Tato funkce je nejen inovativní, ale také praktická. Pro klienty, kteří trpí kinetózou ve VR, nabízí augmentovaná realita komfortnější alternativu, kde se virtuální prvky integrují do reálného prostředí.

Absence omezujících prvků, jako jsou snímací stanice nebo kabely, výrazně zvyšuje uživatelský komfort a usnadňuje interakci s virtuálním prostorem. Tato svoboda pohybu je klíčová pro pohlcující a přirozenou prezentaci návrhů.

Díky své relativně nízké ceně je Meta Quest 3 ideální volbou pro malé firmy či podnikatele, pro které pořízení tohoto zařízení nebude představovat významnou finanční zátěž.

Nicméně, skutečnost, že Meta Quest 3 běží na Androidu, přináší i výzvy. V rámci vývoje je nutné vytvářet samostatnou verzi aplikace, čímž se zvyšuje náročnost na multiplatformní kompatibilitu. I přes tuto komplikaci převažují výhody Meta Quest 3, díky kterým si myslím, že je pro aplikaci ideální.[14]

6 PŘENESENÍ PŮDORYSU DO 3D MODELU

Převod 2D půdorysu do 3D modelu je klíčovým krokem při vizualizaci návrhů. Tato kapitola se zaměřuje na proces transformace plochého 2D zobrazení na interaktivní 3D prostor, který umožňuje lepší pochopení a prezentaci. Představím zde metodu, kterou používám, jež klade důraz na jednoduchost a efektivitu, a která se liší od tradičních přístupů.

6.1 Běžně užívaný postup

Přenesení 2D půdorysu do 3D modelu začíná importem 2D obrázku nebo vektorového souboru do 3D modelovacího softwaru, kde je následně provedena kalibrace pro zajištění shody rozměrů s realitou. Poté se na základě tohoto půdorysu vytvoří základní 3D struktura, zahrnující stěny, podlahy a stropy, přičemž se 2D půdorys používá jako šablona pro vytahování stěn do 3D prostoru. Následuje modelování detailů, jako jsou okna, dveře a schodiště, které se umísťují podle jejich pozice v půdorysu. Pro zvýšení realismu se přidávají textury a materiály, které definují povrchový vzhled a fyzikální vlastnosti objektů. Na závěr se model osvětluje virtuálními zdroji světla a provádí se renderování, proces výpočtu obrazu z 3D modelu, jehož trvání závisí na složitosti modelu a nastavení renderování. Důležitými faktory jsou přesnost půdorysu, možnosti použitého softwaru a zkušenosti modeláře [15][16]

6.2 Metoda používaná v programu

Můj postup pro převod 2D půdorysu do 3D modelu začíná extrakcí obrysů stěn z obrázku. K tomu využívám editační software, jako je Inkscape, který umožňuje oddělit černé linie stěn od bílého pozadí. Tímto způsobem generuji SVG soubor, který obsahuje pouze obrysy stěn.

Následně importuji SVG soubor do 3D modelovacího prostředí jako je Blender. Zde upravím rozměry importovaného obrysů podle měřítka uvedeného na půdorysu nebo pomocí referenčních rozměrů známých objektů, jako jsou dveře nebo okna.

Při práci v Blenderu, převedu SVG obrys na 3Dmesh, což mi umožní manipulovat s ním jako s 3D objektem. Následně odstraním nepotřebné prvky, jako jsou textové popisky místností, které nechci zahrnout do 3D modelu. Dodatečný popis postupu viz příloha

V posledním kroku nastavím požadovanou výšku stěn, čímž vytvořím základní 3D model interiéru. Textury a materiály pro realističtější vzhled modelu mohu aplikovat buď v Blenderu nebo přímo v mé aplikaci, v závislosti na potřebách a preferencích uživatele. [17]

Blender nabízí větší množství úprav a způsobů editace modelu, ale je komplikovanější a pro nezkušeného uživatele budou funkce v aplikaci naprosto dostačující.[17][18][19]

7 VÝBĚR VHODNÉ PLATFORMY PRO VÝVOJ

Výběr vhodné platformy pro vývoj je zásadní pro úspěch jakéhokoli softwarového projektu a u VR aplikací to platí dvojnásob. V této kapitole se zaměřím na klíčové aspekty, které je třeba zvážit při výběru ideálního vývojového prostředí pro mou aplikaci. Proberu různé platformy, zhodnotím jejich výhody a nevýhody, a objasním, proč jsem se nakonec rozhodl pro konkrétní řešení.

7.1 Unity

Unity je univerzální herní engine, který se stal jedním z předních nástrojů pro vývoj VR aplikací a 3D vizualizací. Nabízí robustní sadu nástrojů pro tvorbu interaktivních 3D prostředí, intuitivní uživatelské rozhraní a rozsáhlou knihovnu assetů. Mezi jeho hlavní výhody patří snadná použitelnost, rozsáhlá komunita a široká škála pluginů, které rozšiřují jeho funkčnost. Umožňuje multiplatformní vývoj, včetně desktopových a VR platform, což je ideální pro projekty kombinující obě prostředí. Nevýhodou může být složitější správa velkých projektů a výkonová náročnost u graficky intenzivních aplikací. Unity nabízí různé plány předplatného, od bezplatné verze pro začátečníky až po profesionální plány pro komerční vývoj, s cenami lišícími se podle funkcí a potřeb.[21]

7.2 UN

Unreal Engine je další špičkový herní engine, který vyniká svou vysokou grafickou kvalitou a pokročilými funkcemi pro realistickou vizualizaci. Je oblíbený mezi vývojáři AAA her a profesionály v oblasti vizualizace. Jeho hlavní výhody zahrnují pokročilý vizuální editor, silný nástroj pro tvorbu materiálů a osvětlení a rozsáhlou podporu VR zařízení. Stejně jako Unity, podporuje multiplatformní vývoj, což je klíčové pro aplikace s desktopovým editorem a VR prezentací. Nevýhodou je strmější křivka učení a vyšší nároky na hardware. Unreal Engine nabízí bezplatnou verzi s omezenými funkcemi a licenční model založený na podílu z tržeb pro komerční projekty.[22]

7.3 Android studio

Android Studio je oficiální integrované vývojové prostředí (IDE) pro vývoj aplikací pro platformu Android, na které běží i Meta Quest 3. Poskytuje kompletní sadu nástrojů pro vývoj nativních aplikací, včetně editoru kódu, debuggeru, emulátoru a nástrojů pro profilování výkonu. Jeho hlavní výhodou je přímá integrace s platformou Android, což

umožňuje vývoj aplikací optimalizovaných pro výkon a využívajících specifické funkce zařízení. Je ideální pro vývoj aplikací určených výhradně pro Meta Quest 3, kde je vyžadována maximální optimalizace a využití hardwarových zdrojů. Nevýhodou je omezení na platformu Android a nutnost znalosti nativního vývoje. Android Studio je k dispozici zdarma.[23]

7.4 Zdůvodnění výběru

Po důkladném zvážení všech dostupných platform jsem se rozhodl pro Unreal Engine jako ideální volbu pro vývoj mé VR aplikace s integrovaným desktopovým editorem. Unreal Engine nabízí bezkonkurenční možnosti pro tvorbu vysoce kvalitních vizualizací, což je klíčové pro realistickou prezentaci návrhů prvků zastínění a technik stínění (PZTS). Jednou z největších předností Unreal Engine je jednoduchá práce s assety, což usnadňuje import a správu 3D modelů a textur, čímž se urychluje vývojový proces.

Dalším významným faktorem je možnost programování v C++, což poskytuje flexibilitu a kontrolu nad výkonem aplikace. Pro ty, kteří nemají hluboké znalosti programování, nabízí Unreal Engine unikátní systém Blueprintů, který umožňuje vizuální programování a vytváření komplexních interakcí bez nutnosti psát kód. Tento systém je ideální pro rychlé prototypování a iterace, což je neocenitelné při vývoji interaktivní VR aplikace.

Ačkoli Unity by byla také solidní volbou, především díky své rozsáhlé komunitě a podpoře multiplatformního vývoje, nemám osobní znalosti jazyka C# a preferuji jazyk C++. Android Studio by sice umožnilo vytvořit optimalizovanou verzi pro Meta Quest 3, ale komplikace spojené s vývojem desktopového editoru by byly příliš velké. Unreal Engine se tak ukázal jako nejlepší kompromis mezi výkonem, flexibilitou a snadností použití, což z něj činí ideální platformu pro můj projekt. Viz předchozí zdroje + přidat další

8 TVORBA ASSETŮ

Tvorba kvalitních assetů je klíčová pro pohlcující a realistický zážitek ve virtuální realitě. Tato kapitola se zaměří na proces tvorby 3D modelů a textur, které oživí virtuální prostředí mé aplikace. Probereme zde vhodné softwary, techniky a postupy, které používám pro dosažení optimálních výsledků.

8.1 Proč vytvářím Assety

Jedním z hlavních důvodů, proč se věnuji tvorbě vlastních assetů pro mou aplikaci, je omezená dostupnost kvalitních 3D modelů prvků zastínění a technik stínění (PZTS). Volně dostupné assety v této oblasti jsou téměř neexistující, a i placené zdroje nabízejí jen velmi omezený výběr. Navíc, často postrádají kompletní sady komponent, které jsou nezbytné pro realistické a detailní prezentace. Proto se tvorba vlastních assetů stává klíčovou pro zajištění vysoké kvality a komplexnosti mé aplikace.

8.2 Blender

Blender je bezplatný a otevřený 3D modelovací software, který se stal nepostradatelným nástrojem pro mnohé 3D umělce a vývojáře her. Nabízí komplexní sadu nástrojů pro modelování, animaci, simulace a rendering, což umožňuje tvorbu široké škály assetů pro VR aplikace. Hlavní výhodou je jeho bezplatná dostupnost, což je ideální pro začátečníky i profesionály s omezeným rozpočtem. Díky silné komunitě uživatelů a vývojářů nabízí Blender neustále se rozšiřující škálu pluginů a nástrojů. Nevýhodou může být strmější křivka učení pro začátečníky a občasná nestabilita u náročných projektů. Blender je zdarma.[18]

8.3 Autodesk Maya

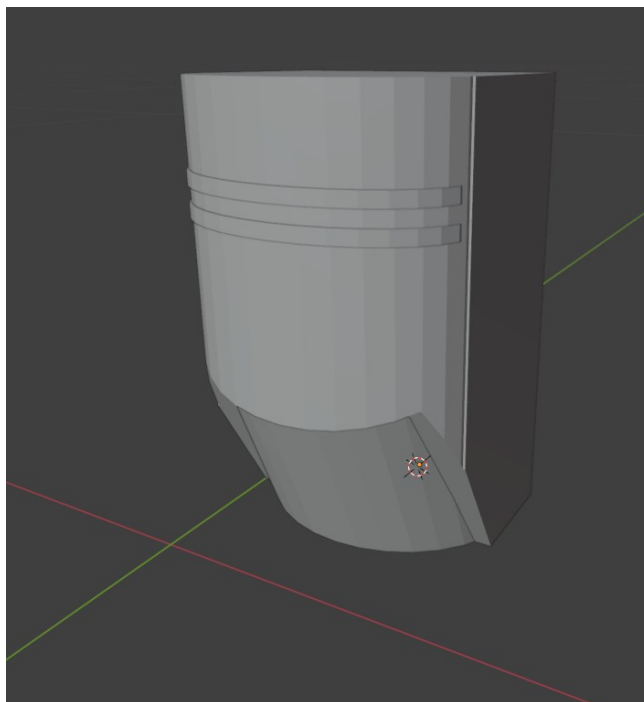
Autodesk Maya je profesionální 3D animační software, který se používá v mnoha odvětvích, včetně filmového průmyslu a vývoje her. Nabízí pokročilé nástroje pro modelování, animaci a rendering, to z něj činí ideální volbu pro tvorbu vysoce kvalitních assetů. Maya se vyznačuje silnými animačními funkcemi, což je klíčové pro tvorbu realistických postav a animací. Výhodou je jeho široké využití v průmyslu, zaručující kompatibilitu s ostatními nástroji a pipeline. Nevýhodou je vysoká cena a náročnost na hardware. Maya se prodává formou předplatného s cenami v řádu desítek tisíc korun ročně.[17]

8.4 ZBrush

ZBrush je digitální sochařský nástroj, který se specializuje na tvorbu organických modelů s vysokým detailem. Je ideální pro tvorbu postav, zvířat a dalších organických assetů, kde je vyžadována vysoká úroveň detailů. ZBrush umožňuje intuitivní sochařský přístup k modelování, což usnadňuje tvorbu komplexních organických tvarů. Výhodou je jeho specializace na organické modelování a vysoká úroveň detailů, které lze dosáhnout. Nevýhodou může být specifický workflow a nutnost použití jiného softwaru pro retopologii a texturování. ZBrush se prodává formou trvalé licence nebo předplatného s cenami v řádu desítek tisíc korun.[27]

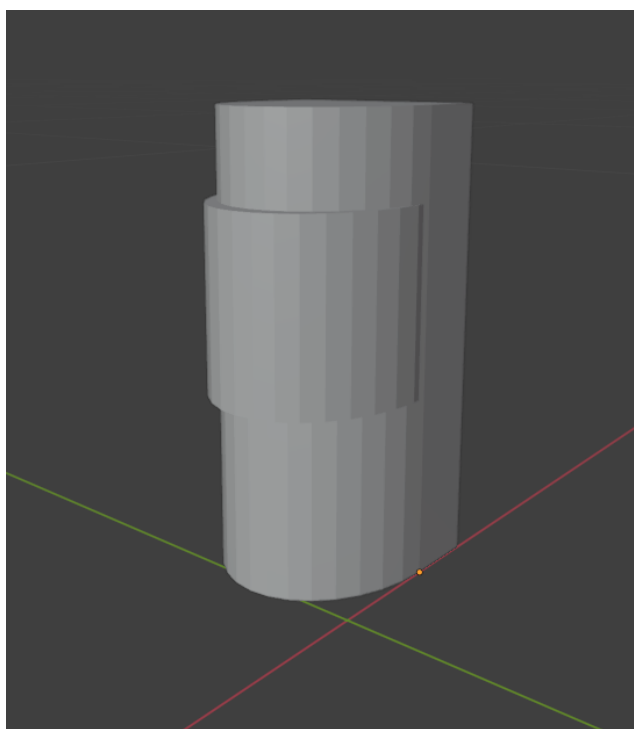
8.5 Moje volba + ukázky

Volba Blenderu pro tvorbu assetů pro mou VR aplikaci byla jednoznačná z několika klíčových důvodů. Především Blender je bezplatný a open-source software, což je pro můj projekt, s ohledem na rozpočet, zásadní. Tato dostupnost mi umožňuje soustředit se na vývoj aplikace samotné, aniž bych se musel starat o licenční poplatky. Dalším významným faktorem je obrovská komunita uživatelů a vývojářů, která stojí za Blenderem. Tato komunita neustále vytváří nové pluginy, tutoriály a návody, což mi usnadňuje učení a rozšiřování mých dovedností. Dlouhodobě považuji Blender za ideální volbu pro 3D modelování a mám s ním bohaté předchozí zkušenosti, což mi umožňuje efektivně pracovat na tvorbě kvalitních assetů. Kombinace těchto faktorů dělá z Blenderu ideální nástroj pro můj projekt.



Obrázek 1 PIR detektor

(zdroj: vlastní)



Obrázek 2 MW detektor

(zdroj: vlastní)

8.6 Co vše se modelovalo

Modelování v rámci projektu se soustředilo na tvorbu zástupných prvků, které mohou být opakovaně použity pro různé typy zařízení. Tímto přístupem bylo dosaženo efektivity a modularity při vizualizaci široké škály komponent.

Ústředna	Malá	Střední	Velká	
PIR	klasik1	Petimunity	Antimastk	
MW	klasik1	Petimunity	Antimastk	
Dual	klasik1	Petimunity	Antimastk	
Magnet	A1	A2	B1	B2
Tříšťák	klasik1	klasik2		
Kouřové	optické	teplotní	kombinované	
Klávesnice	tlačítka	display		
sirény	vnitřní	venkovní	Venkovní s majkákem	
Tísňové tlačítko	klasik1	klasik2		
Kamera	Kamera pevná	Kamera 360	Atrapa pevná	atrapa 360
čtečka	klasik1	Klasik2		

Obrázek 3 List komponentů

(zdroj: vlastní)

9 TVORBA APLIKACE – ZÁKLADNÍ NASTAVENÍ A VLOŽENÍ DAT

Po důkladném výběru vývojového prostředí bylo přistoupeno k samotné realizaci aplikace v Unreal Engine 5. Tato kapitola se zaměřuje na počáteční fáze vývoje, které zahrnovaly základní konfiguraci projektu, nastavení exportních parametrů pro různé platformy a implementaci klíčových mechanismů pro správu dat a objektů ve virtuálním prostředí. Byly zde položeny základy pro efektivní fungování aplikace jak v desktopovém editoru, tak ve virtuální realitě

9.1 Instalace a nastavení UnrealEngine 5

Po pečlivém výběru Unreal Engine 5 jako vývojové platformy bylo započato se samotnou implementací aplikace. Po instalaci bylo přistoupeno k výběru předpřipravené šablony, konkrétně VR Template, která posloužila jako základ pro další směřování vývoje.

Důležitým krokem před samotným vývojem bylo nastavení projektu pro různé typy exportu. Cílem bylo zajistit funkčnost finální aplikace a její bezproblémové vydání. Jelikož bylo v úmyslu vytvořit rozhraní jak pro desktop 3S, vyvstala nutnost separátního exportu pro operační systémy Windows (desktop) a Android (Meta Quest3S).

Ačkoli UnrealEngine v základu podporuje export pro Android, primární zaměření na export pro Windows přineslo při přípravě na Android několik komplikací. Jedním z hlavních problémů bylo, že aktuální verze UnrealEnginu a to 5.3, v té době vývoje neobsahovala plně aktualizované ovladače a potřebné nástroje pro bezproblémový export pro Android. To si vyžádalo značné úsilí při zprovoznění exportního procesu. Většina počítačů bez předchozí přípravy postrádá správné SDK launchery pro export. Chyběly také nejnovější verze Android Studio a specifické starší verze Javy, na kterých export fungoval, neboť na nejnovějších verzích neběžel. Bylo tedy nutné shromáždit a zkompletovat kompatibilní verze všech potřebných aplikací, aby bylo vůbec možné aplikaci úspěšně exportovat. Po překonání těchto překážek bylo dosaženo schopnosti exportovat aplikaci pro obě platformy. V přílohách je k dispozici exportní dokumentace, kterou lze nahrát do UnrealEnginu pro usnadnění exportu s již nastaveným vlastním nastavením.

9.1.1 Zvolená komprese

Pro kompresi textur ve vyvíjené aplikaci byl zvolen formát ETC2 (Ericsson TextureCompression 2) namísto jiných dostupných možností, jako jsou ASTC

(AdaptiveScalableTextureCompression) nebo DXT (S3 TextureCompression). Toto rozhodnutí bylo podloženo několika klíčovými faktory, především pak optimalizací pro cílovou platformu Android a zajištěním široké kompatibility s headsetem Meta Quest 3.

ETC2 představuje standardizovaný kompresní formát pro OpenGL ES 3.0 a novější, což znamená, že je nativně podporován většinou zařízení s operačním systémem Android. Tato nativní podpora zaručuje bezproblémové načítání a zobrazování textur na široké škále hardwaru, včetně mobilních GPU, které jsou součástí samostatných VR headsetů, jako je právě Meta Quest 3.

Formát DXT (S3TC) je primárně navržen pro desktopové grafické karty a jeho efektivní podpora na mobilních zařízeních není zaručena. Využití DXT by mohlo vést k nutnosti dekomprese textur v reálném čase, což by zvýšilo nároky na paměť i výpočetní výkon zařízení.

Ačkoli ASTC je modernější a flexibilnější formát nabízející vynikající kvalitu obrazu při různých kompresních poměrech a je podporován na novějších mobilních GPU, jeho podpora nemusí být tak univerzální jako u ETC2, zejména na starších nebo méně výkonných Android zařízeních. Meta Quest 3 sice podporuje i ASTC, ale volba ETC2 poskytuje širší kompatibilitu a větší jistotu, že vizualizace bude fungovat optimálně na cílovém hardwaru. Cílem bylo zajistit požadovaný výkon a vizuální kvalitu při zachování rozumné velikosti výsledné aplikace.

9.2 Výběr způsobu tvorby

Po dokončení konfigurace projektu a volbě vhodné kompresní metody bylo přistoupeno k přípravné fázi samotného vývoje. Bylo nutné definovat celkovou koncepci a architekturu aplikace, s ohledem na požadovanou funkčnost pro desktopové prostředí a virtuální realitu. Nabízely se tři základní možnosti řešení:

- Vývoj dvou samostatných aplikací, jedné pro desktop a jedné pro VR. Tato varianta by znamenala duplikaci vývojového úsilí a komplikace při správě a aktualizaci.
- Využití Unreal Engine pouze jako editoru a vývoj samostatné aplikace pro VR headset, která by sloužila výhradně jako prohlížeč. Tento přístup by omezil flexibilitu a možnost přímé interakce s editorem.

- Vývoj multifunkční aplikace, která by dynamicky rozpoznávala připojené zařízení PC nebo VR headsetu a na základě toho načítala odpovídající rozhraní, editor pro PC a prohlížeč pro VR

Byla zvolena třetí, multifunkční varianta, která se jeví jako nejefektivnější a nejvýhodnější. Toto řešení přináší značné výhody v průběhu celého vývojového cyklu. Primárně eliminuje nutnost vyvíjet a udržovat dvě separátní aplikace, což zjednodušuje správu kódu, knihoven, souborů, assetů a ukládacích souborů. Veškeré změny a aktualizace lze provádět v jediném projektu, což výrazně snižuje časovou náročnost a riziko vzniku chyb.

Dále tento přístup umožňuje plynulejší pracovní postup sjednocení dat. Modely, textury a další assety jsou sdíleny napříč oběma režimy, což zajišťuje konzistentní vizuální zážitek. Vývojář může snadno testovat změny provedené v desktopovém editoru přímo ve VR prohlížeči, což urychluje iterativní proces a umožňuje okamžitou zpětnou vazbu na designová rozhodnutí. To je klíčové pro optimalizaci scény a zajištění komfortního VR zážitku, kde i drobné změny v osvětlení nebo rozložení prostoru mohou mít významný dopad.

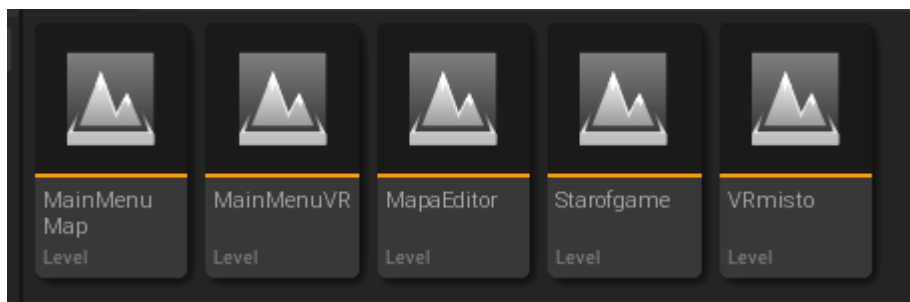
Navíc, multifunkční aplikace poskytuje lepší škálovatelnost pro budoucí rozšíření. Případné přidání nových funkcionalit, interaktivních prvků nebo typů vizualizací lze integrovat do jednoho uceleného systému, aniž by bylo nutné řešit synchronizaci mezi dvěma oddělenými projekty. Tato flexibilita je zvláště cenná pro projekt, jehož cílem je poskytnout komplexní nástroj pro vizualizaci bytových prostor s interaktivními prvky PZTS. V neposlední řadě pro koncového uživatele je tato varianta uživatelsky přívětivější, neboť k dispozici je jedna aplikace, která se adaptuje na jeho aktuální potřeby a hardware.

9.3 Tvorba levelů

S nastavením projektu a rozhodnutím pro multifunkční aplikaci bylo přistoupeno k návrhu a rozvržení herních úrovní (levelů). V kontextu Unreal Engine a jeho využití pro architektonickou vizualizaci se level chápe jako samostatná scéna nebo prostředí, ve kterém se uživatel pohybuje a interaguje s objekty. Tyto levely obsahují veškerou geometrii, osvětlení, materiály a interaktivní prvky.

Pro efektivní správu a optimalizaci bylo rozhodnuto o rozvržení levelů, které umožní plynulou práci v obou režimech aplikace. Budou vytvořeny specializované levely pro editor,

které budou sloužit pro načítání, zapínání a samostatnou tvorbu scény v desktopovém prostředí. Dále bude implementován společný level, jehož úkolem bude logické rozřazení a detekce, zda se jedná o desktopové prostředí či VR headset. V závislosti na detekovaném zařízení pak bude načten odpovídající level. Pro režim virtuální reality budou připraveny levels, které budou sloužit čistě k prohlížení vizualizace včetně základního menu a interaktivních prvků.



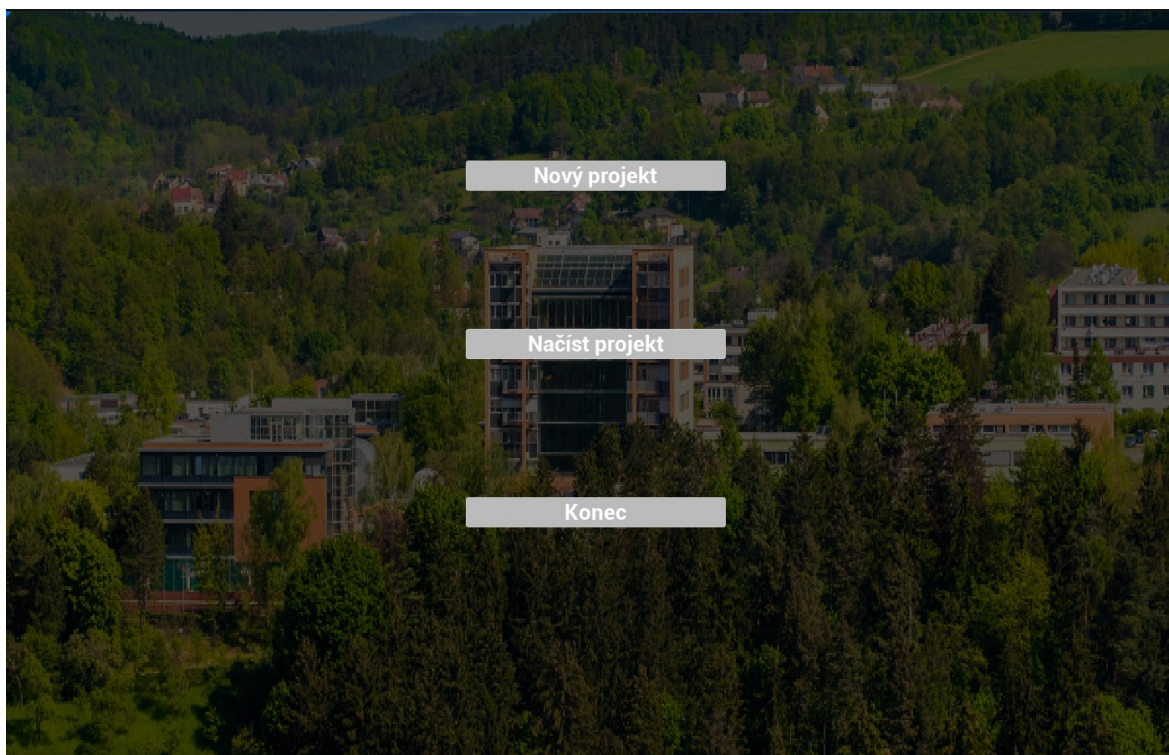
Obrázek 4 Levely

(zdroj: vlastní)

9.3.1 Editor levely

Pro efektivní práci v desktopovém režimu byly vytvořeny dva hlavní levelsy editoru, které zajišťují plynulý průběh práce od úvodního nastavení až po detailní tvorbu scény.

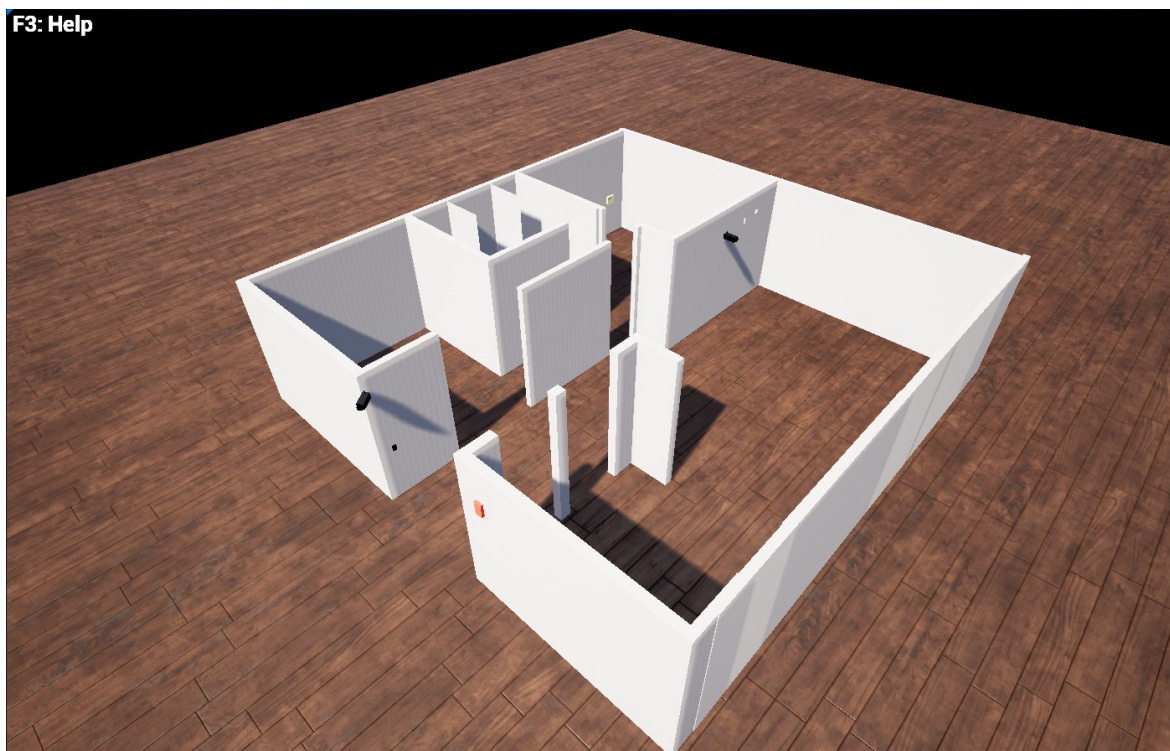
Úvodní Menu (Main Menu Level) slouží jako vstupní brána do aplikace, která se automaticky načte po rozpoznání, že je program spouštěn v desktopovém prostředí. Uživatelé jsou zde přehledně prezentovány klíčové možnosti pro zahájení práce s vizualizací. K dispozici jsou funkce jako Začít nový projekt, která umožňuje iniciovat tvorbu zcela nové 3D scény, nebo Načíst projekt, která slouží k opětovnému otevření a pokračování v již rozpracovaných vizualizacích. Samozřejmostí je i možnost Ukončit program, jež zajišťuje řádné vypnutí aplikace.



Obrázek 5 Main menu level

(zdroj: vlastní)

Hlavní Editor (Editor Map Level) začíná po výběru možnosti začít nový projekt nebo načíst projekt. Uživatel je poté přeměrován do hlavního levelu editoru. Tato mapa představuje centrální pracovní plochu, kde se odehrává podstatná část celého tvůrčího procesu. Zde je primárně umístěn základní objekt, ať už se jedná o předem vymodelovaný půdorys budovy nebo o objekt rodinného domku či menšího bytu, vytvořený na základě půdorysných dat, pro které je aplikace specificky navržena. V tomto prostředí se dále provádí komplexní osazování scény. Jde o integraci prvků poplachových zabezpečovacích a tísňových systémů a elektronických požárních systémů, to je klíčové pro specializované vizualizace chytrých domácností a bezpečnostních řešení. Navíc je zde možné scénu obohatit o interiérové a exteriérové prvky, jako jsou okna, dveře, nábytek a dekorace, která výrazně přispívá k celkovému realismu a estetice výsledné vizualizace. Celkovým cílem je vytvořit co nejpříjemnější a nejvěrohodnější zážitek z budoucí prohlídky.



Obrázek 6 Editor map level

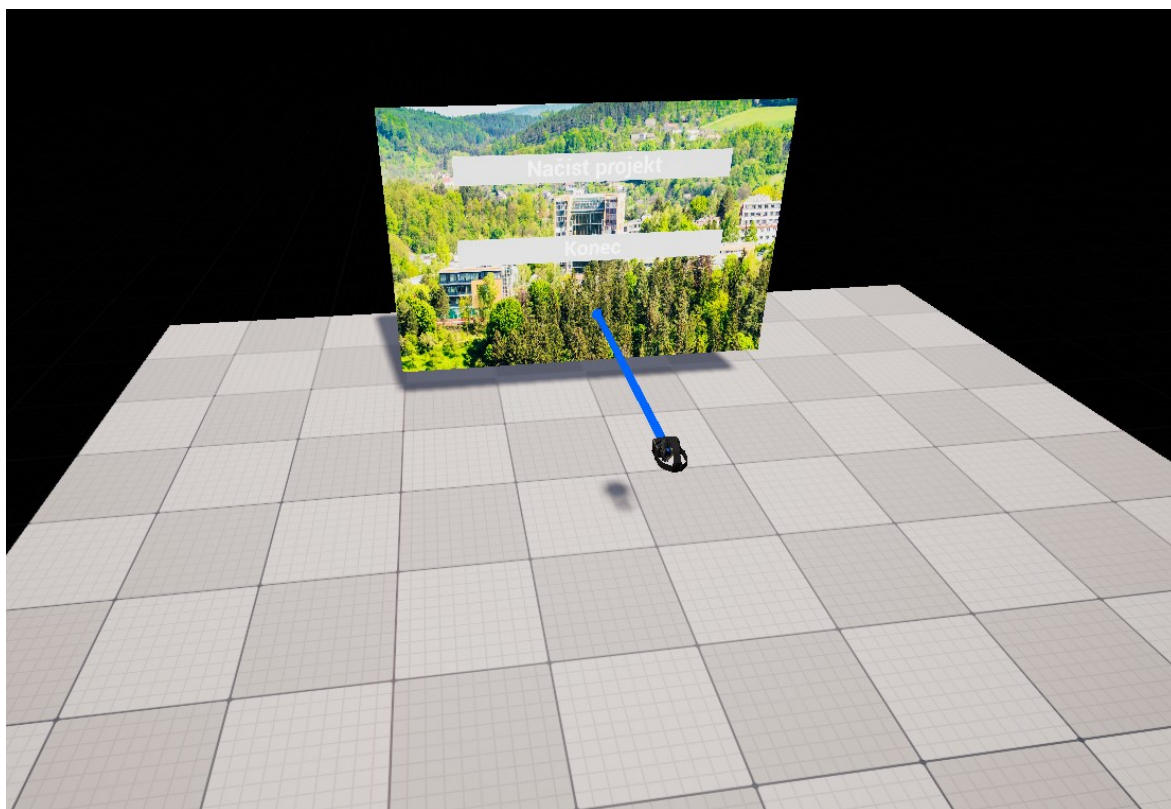
(zdroj: vlastní)

9.3.2 VR levely

Pro zajištění pohlcujícího a interaktivního zážitku ve virtuální realitě byly v aplikaci navrženy dva VR levely. Po spuštění aplikace a detekci připojeného VR headsetu je uživatel přesměrován do úvodního VR menu, které slouží jako vstupní bod pro prohlížení vizualizací.

Úvodní VR Menu (VR Main Menu Level) level je minimalistický a jeho hlavním účelem je zprostředkovat uživateli možnost výběru a načtení existujících projektů. Level jako takový neobsahuje žádnou komplexní geometrii ani složité prvky, ale je tvořen primárně interaktivním widgetem. Tento widget umožňuje uživateli interakci prostřednictvím VR ovladačů, přičemž jsou k dispozici dvě hlavní volby:

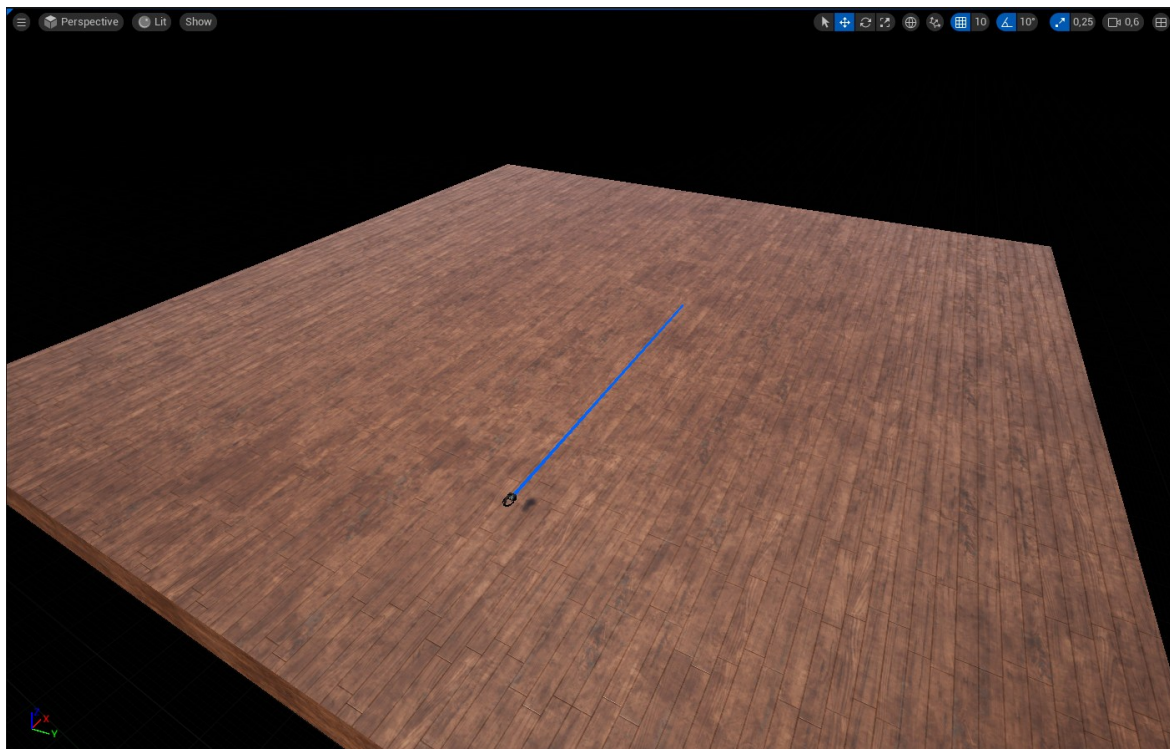
- Načíst projekt: Tato funkce umožňuje načíst jeden z dříve uložených a připravených projektů 3D vizualizací. Po výběru projektu je uživatel plynule přesunut do samotného prohlížečského VR levelu, kde se již nachází načtená vizualizace.
- Ukončit aplikaci: Tato volba slouží k řádnému ukončení programu z prostředí virtuální reality



Obrázek 7 VR main menu level

(zdroj: vlastní)

Po načtení zvoleného projektu z úvodního VR menu je uživatel přenesen do hlavního VR prohlížeče. Je důležité poznamenat, že tento level samotný je prázdný v tom smyslu, že neobsahuje předem nadefinovanou architekturu nebo objekty. Místo toho se do něj dynamicky načítají kompletní vymodelované scény z vybraného projektu, které byly vytvořeny v desktopovém editoru. To zajišťuje modularitu a umožňuje uživateli procházet a zkoumat různé bytové prostory bez nutnosti opouštět VR prostředí. V tomto levelu se uživatel může volně pohybovat prostředím a interagovat s interaktivními prvky, které byly do scény přidány, jako jsou například prvky PZTS nebo nábytek. Hlavní důraz je zde kladen na plynulost a jednoduchost.



Obrázek 8 VRmisto

(zdroj: vlastní)

9.4 Herní módy

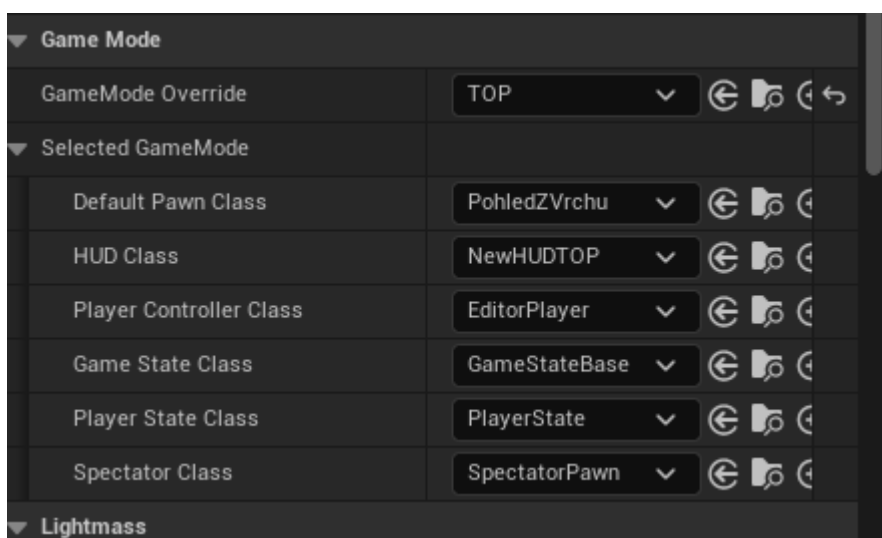
Po úvodním načrtnutí architektury a rozvržení levelů bylo dalším krokem vytvoření a konfigurace Game Modů pro jednotlivé části aplikace. Game Mode v Unreal Engineu definuje pravidla a chování hry nebo interaktivní aplikace pro konkrétní level. Spravuje nastavení, jako jsou třídy hráče (Pawn), ovladače (PlayerController), HUD (Head-Up Display) a další specifickou logiku, která se vztahuje k danému levelu.

Pro VR režim aplikace nebylo nutné vytvářet vlastní Game Mode, jelikož bylo využito předpřipraveného VR templatu v Unreal Engineu, který již obsahuje optimalizovaný Game Mode pro virtuální realitu. Tento Game Mode je přizpůsoben pro ovládání a interakce typické pro VR prostředí, čímž je zajištěna plynulost a komfort uživatelského zážitku.

Pro editorové levely, konkrétně pro Editor Map Level, byl vytvořen vlastní Game Mode nazvaný Game Mode TOP. Tento Game Mode je primárně zodpovědný za nastavení základních tříd a výchozích hodnot pro daný level, nikoli však za přímou správu interakcí či

načítání specifických prvků, což je řešeno primárně skrze Pawn. Zahrnuje nastavení pro základní typy ovladačů a postav, které budou v editoru používány.

Kromě vytvoření Game Modů bylo také zapotřebí nastavit základní parametry každého levelu, které ovlivňují jejich chování a výkon. Tyto parametry zahrnují například nastavení osvětlení, post-processing efekty, kolizní vlastnosti a další, které jsou klíčové pro optimalizaci jak pro real-time VR, tak pro desktopové prostředí.



Obrázek 9 Nastavení světa

(zdroj: vlastní)

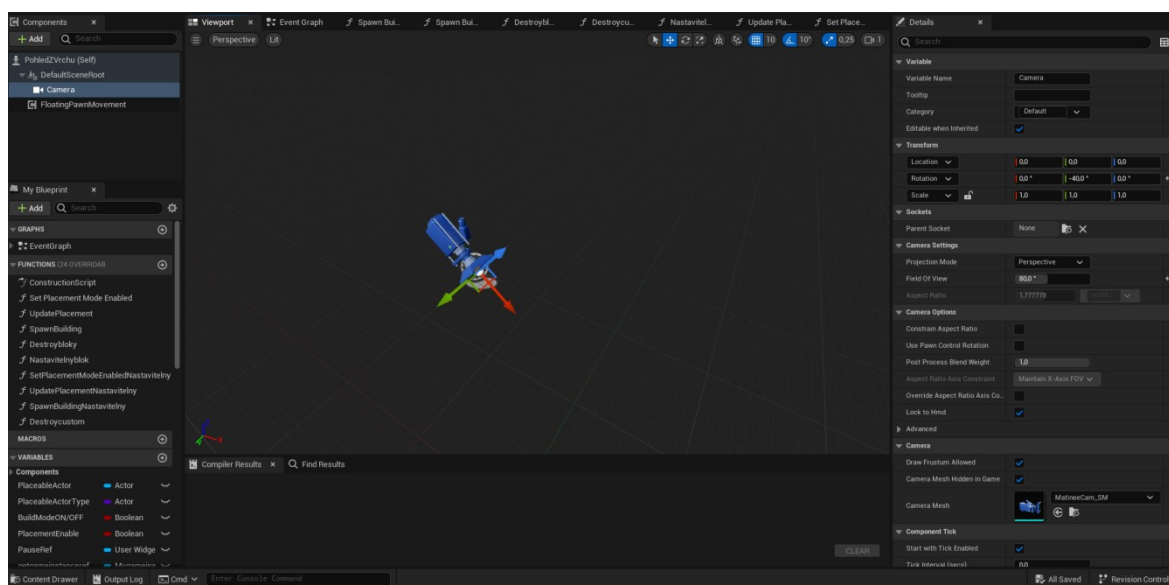
9.5 Vytvoření PawnClass

Dále bylo přistoupeno k vytvoření hratelné postavy, respektive hratelného objektu, který zprostředkovává uživateli pohled na scénu a umožňuje interakci s ní. Tato postava, nazvaná ohled z vrchu, je esenciálně Pawn, do kterého byla umístěna pouze kamera, bez jakéhokoliv viditelného tělesného objektu nebo meshe. Cílem bylo dosáhnout pohledu shora, podobně jako je tomu v herní sérii TheSims nebo v jiných editačních nástrojích.

Pawn v Unreal Engineu představuje základní třídu pro jakýkoli objekt, který může být ovládán hráčem nebo AI. V případě Pohledu z vrchu se jedná o neviditelnou entitu, která je schopna přijímat vstup od uživatele a reagovat na něj. Pečlivě byla napozicována kamera v rámci tohoto Pawna tak, aby odpovídala požadovanému záměru a poskytovala ideální zorné pole pro práci v editoru. Důkladné nastavení pozice a orientace kamery je klíčové pro

intuitivní a efektivní manipulaci s 3D prostředím, neboť přímo ovlivňuje, jak uživatel vnímá a interaguje se scénou.

Pawn Pohled z vrchu slouží jako centrální bod pro veškerou interakci a logiku editoru. Z tohoto Pawna jsou řízeny klíčové funkce, jako je pohyb kamerou po scéně (posun, rotace, zoom), výběr objektů, jejich transformace (posun, rotace, změna měřítka) a správa uživatelských vstupů z klávesnice a myši. Veškeré příkazy, které uživatel zadává pro manipulaci se scénou, jsou zpracovávány právě prostřednictvím tohoto Pawna. Dále Pawn obsahuje logiku pro načítání a ukládání projektů, správu assetů a interakci s prvky uživatelského rozhraní editoru. Díky tomuto přístupu je zajištěna modularita a snadná rozšiřitelnost editoru, jelikož veškerá logika spojená s uživatelským ovládáním je koncentrována v rámci jediné třídy. To také umožňuje plynulý přechod mezi režimy zobrazení, neboť Pawn může dynamicky měnit své chování a ovládat se jak pro desktopový pohled, tak pro případné testování ve VR prostředí, pokud by bylo potřeba. Celkově Pawn Pohled z vrchu představuje páteř, přes kterou je řízena většina funkcionalit editoru nebo interakci s prvky uživatelského rozhraní. Díky tomuto přístupu je zajištěna modularita a snadná rozšiřitelnost editoru, jelikož veškerá logika spojená s uživatelským ovládáním je koncentrována v rámci jediné třídy.



Obrázek 10 Nastavení pawn

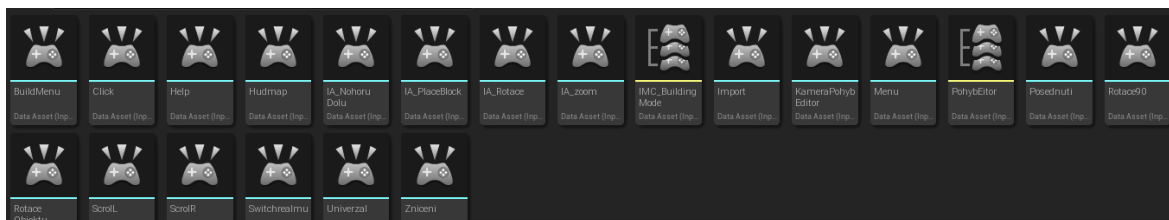
(zdroj: vlastní)

9.6 Přidání vstupů

Po vytvoření a konfiguraci Pawna Pohled z vrchu bylo dalším nezbytným krokem implementovat uživatelské vstupy, které umožní interakci s editorovým prostředím. Pro základní ovládání a manipulaci s prvky ve scéně jsou nezbytné vstupy, které jsou v UnrealEnginu spravovány prostřednictvím systému Input a Input MappingContexts.

Inputy v UnrealEnginu představují abstrakci uživatelských vstupů, jako jsou stisknutí kláves, pohyb myši nebo akce z herních ovladačů. Tyto fyzické vstupy jsou mapovány na logické akce, které pak mohou být zpracovány v rámci Blueprintů nebo C++ kódu. Díky tomuto systému je možné oddělit fyzické zařízení od herní logiky, což usnadňuje přizpůsobení ovládání a zajišťuje kompatibilitu s různými typy vstupních zařízení.

Pro správu těchto mapování jsou využívána Input MappingContexts. Tyto kontexty definují sady vstupních mapování, které jsou aktivní v určitých situacích nebo stavech hry. Například jeden kontext může být aktivní pro režim editoru, zatímco jiný pro režim prohlížení ve VR. Tato modularita umožňuje efektivní správu komplexního ovládání a zabráňuje konfliktům mezi různými sadami vstupů.

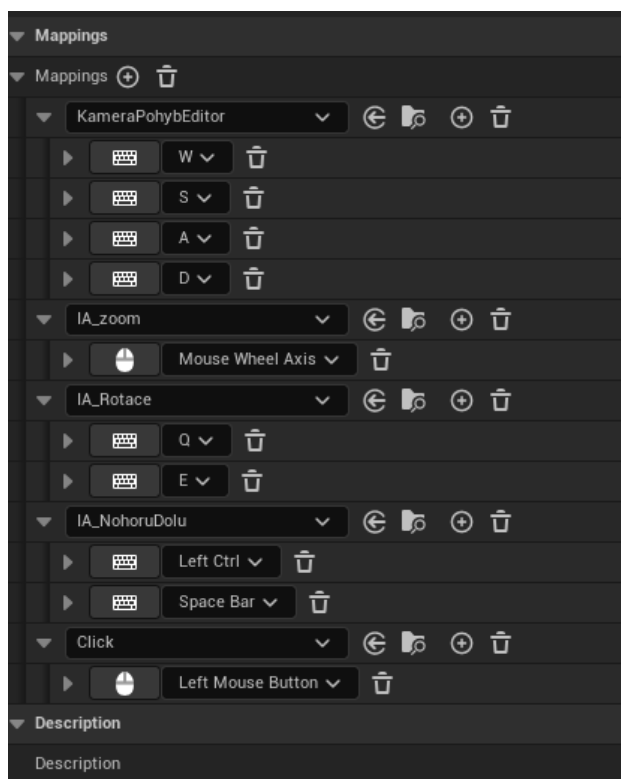


Obrázek 11 Inputy

(zdroj: vlastní)

Na začátku vývoje Blueprintu pro Pawna Pohled z vrchu byly definovány základní Inputy a Input MappingContexty. Tímto bylo zajištěno, že všechny potřebné vstupy pro ovládání kamery a interakci se scénou jsou správně připraveny. První byly implementovány inputy pro pohyb, zahrnující posun do stran, vpřed a vzad, a také pro zoomování, což jsou základní interakce nezbytné pro navigaci v editoru. Konkrétně byly vytvořeny Input MappingContexty jako Building Mode a Pohyb editor. Zpočátku bylo vytvořeno podstatně méně vstupů, které byly v průběhu vývoje postupně doplňovány dle aktuálních potřeb a požadavků na funkcionalitu. Na základě těchto definovaných Inputů a Input

MappingContextů pak v dalších krocích probíhá manipulace a ovládání dalších prvků v editoru.



Obrázek 12 Input mapping

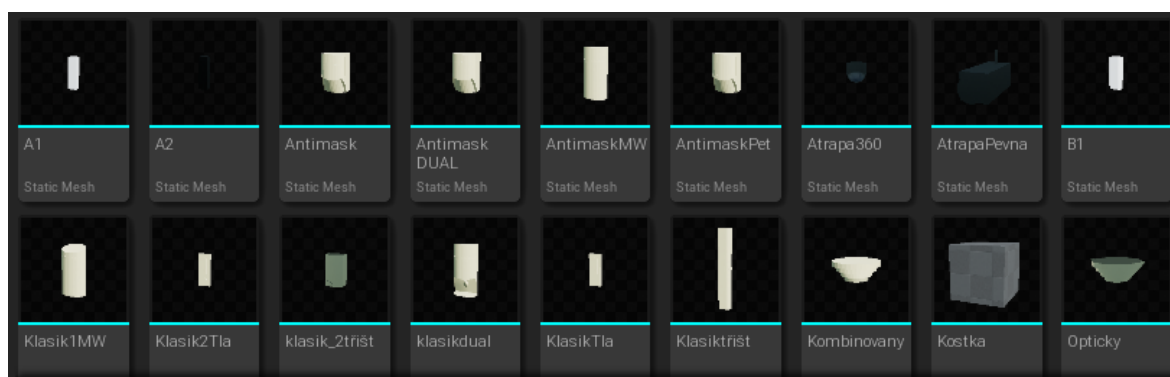
(zdroj: vlastní)

9.7 Vložení assetů

Implementace assetů v kontextu tohoto projektu primárně zahrnovala nahrávání vlastních 3D modelů vytvořených v Blenderu a integraci volně dostupných balíčků z obchodů s assety. Klíčovým důvodem pro tvorbu vlastních assetů byl nedostatek komplexních knihoven obsahujících modely prvků poplachových zabezpečovacích a tísňových systémů a elektronických požárních systémů. Volně dostupné assety v této oblasti byly téměř neexistující, a i placené zdroje nabízely jen velmi omezený výběr, často postrádající kompletní sady komponent nezbytné pro realistické a detailní prezentace.

Dříve vytvořené 3D modely PZTS a EPS prvků byly exportovány do formátu FBX a následně nahrány do UnrealEngine. Tímto bylo zajištěno, že všechny potřebné specializované objekty pro vizualizaci budou k dispozici v projektu.

Kromě vlastních modelů byl do projektu přidán Free FurniturePack ze obchodu. Tento balíček obsahuje různé modely nábytku a dekorací, které doplňují scénu a přispívají k vytvoření vizuálně atraktivního prostředí. Integrace těchto hotových assetů výrazně urychlila proces vybavování interiérů a exteriérů, jelikož se nebylo nutné zabývat modelováním každého jednotlivého kusu nábytku.

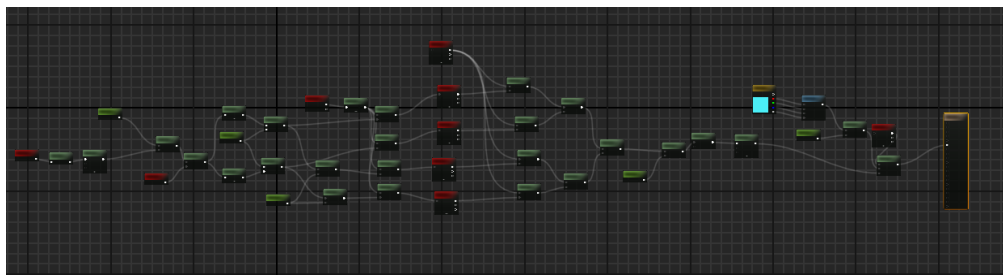


Obrázek 13 Modely

(zdroj: vlastní)

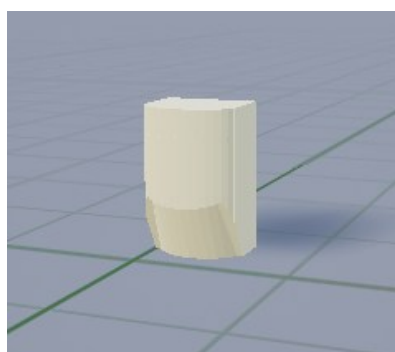
9.8 Tvorba barev

Pro vizuální definici prvků ve scéně bylo přistoupeno k tvorbě základních barev a materiálů. Ačkoli mohly být využity předem definované barevné palety, byla vytvořena i řada vlastních, velice jednoduchých barev, jako je červená, zelená, černá a bílá. Tyto barvy sloužily k rychlé aplikaci na objekty a pro základní vizuální odlišení. Kromě těchto jednoduchých barev byly vytvořeny i komplikovanější materiály, např. pro efekty jako highlighter, které slouží k vizuálnímu zvýraznění vybraných objektů ve scéně. Následně byly tyto barvy a materiály aplikovány na jednotlivé objekty, respektive na mnou vytvořené modely prvků. Tato flexibilita v definici barev a materiálů umožnila přesnější a účinnější vizualizaci jednotlivých prvků.



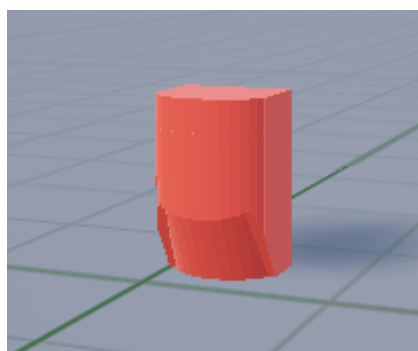
Obrázek 14 Barva zvýraznění

(zdroj: vlastní)



Obrázek 15 Bílé PIR

(zdroj: vlastní)



Obrázek 16 Červené PIR

(zdroj: vlastní)

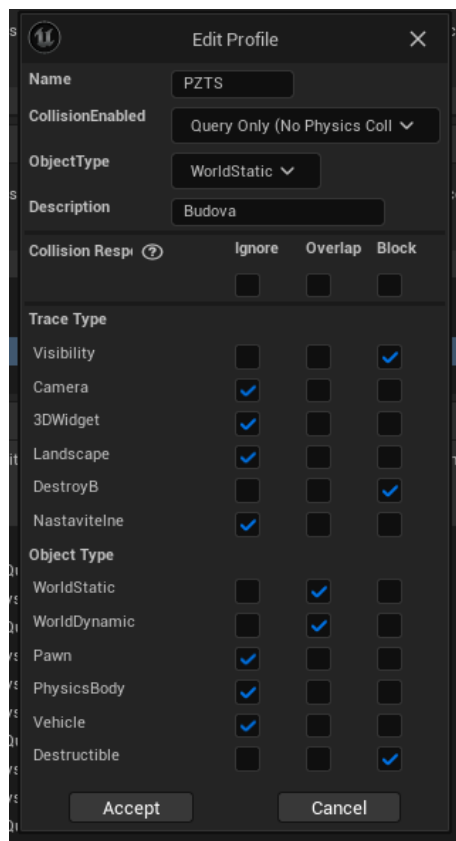
9.9 Tvorba kolizí

Dalším důležitým krokem nastavení TraceChannelů a Presetů, které jsou klíčové pro detekci kolizí a interakci objektů ve virtuálním prostředí. TraceChannel v UnrealEnginu je specifický kanál, který se používá k filtrování a detekci kolizí při trasování paprsků, což je technika, kdy se z určitého bodu ve scéně vystřelí neviditelný paprsek a sleduje se, s čím se setká. Každý objekt ve scéně může být nastaven tak, aby reagoval na jeden nebo více TraceChannelů. To umožňuje selektivní interakci, kdy například paprsek pro výběr objektu reaguje pouze na interaktivní prvky a ignoruje pozadí. V kontextu této aplikace byly nastaveny TraceChannelLandscape a později také DestroyBlock. Landscape, slouží k detekci interakcí s terénem nebo základovou plochou, zatímco DestroyBlock byl přidán pro specifické interakce související s destruktivními akcemi, jako je odstraňování prvků.

Presety kolizí jsou předdefinované sady nastavení kolizí, které lze aplikovat na objekty ve scéně. Usnadňují správu kolizních vlastností a zajišťují konzistenci chování objektů. Každý preset definuje, jak objekt reaguje na různé TraceChannely a jak se chová vůči jiným kolizním objektům např. zda je blokuje, překrývá nebo ignoruje. Pro potřeby této aplikace byly použity Presety PZTS a Budova.

- **Preset PZTS:** Tento preset byl nastaven pro všechny modely prvků poplachových zabezpečovacích a tísňových systémů. Zajišťuje, že tyto prvky správně reagují na trasování paprsků (například pro výběr a manipulaci) a zároveň mají definované kolizní chování s ostatními objekty ve scéně.
- **Preset Budova:** Tento preset byl aplikován na všechny stavební prvky, jako jsou stěny, podlahy a stropy. Jeho účelem je zajistit, že hráč nebo kamera neprochází skrze tyto objekty a že interakce s nimi je realistická.

Nastavení těchto TraceChannelů a Presetů je zásadní pro funkčnost interakce v editoru i VR prostředí, jelikož správně definované kolize umožňují přesný výběr objektů, detekci událostí a celkově věrohodné chování virtuálního světa.



Obrázek 17 Preset

(zdroj: vlastní)

9.10 Tvorba aktorů

Pro efektivní správu a manipulaci s interaktivními prvky ve scéně byla zvolena hierarchická struktura založená na koncepci parent actorů a child actorů. V Unreal Engineu je actor základní třída pro jakýkoli objekt, který může být umístěn do levelu. Může mít pozici, rotaci a měřítko a může obsahovat různé komponenty např. vizuální mesh, kolizní těleso, světlo nebo kamera.

Byl vytvořen mateřský actor, který slouží jako základ pro všechny prvky PZTS a EPS. Tento parent actor v sobě inkapsuloval společné vlastnosti a chování, které jsou pro všechny tyto prvky relevantní. Jednalo se o nastavení, jako je například základní vizuální mesh, výchozí barva a pole pro textový popis. Tímto přístupem bylo dosaženo centralizované správy a snadné modifikace společných atributů.

Z tohoto jediného parentactora pak bylo vytvořeno mnoho childactorů. Každý childactor zdědil vlastnosti a chování od svého parentactora, ale zároveň mu mohly být přiděleny specifické, unikátní atributy, které jej odlišovaly od ostatních. Například zatímco parentactor definoval, že všechny PZTS prvky mají mesh a barvu, konkrétní childactor pro pohybový detektor mohl mít specifický mesh pohybového detektoru a defaultní barvu nastavenou na bílou. Díky této hierarchii je zajištěna vysoká míra modularity, snadná rozšiřitelnost o nové typy prvků a efektivní správa vizuálních a funkčních vlastností napříč celou aplikací. Každému childactorovi bylo takto přiřazeno specifické vizuální ztvárnění a výchozí barva.[24]

9.11 Vytvoření data table

Dalším klíčovým krokem ve vývoji bylo zavedení datových tabulek, které slouží k efektivní správě a organizaci velkého množství strukturovaných dat. Datové tabulky představují efektivní způsob ukládání a načítání dat v Unreal Enginu, podobně jako tabulky v tabulkovém procesoru, kde každý řádek reprezentuje záznam a každý sloupec určitou vlastnost. Používají se pro uložení dat, která se nemění dynamicky během běhu aplikace, ale jsou potřeba pro konfiguraci nebo inicializaci objektů.

Do těchto datových tabulek byly vloženy informace o všech actorech, které reprezentují prvky PZTS a EPS. Každý záznam v tabulce obsahuje specifické vlastnosti daného typu prvku, jako je například odkaz na jeho 3D model, název, cena, stupeň zabezpečení, výrobce a může obsahovat další relevantní atributy.

Datové tabulky byly dále rozšířeny o modely nábytku, dveří a oken. Tímto způsobem bylo dosaženo centralizovaného úložiště pro veškeré vizuální assety, které se používají pro osazování scény. Díky datovým tabulkám je možné snadno přistupovat k těmto informacím a dynamicky vytvářet instance objektů ve scéně.

Hlavním účelem využití datových tabulek je jejich následné použití pro systém stavění, který je implementován v pozdějších fázích projektu. Systém stavění tak může efektivně načítat potřebné modely a jejich vlastnosti z datových tabulek a dynamicky je umisťovat do scény na základě uživatelských vstupů.

	Row Name	Název	Mesh
1	MagnetA1	MagnetA1	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/A1.A1'
2	MagnetA2	MagnetA2	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/A2.A2'
3	Anntimask	Antimask	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/A1.A1'
4	AntimaskDual	AntimaskDual	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/AntimaskDUAL
5	AntimaskMW	AntimaskMW	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/AntimaskMW.A
6	AntimaskPet	AntimaskPet	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/AntimaskPet.Ai
7	Atrapa360	Atrapa360	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/Atrapa360.Atra
8	AtrapaPevna	AtrapaPevna	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/AtrapaPevna.A
9	MagnetB1	MagnetB1	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/B1.B1'
10	MagnetB2	MagnetB2	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/B2.B2'
11	Display	Display	/Script/Engine.StaticMesh'/Game/Editor/FBXdokumenty/Display.Display

Obrázek 18 Data table1

(zdroj: vlastní)

Blueprint	LayerTraceChanne	cislo	Cena	Stupeň zabezpečen	Výrobce
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	1	300	2	Jablotron
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	2	350	2	Jablotron
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	3	1000	2	Jablotron
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	4	1500	2	Jablotron
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	5	1200	2	Jablotron
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	6	1800	3	XY
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	7	500		Hikvision
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	8	500		Hikvision
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	9	800	3	Honeywell
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	10	900	3	Honeywell
/Script/Engine.BlueprintGeneratedClass'/Game/Editor/PZTSprvky/PZTS_	Visibility	11	8500	3	Honeywell

Obrázek 19 Data table2

(zdroj: vlastní)

10 TVORBA APLIKACE PROSTŘEDNICTVÍM BLUEPRINTU

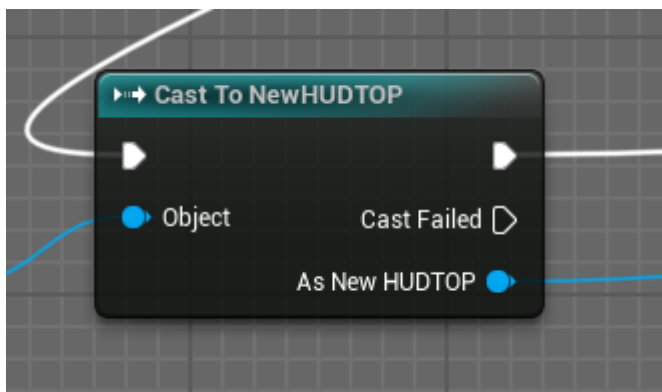
Další fáze vývoje aplikace byla primárně realizována prostřednictvím vizuálního skriptovacího systému Blueprint v Unreal Enginu. Nastavování a konfigurace v této části již nebylo tak komplexní jako v počátečních fázích, kdy byly definovány základní struktury projektu.

Blueprint představuje vizuální programovací jazyk v Unreal Enginu, který umožňuje vytvářet komplexní herní logiku a interaktivitu bez nutnosti psaní kódu v C++. Funguje na principu grafického propojení uzlů, z nichž každý reprezentuje určitou funkci nebo událost. Tyto uzly jsou pak spojovány vodiči, které určují tok dat a vykonávání logiky. Tento přístup výrazně urychluje vývoj a je přístupný i pro uživatele bez hlubokých programovacích znalostí.

10.1 Často používané bloky

Při tvorbě logiky aplikace v prostředí Blueprintu se opakovaně využívaly specifické bloky, které tvoří základ pro interaktivitu a funkcionalitu. Tyto uzly umožňují vizuální programování a propojování různých akcí a událostí. Detailnější seznam s popisem dalších často se vyskytujících bloků je možné nalézt v příloze této práce.

Cast To je v Blueprintu zásadní pro práci s hierarchií tříd a pro zajištění správné komunikace mezi různými typy objektů. Jeho primární funkcí je pokusit se převést obecný typ objektu na specifitější typ. Pokud je převod úspěšný, tj. pokud je objekt skutečně daného specifitějšího typu, tak je spuštěn výstup a navazující logika. V opačném případě, kdy převod selže, je aktivován výstup CastFailed. Tento uzel je nezbytný pro bezpečné volání funkcí nebo přístup k proměnným, které jsou specifické pro danou třídu, a zabraňuje chybám v případě, že objekt není očekávaného typu.



Obrázek 20 Cast to

(zdroj: vlastní)

Uzel Set je jedním z nejzákladnějších a nejčastěji používaných uzlů v Blueprintu, jehož účelem je nastavit nebo změnit hodnotu proměnné. Umožňuje přiřadit novou hodnotu proměnné specifického typu, ať už se jedná o jednoduché typy jako Boolean, Integer, Float, Vector, Rotator, String, nebo o komplexnější typy, jako jsou reference na objekty či struktury. Tento uzel je klíčový pro dynamickou manipulaci s daty v rámci Blueprintu, neboť umožňuje aktualizovat stavy objektů, ukládat uživatelské preference nebo řídit průběh logiky na základě změn podmínek. Vizuálně se uzel Set vyznačuje vstupním pinem pro vykonání a vstupním pinem pro samotnou hodnotu, která má být proměnné přiřazena.

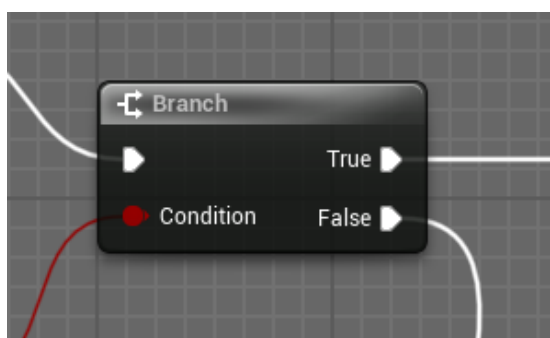


Obrázek 21 Set

(zdroj: vlastní)

Uzel Branch je ekvivalentem programovacího příkazu IF/ELSE a slouží k podmíněnému větvení toku logiky. Obsahuje jeden vstupní vykonávací pin a dva výstupní vykonávací piny označené True a False. Rozhodnutí o tom, který z výstupních pinů se

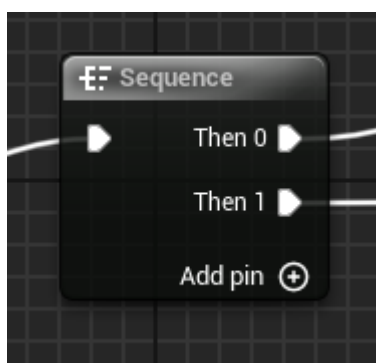
aktivuje, závisí na hodnotě boolovské proměnné připojené ke vstupnímu pinu Condition. Pokud je Condition pravdivá, aktivuje se větev True, v opačném případě se aktivuje větev False. Tento uzel je naprosto zásadní pro implementaci rozhodovací logiky v aplikaci, jelikož umožňuje provádět různé akce na základě splnění či nesplnění určitých podmínek, například zda je připojen VR headset, nebo zda uživatel stiskl určité tlačítko.



Obrázek 22 Branch

(zdroj: vlastní)

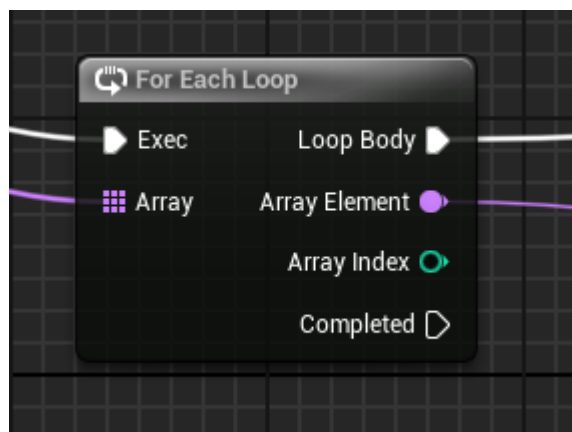
Sequence slouží k uspořádání vykonávání Blueprint logiky do sekvence. Disponuje jedním vstupním vykonávacím pinem a více očíslovanými výstupními vykonávacími piny např. Then 0, Then 1, Then 2 atd. Když je vstupní pin aktivován, blok provede vykonávací pin Then 0, po jeho dokončení Then 1, a takto pokračuje v číselném pořadí, dokud nejsou aktivovány všechny výstupní piny. Tento uzel je obzvláště užitečný, pokud je potřeba provést více na sobě nezávislých operací v přesně daném pořadí, aniž by se musela logicky propojovat každá operace s předchozí. Často se využívá pro přehlednější uspořádání složitějších Blueprint grafů.



Obrázek 23 Sekvence

(zdroj: vlastní)

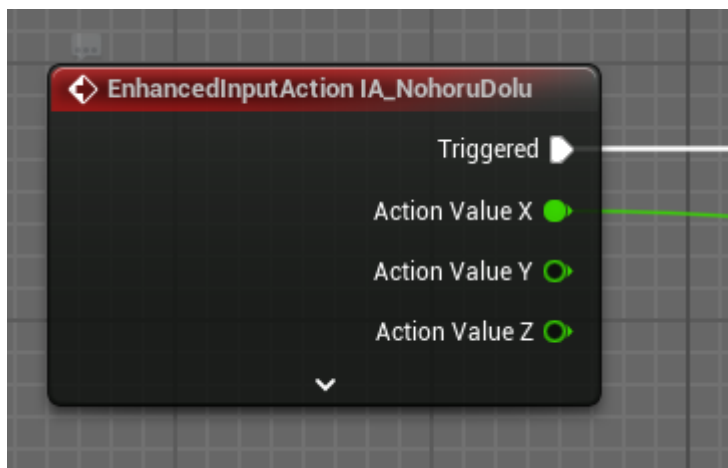
ForEachLoop je v Blueprintu určen pro iteraci přes prvky v poli. Přijímá jeden vstupní vykonávací pin a pole libovolného typu dat. Po aktivaci se uzel spustí a pro každý prvek v poli provede iteraci, přičemž aktivuje výstupní pin Loop Body. Během každé iterace je hodnota aktuálního prvku pole k dispozici na výstupním pinu Array Element a index aktuální iterace na pinu Array Index. Po dokončení zpracování všech prvků v poli se aktivuje výstupní pin Completed. Tento uzel je klíčový pro zpracování kolekcí objektů, například pro procházení seznamu prvků PZTS ve scéně a provádění akcí na každém z nich nebo pro dynamické načítání více objektů.



Obrázek 24 Loop

(zdroj: vlastní)

Uzel Enhanced Input Action je základním stavebním kamenem nového systému vstupů v UnrealEngine, který nahradil starší Input Action a Axis. Tento uzel reprezentuje konkrétní logickou akci definovanou v Input MappingContextu a je spuštěn, když uživatel provede odpovídající fyzický vstup např. stiskne klávesu, pohne myší. Uzel poskytuje různé výstupní piny, které se aktivují v různých fázích akce, například Started, Triggered, Completed nebo Canceled. Díky tomu lze snadno reagovat na různé stavy vstupu, jako je první stisk, držení tlačítka nebo jeho uvolnění. Enhanced Input Action umožňuje robustnější a flexibilnější správu vstupů, což je klíčové pro implementaci komplexních interakcí v editoru i VR prostředí.



Obrázek 25 Enhanced Input Action

(zdroj: vlastní)

10.2 Základní pohyb

Po definici Pawna Pohled z vrchu a nastavení vstupních kontextů bylo přistoupeno k implementaci základních pohybových mechanik v Blueprintu, které umožňují uživateli navigaci v editorovém prostředí.

Prvním krokem pro aktivaci systému Enhanced Input Action bylo získání reference na PlayerController. To bylo provedeno pomocí uzlu GetPlayerController, který poskytuje přístup k ovladači hráče. Tato reference je následně propojena s uzlem Enhanced Input LocalPlayerSubsystem, který je bránou k novému, vylepšenému systému vstupů v UnrealEnginu. Získáním přístupu k tomuto subsystému je možné aktivovat a deaktivovat Input MappingContexty, které definují, jaké vstupy jsou v daném okamžiku aktivní a jaké akce spouštějí.

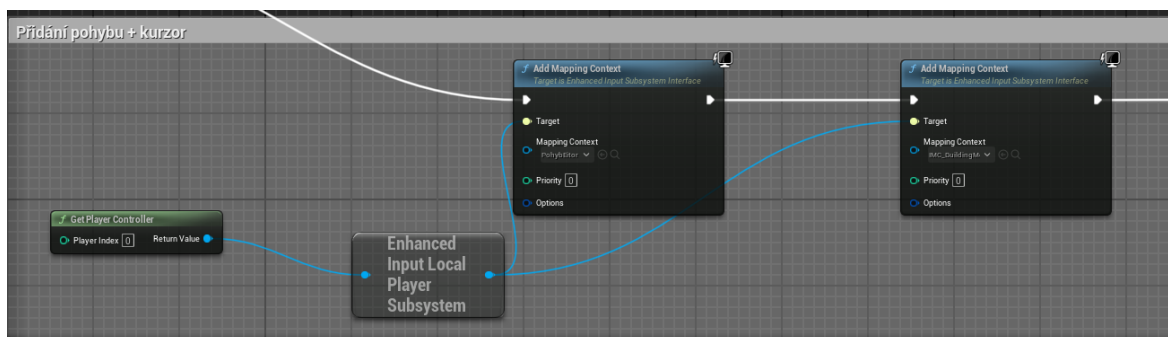
Následně byly aktivovány relevantní Input MappingContexty pro ovládání editoru. Byly přidány dva hlavní kontexty pomocí uzlů AddMappingContext: IMC_PohybEditor a IMC_Building. Tímto bylo zajištěno, že vstupy pro navigaci a pozdější stavění budou rozpoznávány a zpracovávány.

Pro pohyb vpřed, vzad a do stran ovládání pomocí kláves WSAD byl využit uzel EnhancedInputActionIA_PohybEditor. Tento uzel přijímá vstupní hodnoty pro pohyb po osách X a Y, které jsou získány z klávesnice. Pro převod těchto lokálních pohybů na globální směry ve světě byly využity uzly GetActorRotation, ze kterých jsou následně získány Get

Forward Vector pro pohyb vpřed a vzad a GetRightVector pro pohyb do stran. Tyto vektorové hodnoty jsou pak propojeny s uzly AddMovement Input, které Pawnu sdělují, kterým směrem se má pohybovat.

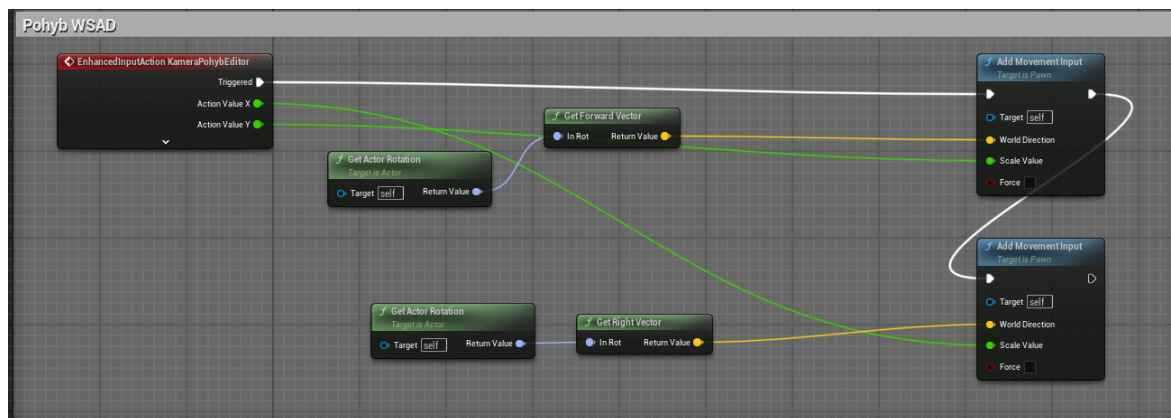
Dále byl implementován pohyb nahoru a dolů, který je řízen klávesami Control a mezerník. Klávesa mezerník slouží ke stoupání, zatímco klávesa Control klesání. Tato funkcionality je zajištěna uzlem EnhancedInputActionIA_NahoruDolu, který pracuje s osou Z. Propojením s uzlem AddMovement Input je pak Pawn pohybován vertikálně ve scéně.

Poslední implementovanou základní pohybovou funkcí bylo přiblížení a oddálení kamery (zoom in/out). Pro tuto funkci bylo využito kolečko myši, které generuje vstupní hodnoty. Uzel EnhancedInputActionIA_Zoom přijímá tyto hodnoty a na jejich základě se dynamicky mění zorné pole kamery. Rozsah zoomování je omezen nastaveným rozmezím od 15 do 90, což zajišťuje kontrolované a použitelné přiblížení a oddálení. Pro plynulost zoomu je využíván uzel FInterp To, který zajišťuje hladký přechod mezi aktuálním a cílovým zorným polem.

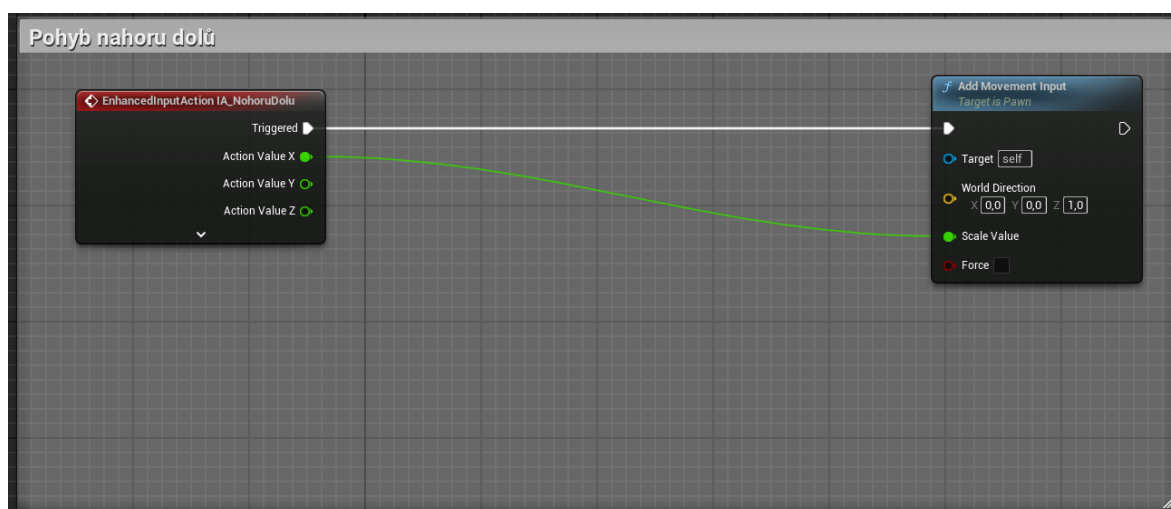


Obrázek 26 Pohyb1

(zdroj: vlastní)



Obrázek 27 Pohyb2

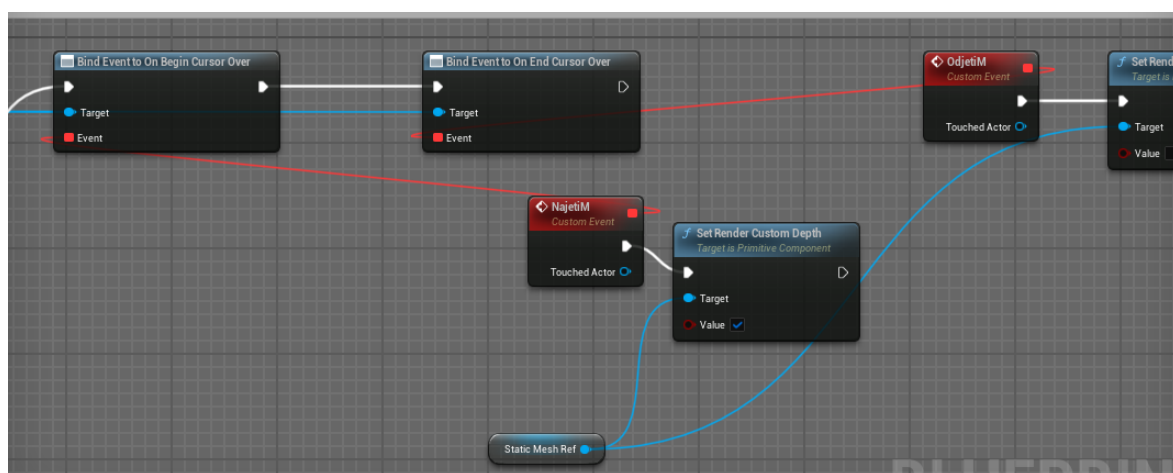
(zdroj: vlastní)

Obrázek 28 Pohyb3

(zdroj: vlastní)

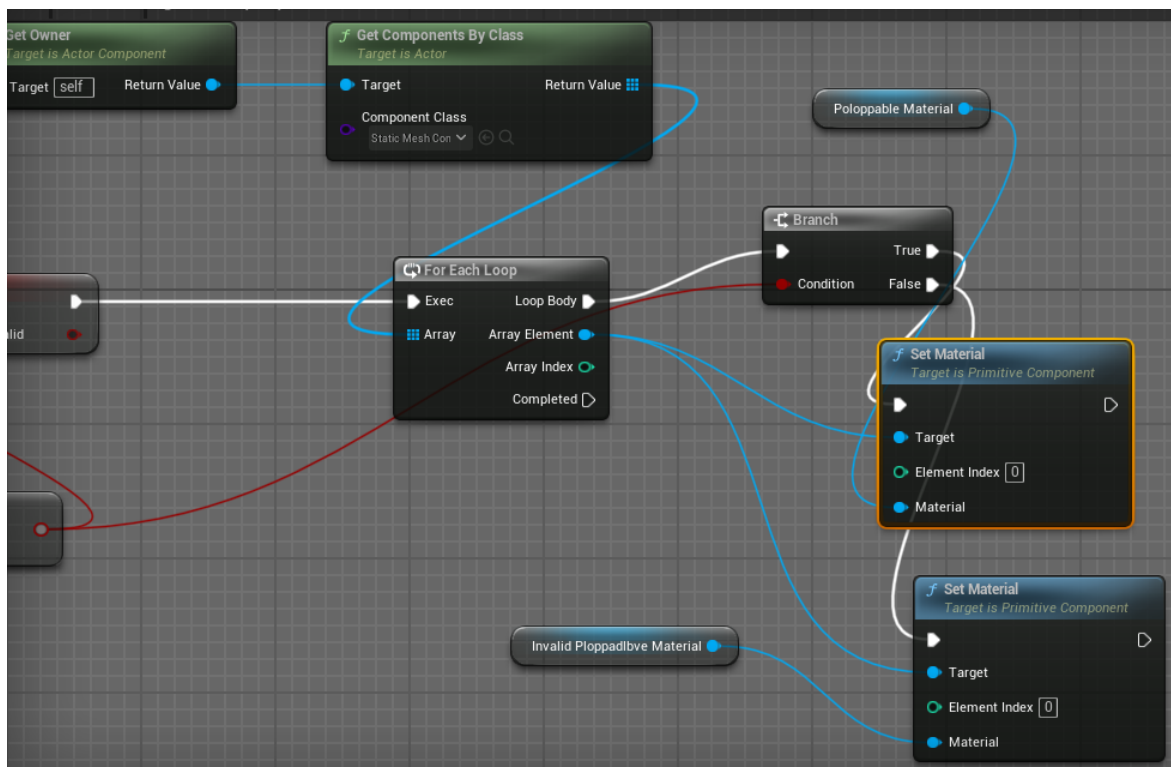
žádnými kolidujícími prvky, je zobrazen zelenou barvou, která signalizuje, že umístění je možné. Tato vizuální zpětná vazba je klíčová pro intuitivní a přesné umísťování stavebních prvků.

Pro tuto detekci kolizí jsou využívány dříve nastavené presetty a trasovací kanály. Tímto způsobem je zajištěno, že systém správně rozpozná platné a neplatné pozice pro umístění objektů, což výrazně zjednodušuje proces stavění a minimalizuje chyby uživatele. Pro sledování interakce kurzoru s objekty jsou použity události jako Bind Event to On ActorBeginOverlap a Bind Event to On Actor End Overlap.



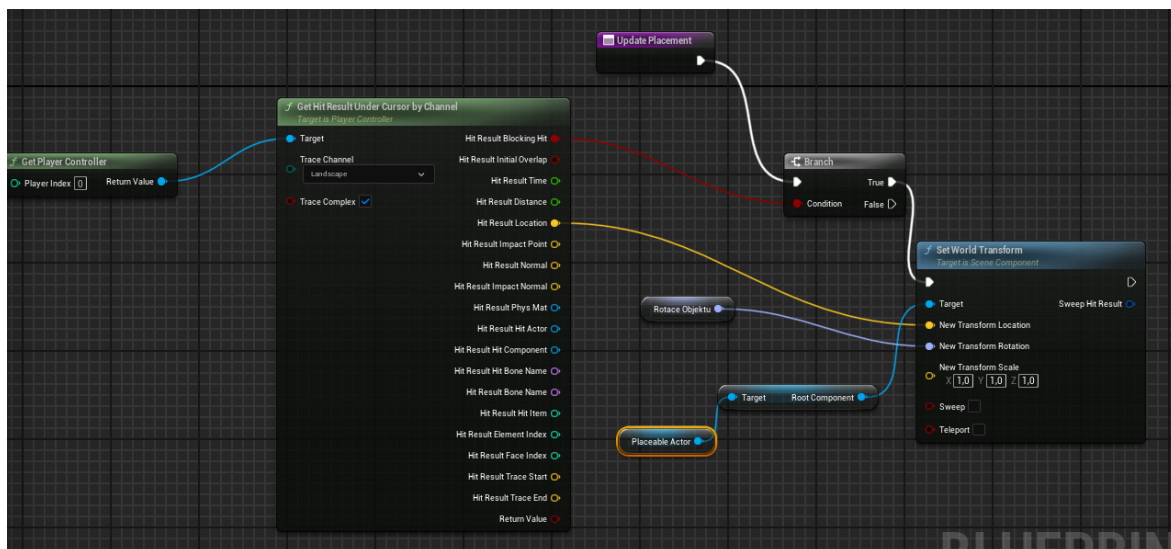
Obrázek 30 Detekce1

(zdroj: vlastní)



Obrázek 31 Detekce2

(zdroj: vlastní)



Obrázek 32 Detekce 3

(zdroj: vlastní)

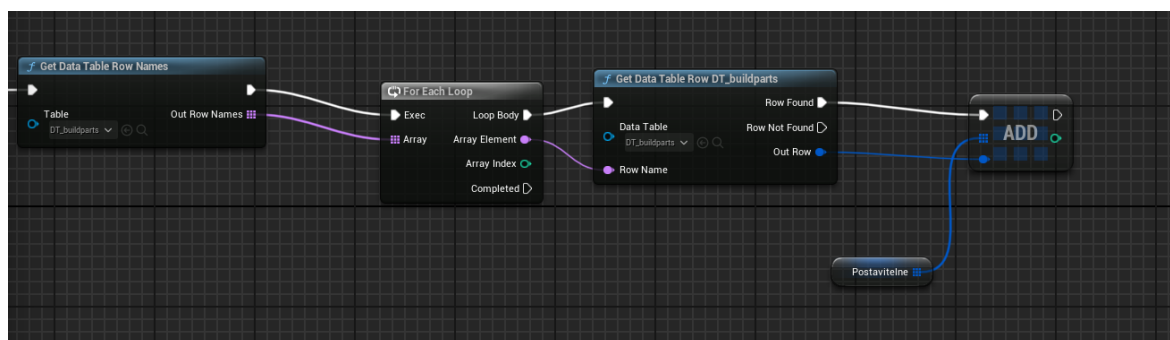
10.3.2 Výběr bloků

Výběr stavebních bloků představuje první krok v procesu umisťování prvků do scény a je úzce spojen s využitím datových tabulek, které byly popsány dříve. Cílem je poskytnout uživateli přehledný seznam dostupných stavebních prvků a umožnit mu vybrat ten, který chce následně umístit.

Tato funkcionality je realizována načtením dat z datové tabulky DT_buildparts. Pomocí uzlu Get Data Table RowNames jsou získány názvy všech řádků z této tabulky. Tyto názvy reprezentují jednotlivé stavební prvky, jako jsou různé typy prvků PZTS, nábytku, dveří nebo oken. Získané názvy jsou následně iterovány pomocí uzlu ForEachLoop.

Pro každý název řádku je volán uzel Get Data Table RowDT_buildparts, který načte kompletní data pro daný stavební prvek z datové tabulky. Tato data obsahují veškeré potřebné informace o objektu, včetně odkazu na jeho 3D model, výchozí barvy a popisu. Načtená data jsou poté přidána do pole PostavitelnáItem, což je dynamické pole, které obsahuje všechny dostupné stavební prvky pro aktuální session. Díky tomuto načítání dat je možné je následně využívat v jiných blocích, například pro pokládání nebo náhled bloku.

Tímto způsobem je vytvořena dynamická knihovna stavebních prvků, ze které si uživatel může vybírat. Výběr konkrétního bloku se obvykle provádí prostřednictvím uživatelského rozhraní (např. menu nebo seznamu), kde jsou zobrazeny dostupné prvky z pole PostavitelnáItem. Po výběru je daný stavební prvek připraven k následnému umístění do scény.



Obrázek 33 Výběr bloků

(zdroj: vlastní)

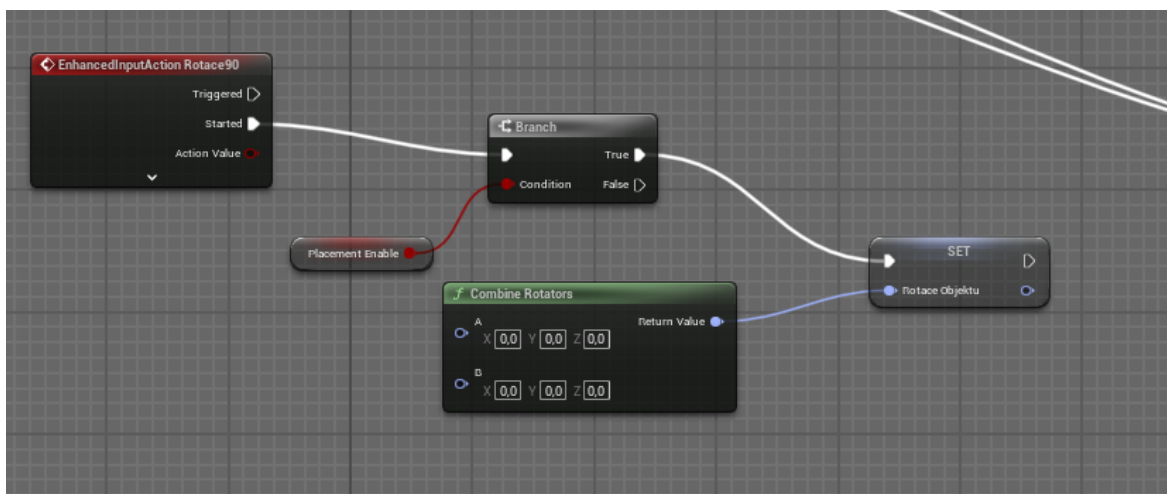
10.3.3 Rotace

V rámci systému stavění byla implementována funkce pro rotaci objektů, která uživateli umožňuje upravovat orientaci umístěvaných prvků ve scéně. Tato funkcionalita je dostupná v režimu náhledu objektu před jeho finálním položením.

Objekty lze rotovat několika způsoby:

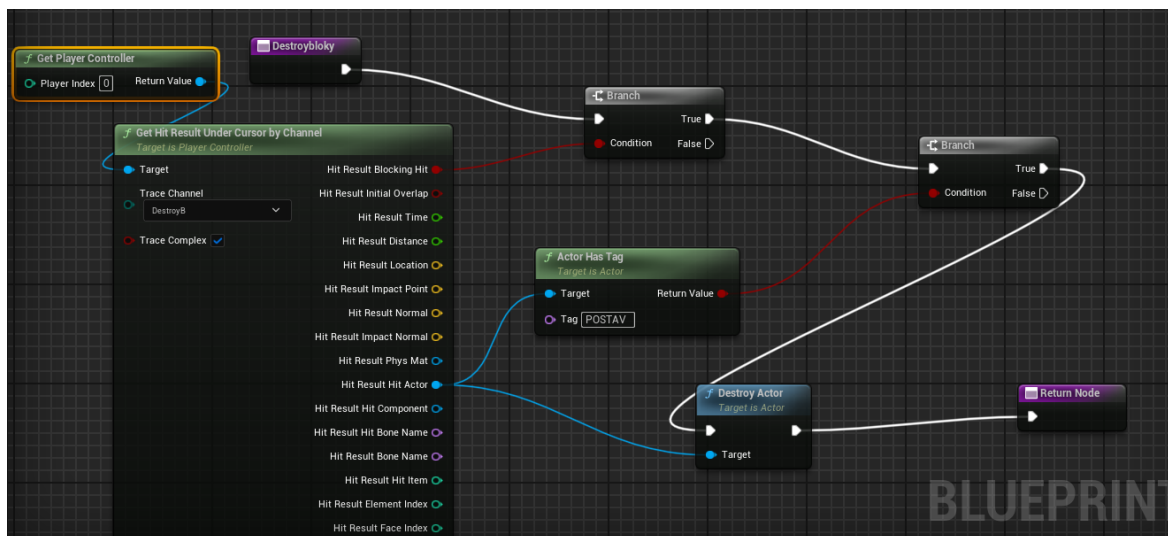
- Pravidelná rotace o předem danou vzdálenost: Tato metoda umožňuje plynulou rotaci objektu o malé, předem definované úhly.
- Pro rychlou a přesnou orientaci objektů je k dispozici funkce rotace o 90 stupňů. Tato je spouštěna prostřednictvím uzlu EnhancedInputAction Rotace90. Logika využívá uzly Branch a CombineRotators.
- Reset rotace: Rotaci objektu je možné kdykoli resetovat do jeho původní výchozí pozice.

Dále bylo zajištěno, že při každém vypnutí a následném zapnutí stavebního menu, o kterém bude pojednáno později, dojde automaticky k resetu rotace všech objektů na jejich výchozí pozice. To zajišťuje konzistentní chování a usnadňuje práci uživateli tím, že se objekty vždy objeví v očekávané orientaci. Rotace objektů je řízena funkcí CombineRotators a je možné ji ovládat v reálném čase. Příklad implementace rotace objektů je zachycen v Blueprintu Rotace Objektu.



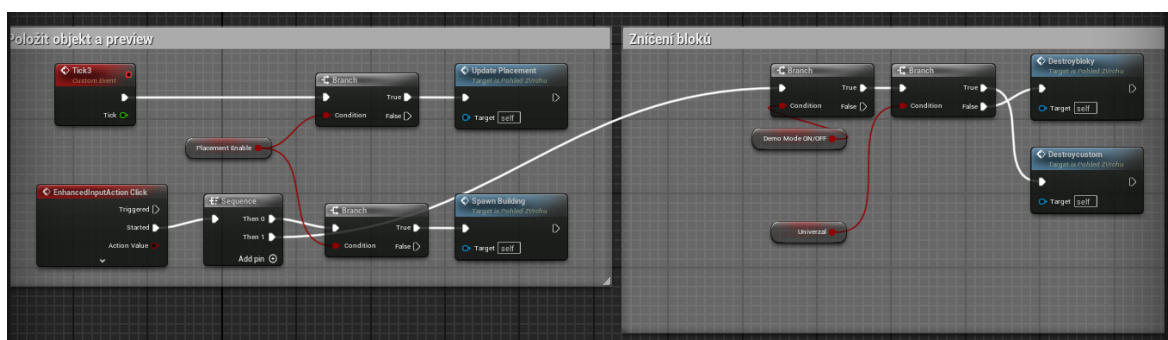
Obrázek 34 Rotace1

(zdroj: vlastní)



Obrázek 36 Mazni1

(zdroj: vlastní)



Obrázek 37Mazani2

(zdroj: vlastní)

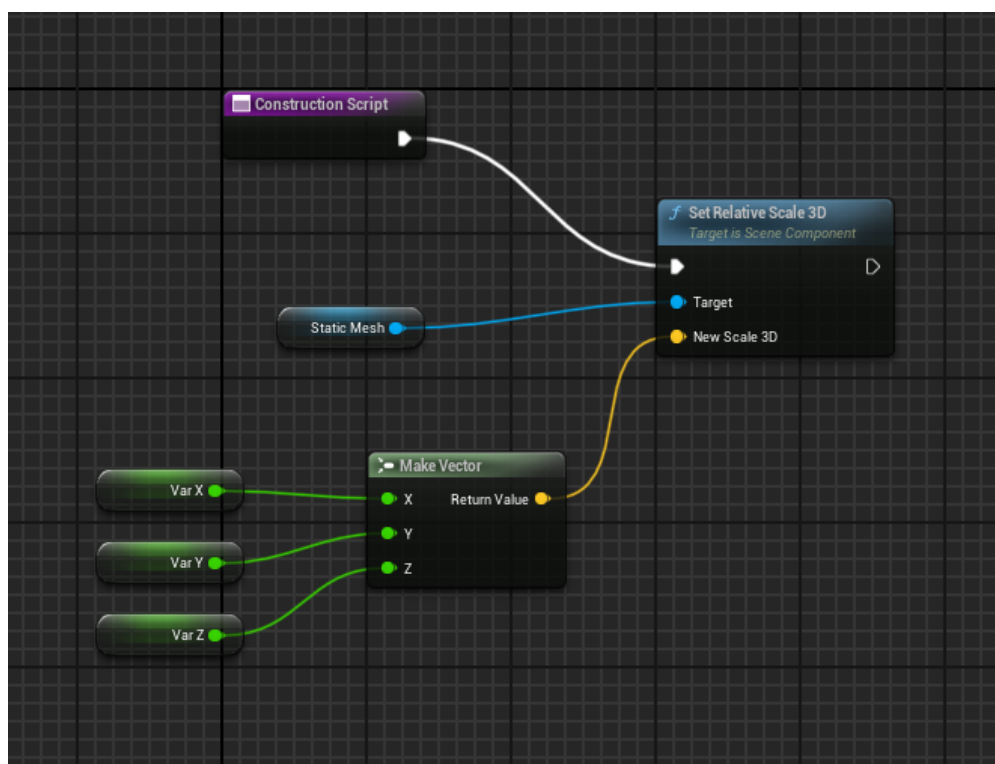
10.3.5 Nastavitelný blok

Mezi stavebními prvky systému stavění zaujímá zvláštní místo Nastavitelný blok. Na rozdíl od ostatních bloků, které jsou dceřinými herci child actors odvozenými z předem definovaných parent actorů, Nastavitelný blok je unikátní entitou. Jedná se o samostatný objekt, u něhož je možné dynamicky nastavovat jeho rozměry pomocí 3D vektoru nebo 3D škálování, využívajícího kartézské souřadnice X, Y a Z.

Hlavní význam Nastavitelného bloku spočívá v jeho flexibilitě při doplňování chybějících nebo nesprávně vygenerovaných stěn a stropů. Tyto nedostatky mohou

vzniknout při importu z meshe vytvořeného v Blenderu, Inkscapu, nebo z jiného podkladového materiálu, jako jsou soubory DWG. V případech, kdy automatické generování scény z 2D půdorysů není zcela přesné nebo kompletní, Nastavitelný blok umožňuje uživateli manuálně doplnit a upravit geometrii tak, aby vizualizace odpovídala zamýšlenému návrhu.

I přes svou unikátní vlastnost nastavitelnosti rozměrů, sdílí Nastavitelný blok základní možnosti interakce s ostatními stavebními prvky. Lze jej položit do scény, otáčet pro správnou orientaci a také zničit v případě potřeby úprav. Klíčovou výhodou je však právě možnost plynule upravovat jeho velikost, což z něj činí nepostradatelný nástroj pro finální doladění architektonických detailů a zajištění bezchybné geometrie v celé vizualizaci.



Obrázek 38 Nastavitelný1

(zdroj: vlastní)

Z tohoto důvodu byl pro efektivní import budov v reálném čase použit specializovaný plugin. Tento plugin umožňuje dynamické načítání FBX souborů přímo během běhu aplikace, což poskytuje značnou flexibilitu. Umožňuje uživatelům importovat své vlastní architektonické modely vytvořené například v Blenderu nebo jiném 3D modelovacím softwaru, aniž by museli předem kompilovat aplikaci s každou novou budovou. Tím je výrazně usnadněna práce s uživatelským obsahem a přizpůsobení vizualizace specifickým požadavkům klienta.

10.4.1 Real Time Import/Export Plugin

Pro usnadnění dynamického načítání a ukládání dat v reálném čase byla v rámci projektu využita funkcionalita pluginu Real Time Import/Export Plugin. Tento plugin, jehož podrobnosti byly získány ze stránky výrobce, nabízí širokou škálu možností pro práci s různými formáty souborů přímo během běhu aplikace, což eliminuje nutnost použití editoru pro každou změnu.

Plugin je navržen tak, aby umožňoval import zvukových souborů WAV, OGG, obrazových souborů PNG, JPG, BMP, TGA, TIFF a meshů OBJ, FBX, STL. Kromě importních funkcí nabízí také exportní možnosti pro TextureRenderTarget2D do bytů (BMP, PNG, JPEG) a procedurálních meshů do souborů OBJ. Významnou výhodou tohoto pluginu je, že nespolehá na knihovny třetích stran, což zjednodušuje instalaci a předchází potenciálním licenčním problémům.

10.4.2 Implementace

Pro import budovy do aplikace je využíván formát FBX, který je vhodný pro přenos 3D modelů s texturami a animacemi. Důvodem pro samostatný import budov je především možnost jejich načítání v reálném čase (runtime). Implementace takového systému bez využití externích nástrojů by si vyžádala pokročilejší znalosti programovacího jazyka, který je v Unreal Engine primárně využíván (C++), a nelze ji plně realizovat pouze pomocí vizuálního skriptování v Blueprintu.

Z tohoto důvodu byl pro efektivní import budov v reálném čase použit specializovaný plugin. Tento plugin umožňuje dynamické načítání FBX souborů přímo během běhu aplikace, což poskytuje značnou flexibilitu. Umožňuje uživatelům importovat své vlastní architektonické modely vytvořené například v Blenderu nebo jiném 3D modelovacím softwaru, aniž by museli předem kompilovat aplikaci s každou novou budovou. Tím je

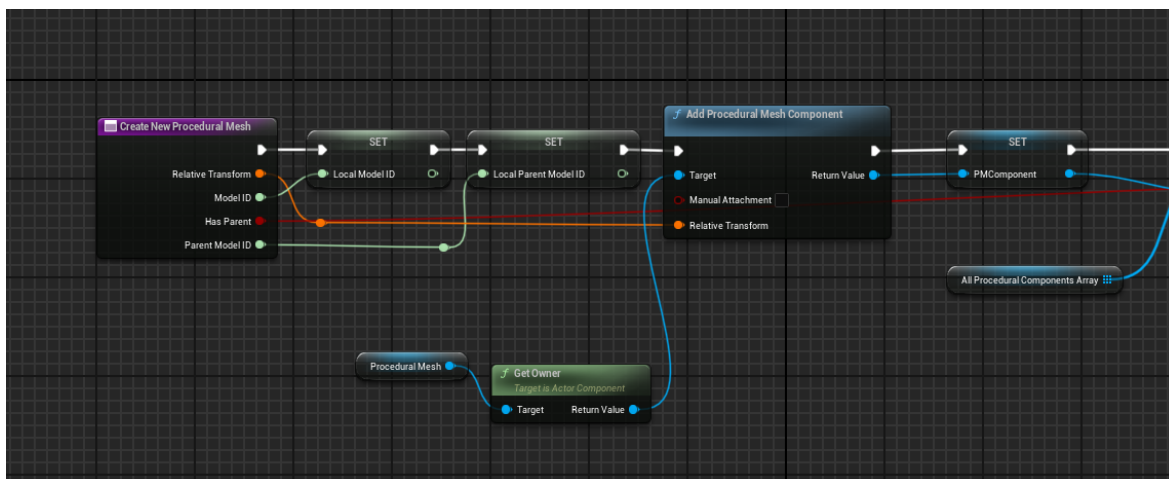
výrazně usnadněna práce s uživatelským obsahem a přizpůsobení vizualizace specifickým požadavkům klienta.

Implementace tohoto pluginu spočívá v načítání statického meshe ve formátu FBX. Tento mesh budovy byl předem vytvořen za pomoci softwaru jako je Inscape a Blender nebo byl získán z jiného externího softwaru. Načtený mesh je automaticky zobrazen v měřítku 1:1 a je na něj aplikována výchozí bílá barva. Tímto je budova umístěna do našeho virtuálního prostoru. Jelikož jsou na tento statický mesh aplikovány předem definované kolizní presety, funguje jako podkladový kámen pro umísťování dalších objektů. Na něj lze pokládat další stavební prvky bez problémů s kolizemi, což efektivně slouží jako základ pro další projektování.



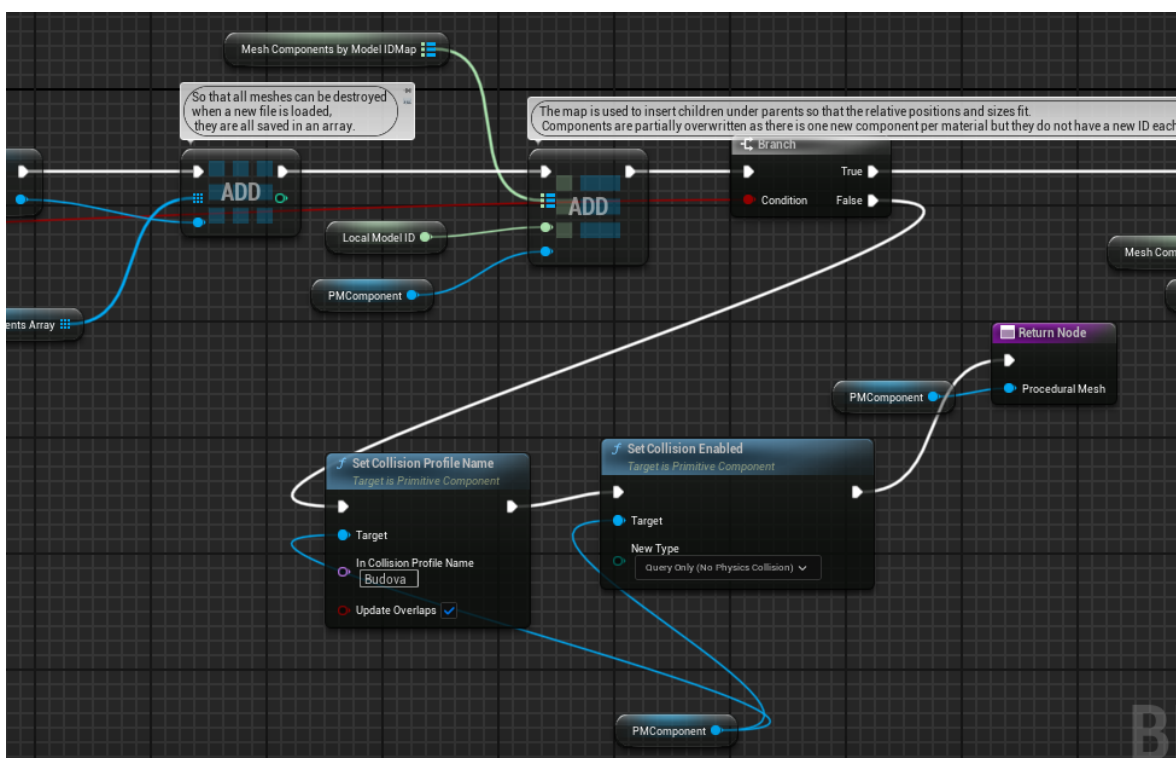
Obrázek 41 Import1

(zdroj: vlastní)



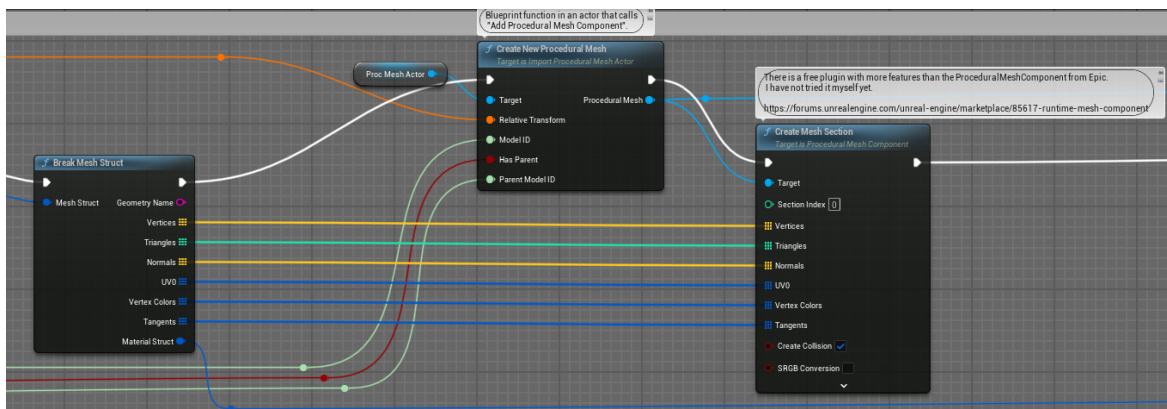
Obrázek 42 Import2

(zdroj: vlastní)



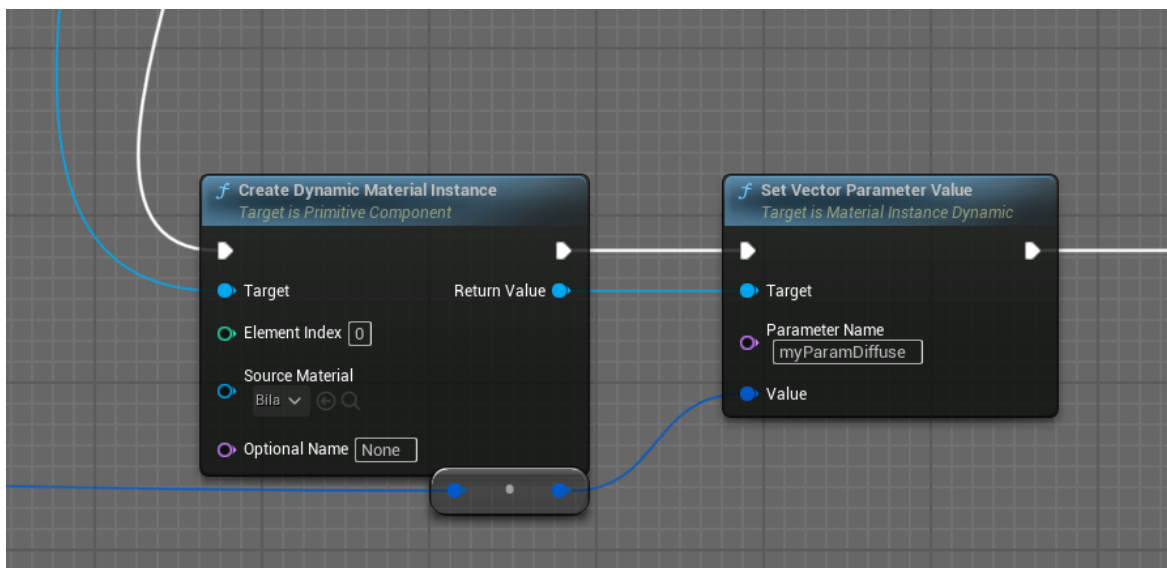
Obrázek 43 Import3

(zdroj: vlastní)



Obrázek 44I Import4

(zdroj: vlastní)



Obrázek 45 Import5

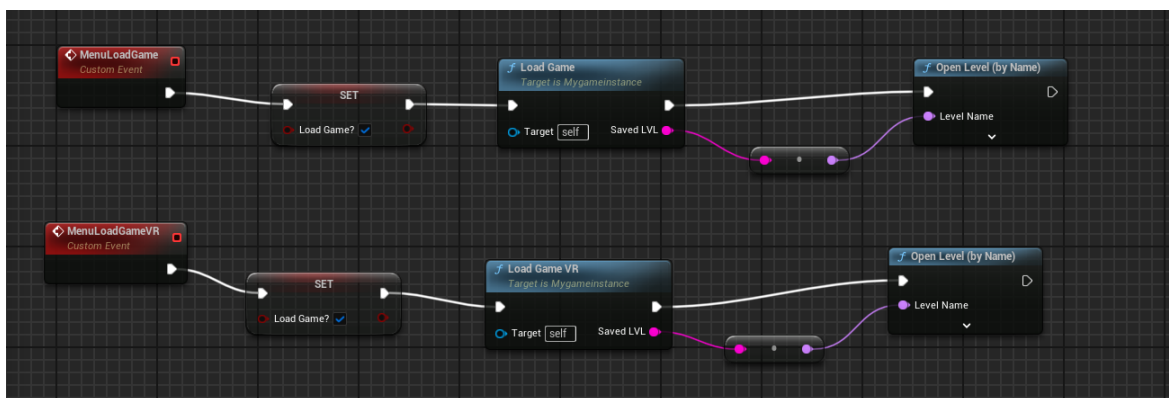
(zdroj: vlastní)

10.5 Ukládání

Pro zajištění persistentního ukládání a načítání uživatelských projektů byla v aplikaci implementována robustní mechanika, která využívá specifické Blueprint uzly a Game Instance pro správu dat. Cílem bylo umožnit uživatelům ukládat rozpracované vizualizace a kdykoli se k nim vrátit, ať už v desktopovém editoru, nebo v rámci VR prohlížeče.

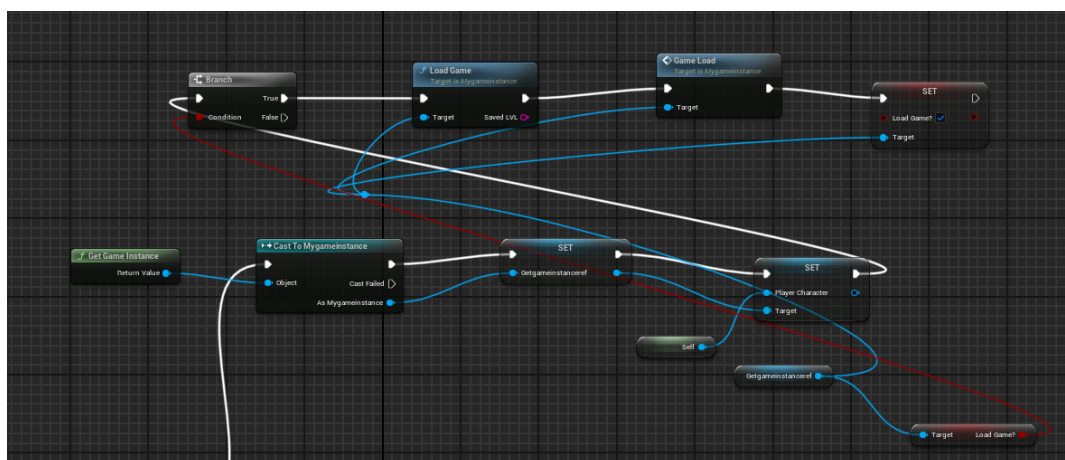
Systém ukládání je řešen dvojím způsobem: vždy se ukládá jedna verze pro desktop a druhá verze pro VR. Desktopová verze je určena pro další práci v editoru, zatímco VR verze je uložena samostatně pro účely prohlížení. Pro strukturování ukládaných dat byly vytvořeny dvě hlavní struktury: Barák Safe a Bolky. Tyto struktury jsou ukládány do proměnné Superulozeni, která slouží jako kontejner pro veškerá uložená data projektu. V rámci Game Instance je pak vytvořena instance, ze které jsou volány hlavní funkce pro správu ukládání a načítání, jako jsou Load Game, Load Game VR a Save Game.

Tyto funkce umožňují efektivní správu stavu aplikace a jejích projektů napříč různými platformami a režimy zobrazení. Kromě těchto základních mechanismů je dále řešeno ukládání vrstev, o čemž bude podrobněji pojednáno v samostatné kapitole.



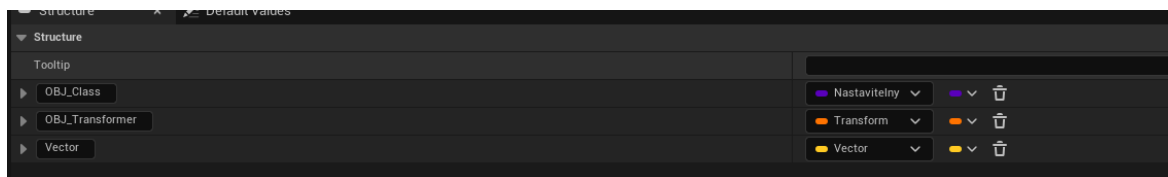
Obrázek 46 Ulozen1

(zdroj: vlastní)

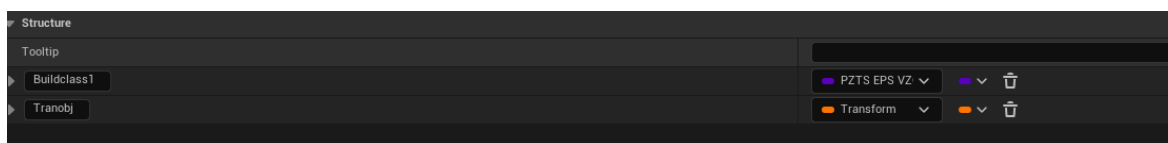


Obrázek 47 Ulozena2

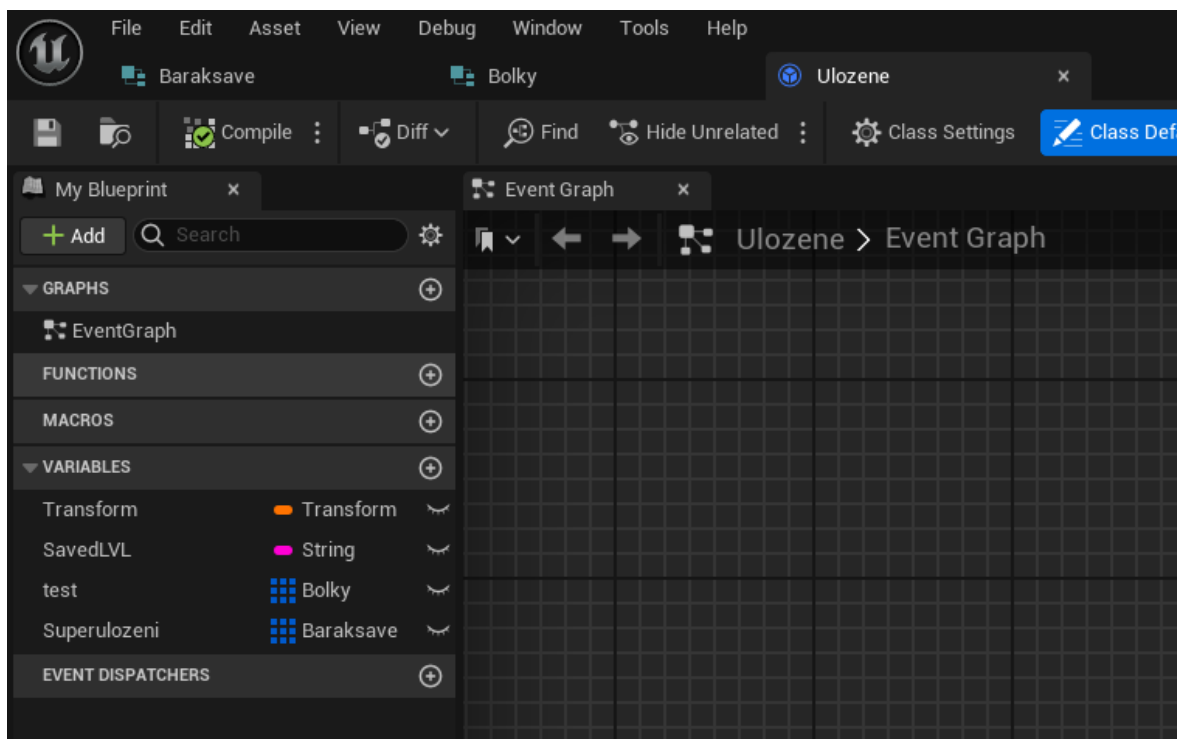
(zdroj: vlastní)



Obrázek 48 Ulozena3

(zdroj: vlastní)

Obrázek 49 Ulozena4

(zdroj: vlastní)

Obrázek 50 Ulozeno5

(zdroj: vlastní)

10.5.1 Ukládání a načítání do slotů

Pro ukládání a načítání projektů je v aplikaci implementován systém slotů, který umožňuje uživateli spravovat více verzí svých vizualizací. Konkrétně bylo vytvořeno šest ukládacích slotů určených pro PC (desktop) verzi aplikace.

Při ukládání projektu je vytvořena jedna kopie dat pro každý zvolený slot. Tato data jsou pak uložena v souborech, které reprezentují jednotlivé uložené stavy projektu. Tyto uložené soubory lze následně přenášet na VR headset, jako je například Meta Quest 3, za pomoci externí aplikace SideQuest. Aplikace SideQuest umožňuje snadný přenos souborů mezi počítačem a VR zařízením, což zajišťuje flexibilitu v práci s projekty napříč platformami. Tímto způsobem je zajištěna možnost pokračovat v práci na projektu v desktopovém editoru a poté si jej prohlédnout ve virtuální realitě.

10.5.2 Ukládání a načítání vrstev

Ukládání a načítání vrstev představuje rozšířenou funkcionalitu systému pro správu projektů, která umožňuje vytvářet a spravovat více návrhů na jednom objektu. Tato schopnost je zvláště užitečná pro prezentaci různých variant zabezpečení nebo interiérového designu v rámci jediné budovy.

Princip spočívá v tom, že k jednomu hlavnímu ukládacímu souboru save file lze přidružit až tři další pod-save soubory. Mezi těmito vrstvami se pak lze dynamicky přepínat pomocí uživatelského rozhraní widgetu. To znamená, že uživatel může mít jeden základní model budovy a k němu vytvořit například tři různé návrhy zabezpečení, z nichž každý je uložen jako samostatná vrstva.

Při uložení projektu s aktivními vrstvami dojde k uložení kopií pro každou vrstvu. Například, pokud je aktivní vrstva 1, uloží se jako Slot1.1, Slot1.2 a Slot1.3 pro desktopovou verzi a odpovídajícím způsobem Slot1.1VR, Slot1.2VR a Slot1.3VR pro VR verzi. Tímto způsobem jsou všechny verze projektu přeneseny do VR složky. Následně si uživatel ve VR prostředí může procházet všechny tři modely a porovnávat různé návrhy, což výrazně zvyšuje flexibilitu a prezentační možnosti aplikace.

10.6 Widgety

Widgety v Unreal Enginu jsou základními stavebními kameny pro tvorbu uživatelských rozhraní (UI). Jedná se o vizuální prvky, jako jsou tlačítka, textová pole, posuvníky, obrázky, seznamy a další interaktivní komponenty, které umožňují uživateli komunikovat s aplikací. Celkové uživatelské rozhraní (UI) je pak souhrnem všech těchto widgetů a jejich uspořádání, které zajišťuje vizuální prezentaci a interaktivitu.

V Unreal Enginu se widgety vytvářejí pomocí systému UMG (Unreal Motion Graphics) UI Designer. Tento vizuální nástroj umožňuje návrhářům a vývojářům snadno skládat UI prvky, definovat jejich vzhled a programovat jejich chování pomocí Blueprintů. Widgety jsou hierarchicky uspořádány, což umožňuje vytvářet složitá rozhraní z menších, znovupoužitelných komponent.

Implementace uživatelského rozhraní v této aplikaci byla shledána jako velice komplikovaná, zejména vzhledem k nutnosti správy velkého množství interakcí. Složitost pramení z potřeby reagovat na rozmanité uživatelské vstupy, dynamicky měnit stav UI prvků a zajistit plynulou komunikaci mezi UI a herní logikou, což vyžaduje pečlivé plánování a implementaci.

UI v aplikaci slouží k několika klíčovým účelům:

- Navigace: Poskytuje uživatelům možnost přepínat mezi různými částmi aplikace, jako je hlavní menu, editor nebo VR prohlížeč.
- Interakce: Umožňuje uživatelům provádět akce, jako je výběr stavebních prvků, změna materiálů, ukládání a načítání projektů.
- Zpětná vazba: Poskytuje uživatelům vizuální informace o stavu aplikace, například aktuálně vybraném nástroji nebo upozornění na kolize.

Správně navržené a implementované UI je zásadní pro uživatelskou přívětivost a intuitivní ovládání, což je klíčové pro efektivní práci v editoru i pro pohlcující zážitek ve virtuální realitě.

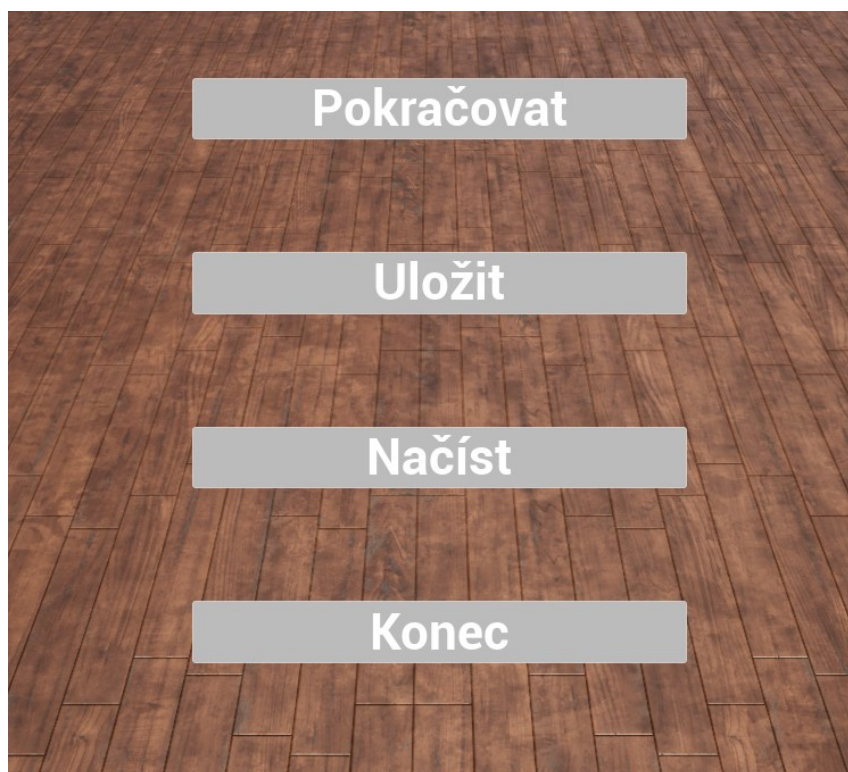
10.6.1 Tvorba menu

Tvorba pauzovacího a hlavního menu představovala klíčovou součást vývoje uživatelského rozhraní aplikace. Přestože se na první pohled mohou zdát jako relativně jednoduché prvky,

jejich implementace vyžadovala značné množství Blueprint logiky a komplexních propojení, aby byla zajištěna jejich plná funkčnost a intuitivní ovládání.

Pauzovací menu, dostupné během aktivní vizualizace, a hlavní menu, které slouží jako vstupní brána do aplikace, musely být navrženy tak, aby poskytovaly plynulý přechod mezi stavy. Za zdánlivě jednoduchými tlačítky a možnostmi se skrývá složitá síť událostí, podmínek a volání funkcí. Například, stisk tlačítka Načíst projekt v hlavním menu nespouští pouze načtení souboru, ale iniciuje celou sekvenci akcí: kontrolu existence souboru, deserializaci dat, dynamické generování scény a přenesení uživatele do příslušného levelu. Stejně tak aktivace pauzovacího menu zahrnuje pozastavení herní logiky, zobrazení widgetu a správu vstupů, aby se zabránilo nechtěným interakcím se scénou na pozadí.

Díky vizuální povaze Blueprintu je možné tuto komplexnost spravovat, avšak vyžaduje to pečlivé plánování a strukturování jednotlivých bloků a jejich propojení. Každé tlačítko, posuvník nebo textové pole má svou vlastní logiku, která se proplétá s logikou ostatních UI prvků a s celkovou herní logikou aplikace. Výsledkem je funkční a uživatelsky přívětivé rozhraní, které efektivně plní svůj účel navzdory vnitřní složitosti své implementace.



Obrázek 51 Tvorba menu1

(zdroj: vlastní)



Obrázek 52 Tvorba menu2

(zdroj: vlastní)

10.6.2 Tvorba UI

Uživatelské rozhraní (UI) představuje nedílnou součást aplikace, která je uživateli neustále na očích a slouží jako primární prostředek pro interakci s virtuálním prostředím. Jeho důležitost spočívá v tom, že zprostředkovává veškerou komunikaci mezi uživatelem a aplikací.

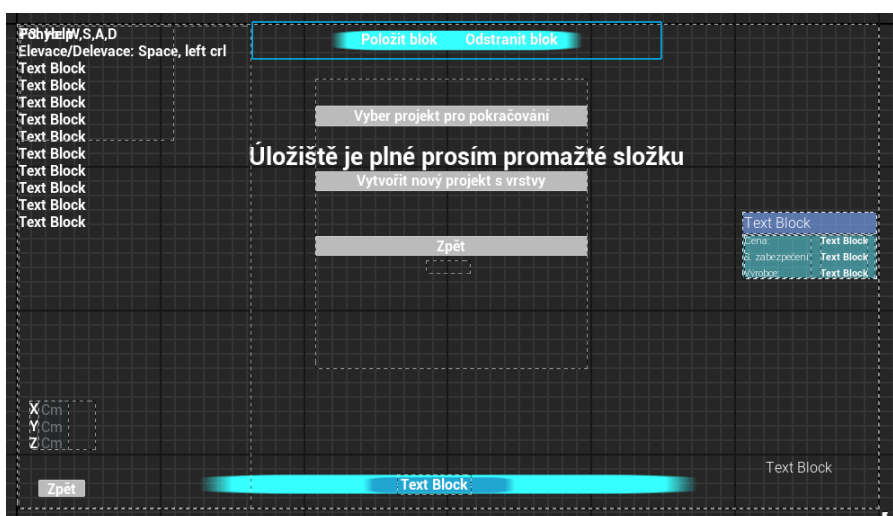
V rámci UI se klíčovou roli hrají datové bindingy. Jedná se o mechanismus, který umožňuje dynamicky propojovat vizuální prvky rozhraní s proměnnými a daty v podkladové logice aplikace (Blueprintu nebo C++ kódu). Díky bindingům se obsah UI aktualizuje v reálném čase, kdykoli dojde ke změně propojené proměnné.

Tento systém má dvě hlavní funkce:

- Příjem dat: UI dokáže čerpat data z různých zdrojů v aplikaci. Například, může zobrazovat aktuální stav systému zabezpečení, název místnosti, ve které se uživatel nachází, nebo jiné relevantní informace o prohlíženém objektu.

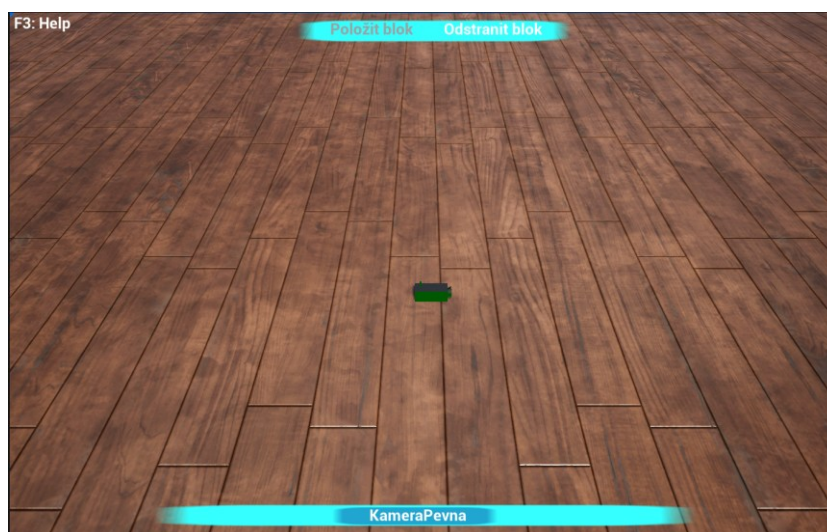
- Zadávání dat: Skrz UI může uživatel zadávat data a spouštět akce. Příkladem může být výběr stavebních prvků z nabídky, změna materiálů na stěnách, ukládání nebo načítání projektů, nebo přepínání mezi různými vrstvami návrhů.

Správně navržené a implementované UI, které efektivně využívá datové bindingy, je zásadní pro uživatelskou přívětivost a intuitivní ovládání. Tím je zajištěn plynulý a informativní zážitek uživatele, ať už pracuje v desktopovém editoru, nebo prochází vizualizací ve virtuální realitě.



Obrázek 53 Tvorba UI1

(zdroj: vlastní)



Obrázek 54 Tvorba UI2

(zdroj: vlastní)

11 DOKUMENTACE APLIKACE

Vzhledem k rozsahu všech Blueprintů, datových struktur a celkové komplexnosti činností provedených na aplikaci není úplné grafické nebo textové zobrazení veškeré implementace diplomové práce zcela praktické.

11.1 Klíčové prvky

Mezi klíčové prvky, které jsou v dokumentaci detailně popsány a které jsou stěžejní pro funkcionalitu celého programu, patří importace, ukládání a načítání, stavění a prohlížení. Každý z těchto modulů hraje nezastupitelnou roli a jejich správná implementace a vzájemná souhra jsou pro bezproblémový chod aplikace zásadní.

- **Importace:** Tento modul se zabývá procesem načítání externích dat, jako jsou 2D půdorysy nebo 3D modely budov ve formátu FBX, do prostředí aplikace. Je popsán mechanismus, jakým jsou tato data zpracovávána a převáděna do formátu použitelného v Unreal Enginu, včetně využití pluginu pro runtime import. Bez možnosti importovat podkladové architektonické plány a modely by nebylo možné vytvářet základní strukturu vizualizací.
- **Ukládání a načítání:** Tato sekce dokumentace se věnuje systému persistentního uchovávání uživatelských projektů. Je detailně popsán proces ukládání a načítání dat do slotů, včetně správy vrstev, což umožňuje uživatelům vytvářet více variant návrhů na jednom objektu. Správné fungování ukládání a načítání je kritické pro zachování uživatelské práce a možnost vracet se k projektům.
- **Stavění:** V dokumentaci je podrobně popsán komplexní systém stavění, který zahrnuje výběr stavebních bloků z datových tabulek, jejich pokládání s vizuální detekcí kolizí, rotaci a mazání. Dále je zde popsán unikátní nastavitelný blok, který slouží k doplňování a úpravám geometrie. Tento modul je srdcem editoru a umožňuje uživatelům vytvářet a modifikovat 3D vizualizace bytových prostor.
- **Prohlížení:** Tato část dokumentace se zabývá implementací režimu prohlížení, a to jak v desktopovém prostředí, tak především ve virtuální realitě. Jsou zde popsány mechaniky navigace, interakce s objekty a uživatelské rozhraní, které je uživateli k dispozici během průchodu vizualizací. Funkcionalita prohlížení je klíčová pro prezentaci vytvořených návrhů klientům a pro pohlcující zážitek ve VR

11.2 Základní ovládání

Veškerá dokumentace týkající se ovládání a možnostem programu je přiložena v příloze. Uživatelé zde naleznou podrobné informace k ovládání pohybu, včetně návodů pro navigaci virtuálním prostorem a pohyb kamery. Dále je popsán systém stavění, kde jsou detailně rozebrány funkcionality výběru bloků, jejich pokládání, rotace a mazání. Sekce ukládání a načítání poskytuje informace o ukládání a načítání projektů do slotů a správě vrstev. Importace je vysvětlena v souvislosti s nahráváním 2D půdorysů a 3D modelů. Hlavní menu je popsáno z hlediska svých funkcí a navigace. Návod pro spuštění VR a práci s aplikací ve virtuální realitě je také součástí dokumentace. Uživatelé se rovněž dozvědí, jak pracovat s půdorysem, tedy jak vytvořit a nahrát půdorys do programu. V neposlední řadě jsou uvedeny instrukce pro přenos uložených souborů, včetně práce se SideQuestem pro přenos dat do VR headsetu.

12 MOŽNÝ BUDOUCÍ VÝVOJ

Možný budoucí vývoj aplikace by se mohl zaměřovat na několik klíčových oblastí, které by vedly k výraznému zlepšení uživatelského zážitku, flexibility a celkové kvality vizualizací. Jedná se o potenciální směry pro další zdokonalování, které by mohly být implementovány.

Jedním z hlavních cílů by se mohlo stát výrazné zlepšení celkové vizualizace. To by mohlo zahrnovat optimalizaci renderování pro ještě realističtější zobrazení, vylepšení osvětlení a stínů a implementaci pokročilejších vizuálních efektů. Současně by se nabízela možnost rozšíření a zkvalitnění stávajících knihoven assetů. Klíčovým prvkem by se také mohla stát implementace robustní možnosti doplňování knihoven, což by uživatelům umožnilo snadno integrovat vlastní 3D modely a materiály do aplikace, čímž by se značně zvýšila její flexibilita a přizpůsobitelnost.

Další významnou oblastí pro budoucí vývoj by mohlo být zdokonalení grafického rozhraní (UI). To by mohlo zahrnovat revizi stávajících prvků, optimalizaci pracovních postupů a přidání nových nástrojů pro intuitivnější a efektivnější práci. Zlepšení by se týkalo také hlavní nabídky a systému správy ukládaných souborů save managementu. Implementace pokročilejších funkcí pro organizaci, zálohování a sdílení projektů by výrazně usnadnila práci s větším množstvím vizualizací a přispěla k celkové uživatelské přívětivosti aplikace

ZÁVĚR

Diplomová práce úspěšně představila a popsala vývoj rozhraní pro tvorbu a prezentaci 3D vizualizací bytů a menších rodinných domů ve virtuální realitě. Hlavním cílem projektu bylo zpřístupnit 3D modelování a vizualizaci i širšímu okruhu uživatelů, kteří dosud neměli s touto disciplínou zkušenosti, a to s důrazem na jednoduchost, intuitivnost a finanční dostupnost. Vytvořený desktopový program se specializuje na snadnou tvorbu a prezentaci návrhů prvků zastínění a technik stínění (PZTS), což firmám umožňuje srozumitelně a poutavě demonstrovat finální podobu zakázky svým klientům. Díky zvolenému přístupu, který připomíná intuitivnost hry The Sims, je i pro naprosté začátečníky možné bez složitého učení vytvářet vlastní vizualizace. Implementace multifunkční aplikace, která dynamicky rozpoznává prostředí PC nebo VR, přinesla značné výhody v efektivitě vývoje a správě dat. Aplikace splňuje přísné požadavky na výkon a plynulost, což je klíčové pro komfortní VR zážitek. Možný budoucí vývoj by se mohl zaměřit na další zlepšení vizualizace, rozšíření knihoven assetů a zdokonalení uživatelského rozhraní, což by dále zvýšilo flexibilitu aplikace. Celkově inovativní rozhraní a snadné ovládání otevírají dveře do světa 3D modelování i těm, kteří se dosud obávali složitosti tradičních softwarů

SEZNAM POUŽITÉ LITERATURY

- [1] SUN, Yige, Zixuan ZHANG a Nan YAN. Visualization Challenges of Virtual Reality 3D Images in New Media Environments. Applied Mathematics and Nonlinear Sciences [online]. 2024, 9(1), s. 1-20 [cit. 2025-05-30].
- [2] RACE, Steve. BIM Demystified. RIBA Publishing, 2019. ISBN 9780429347627.
- [3] OGTEROP, Wilhelm a VENTER, Henk. Unreal Engine 5 Character Creation, Animation, and Cinematics. Packt Publishing, 2022. ISBN 9781801812443.
- [4] ROMERO, Marcos a SEWELL Brenden. Blueprints Visual Scripting for Unreal Engine 5. Packt Publishing, 2022. ISBN 9781801811583.
- [5] AUTODESK. Revit – přehled, 2025. [online]. Autodesk, ©2025 [cit. 2025-05-30]. Dostupné z: <https://www.autodesk.com/cz/products/revit/overview?term=1-YEAR&tab=subscription>
- [6] SketchUp - 3D modelovací program, 2025. [online]. SketchUp, [cit. 2025-05-30]. Dostupné z: <https://sketchup.cz>
- [7] SIMLAB SOFT. SimLab Showroom, 2025. [online]. SimLab Soft, ©2024 [cit. 2025-05-30]. Dostupné z: <https://www.simlab-soft.com/technologies/simlab-showroom.aspx>
- [8] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management I. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBUM, 2011. ISBN 978-80-87500-05-7.
- [9] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management II. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBUM, 2012. ISBN 978-80-87500-19-4.
- [10] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management III. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VeRBUM, 2013. ISBN 978-80-87500-35-4.
- [11] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management IV. Zlín: Radim Bačuvčík – VeRBUM, 2014. ISBN 978-80-87500-57-6.
- [12] VARJO. Varjo Aero - Professional VR headset for stunning virtual experiences, 2025. [online]. Varjo Technologies Oy, [cit. 2025-05-30]. Dostupné z: <https://varjo.com/products/aero/>

- [13] HTC VIVE. VIVE Pro 2, 2025. [online]. HTC Corporation, [cit. 2025-05-30]. Dostupné z: <https://www.vive.com/us/product/vive-pro2/overview/>
- [14] META. Meta Quest 3: Průlomová smíšená realita. Bezkonkurenční výkon, 2025. [online]. Meta, [cit. 2025-05-30]. Dostupné z: <https://www.meta.com/quest/quest-3/?srslid=AfmBOoo4oYRuGkY7nBSlZmPclwjszmmFdw7nodJdo1iHjKYfqyyMirry>
- [15] HUANG, Hsinfu a Chang-Franw LEE. Factors affecting usability of 3D model learning in a virtual reality environment. Interactive Learning Environments [online]. 2022, 30(5), s. 848-861 [cit. 2025-05-30]. ISSN 1049-4820.
- [16] PRŮŠA, Josef. Ze 2D do 3D: Jak udělat 3D model z obrázku nebo fotografie, 2025. [online]. Prusa Research a.s., poslední aktualizace 14. 3. 2024 [cit. 2025-05-30]. Dostupné z: <https://josefprusa.cz/ze-2d-do-3d-jak-udelat-3d-model-z-obrazku-nebo-fotografie/>
- [17] AUTODESK. Postup importu nebo vložení souborů PDF, 2025. [online]. Autodesk, ©2020 [cit. 2025-05-30]. Dostupné z: <https://help.autodesk.com/view/ACD/2021/CSY/?guid=GUID-C0EBD080-D074-4AD5-A508-51F208827E97>
- [18] BLENDER FOUNDATION. blender.org - Home of the Blender project - Free and Open 3D Creation Software, 2025. [online]. Blender Foundation, [cit. 2025-05-30]. Dostupné z: <https://www.blender.org>
- [19] INKSCAPE. Inkscape, 2025. [online]. Inkscape, [cit. 2025-05-30]. Dostupné z: <https://inkscape.org>
- [20] How To Turn JPG image to 3D Floor Plan - Blender, 2025. [online]. [cit. 2025-05-30]. Dostupné z: <https://www.youtube.com/watch?v=mq63GWbgWdM&t>
- [21] UNITY TECHNOLOGIES. Unity Real-Time Development Platform | 3D, 2D VR & AR Engine, 2025. [online]. Unity Technologies, ©2025 [cit. 2025-05-30]. Dostupné z: <https://unity.com>
- [22] EPIC GAMES. Unreal Engine 5. [online]. Epic Games, Inc., ©2004-2025 [cit. 2025-05-30]. Dostupné z: <https://www.unrealengine.com/en-US/unreal-engine-5>
- [23] GOOGLE. Stáhnout Android Studio a nástroje SDK, 2025. [online]. Google, [cit. 2025-05-30]. Dostupné z: <https://developer.android.com/studio>

- [24] KODERZ. Runtime Mesh Component. [online]. Epic Games, Inc., ©2004-2025 [cit. 2025-05-30]. Dostupné z: <https://forums.unrealengine.com/marketplace/en-US/product/runtime-mesh-component>
- [25] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management V. Teorie a praxe ochrany majetku a fyzické bezpečnosti. Zlín: Radim Bačuvčík – VerBUM, 2015. ISBN 978-80-87500-67-5.
- [26] AUTODESK. Maya Overview, 2025. [online]. Autodesk, ©2025 [cit. 2025-05-30]. Dostupné z: <https://www.autodesk.com/products/maya/overview?term=1-YEAR&tab=subscription>
- [27] MAXON. Maxon ZBrush - The Industry Standard Digital Sculpting Software, 2025. [online]. Maxon Computer GmbH, ©2024 [cit. 2025-05-30]. Dostupné z: https://www.maxon.net/en/zbrush?srsId=AfmBOoqpl0AR4U_TQMBJtgbf72qYRI4EZ6EA8ZY4MV-3EePiJQAKxzwP

SEZNAM OBRÁZKŮ

Obrázek 1 PIR detektor.....	47
Obrázek 2 MW detektor.....	47
Obrázek 3 List komponentů.....	48
Obrázek 4 Levely.....	52
Obrázek 5 Main menu level.....	53
Obrázek 6 Editor map level.....	54
Obrázek 7 VR main menu level.....	55
Obrázek 8 VRmisto.....	56
Obrázek 9 Nastavení světa.....	57
Obrázek 10 Nastavení pawn.....	58
Obrázek 11 Inputy.....	59
Obrázek 12 Input maping.....	60
Obrázek 13 Modely.....	61
Obrázek 14 Barva zvýraznění.....	62
Obrázek 15 Bílé PIR.....	62
Obrázek 16 Červené PIR.....	62
Obrázek 17 Preset.....	64
Obrázek 18 Data table1.....	66
Obrázek 19 Data table2.....	66
Obrázek 20 Cast to.....	68
Obrázek 21 Set.....	68
Obrázek 22 Branch.....	69
Obrázek 23 Sekvence.....	69
Obrázek 24 Loop.....	70
Obrázek 25 Enhanced Input Action.....	71
Obrázek 26 Pohyb1.....	72
Obrázek 27 Pohyb2.....	73
Obrázek 28 Pohyb3.....	73
Obrázek 29 Pohyb4.....	74
Obrázek 30 Detekce1.....	75
Obrázek 31 Detekce2.....	76
Obrázek 32 Detekce 3.....	76
Obrázek 33 Výběr bloků.....	77
Obrázek 34 Rotace1.....	78

Obrázek 35 Rotace2	79
Obrázek 36 Mazni1	80
Obrázek 37Mazani2	80
Obrázek 38 Nastavitelny1	81
Obrázek 39 Nastavitelny2.....	82
Obrázek 40 Nastavitelny3.....	82
Obrázek 41 Import1	84
Obrázek 42 Import2	85
Obrázek 43 Import3	85
Obrázek 44I Import4.....	86
Obrázek 45 Import5	86
Obrázek 46 Ulozen1	87
Obrázek 47 Ulozena2.....	87
Obrázek 48 Ulozena3.....	88
Obrázek 49 Ulozena4.....	88
Obrázek 50 Ulozeno5	88
Obrázek 51 Tvorba menu1	91
Obrázek 52 Tvorba menu2	92
Obrázek 53 Tvorba UI1	93
Obrázek 54 Tvorba UI2	93

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CAD Computer-Aided Design

BIM Building Information Modeling

PBR Physically Based Rendering

NURBS Non-Uniform Rational B-Splines

VR Virtual Reality

LOD Level of Detail

SDK Software Development Kits

PZTS Poplachové zabezpečovací a tísňové systémy

EPS Elektrická požární signalizace

DWG Drawing

DXF Drawing Exchange Format

OBJ Object

FBX Filmbox

IFC Industry Foundation Classes

SEZNAM PŘÍLOH

Příloha P I: AHVR

Příloha P II: VRapp

PřílohaP III: Dapp

Příloha P IV: Assety

Příloha P V: Exportační file