Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Multiplayerová 3D hra s využitím moderního herního enginu

Bakalářská práce

2025 Daniel Šimek

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2024/2025

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Daniel Šimek**
Osobní číslo:	**I22135**
Studijní program:	**B0688A140009 Informační technologie**
Téma práce:	**Multiplayerová 3D hra s využitím moderního herního enginu**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování

Cílem bakalářské práce je navrhnout a implementovat multiplayerovou 3D hru žánru CO-OP RPG, inspirovanou herními sériemi jako Diablo a KingsRoad a prvky z Dungeons & Dragons. Před zahájením hry bude vyžadována registrace uživatelů, která bude integrována do uživatelského rozhraní herního klienta bez potřeby potvrzovacích e-mailů. Po přihlášení se hráči objeví v lobby města, kde mohou vytvářet party a organizovat instance levelů. Každý hráč si zvolí specializaci, která ovlivní jeho dovednosti. V instancích budou hráči bojovat s nepřáteli, získávat zkušenosti a materiály na výrobu a řešit náhodné události, jako je otevírání truhel. V teoretické části student popíše použité technologie, žánr CO-OP RPG a problematiku programování her. Hra bude vyvinuta pomocí Unreal Engine 5, s programováním v jazyce C++ a skriptovacím jazykem Blueprints.

Rozsah pracovní zprávy: **min. 30 stran**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

DOBROVSKÝ, Pavel. 50 let videoher. Praha: Naked Dog ve spolupráci s Xzone.cz, 2023. ISBN 978-80--907964-8-5.
DOBROVSKÝ, Pavel; BACH, Martin; POLÁČEK, Petr a GRYGAR, Lukáš. O hrách a lidech. V Praze: Naked Dog, 2022. ISBN 978-80-907964-4-7.
ROMERO, Marcos a SEWELL, Brenden. Blueprints Visual Scripting for Unreal Engine 5: Unleash the true power of Blueprints to create impressive games and applications in UE5. 3rd edition. Packt Publishing, 2022. ISBN 978-1801811583.

Vedoucí bakalářské práce: **Ing. Jan Merta, Ph.D.**
Katedra softwarových technologií

Datum zadání bakalářské práce: **15. prosince 2024**
Termín odevzdání bakalářské práce: **16. května 2025**

L.S.

prof. Ing. Petr Doležel, Ph.D. v.r.
děkan

Ing. Jan Panuš, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 28. února 2025

Prohlašuji:

Práci s názvem Multiplayerová 3D hra s využitím moderního herního enginu jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

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

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 15.05.2025

Daniel Šimek v.r.

PODĚKOVÁNÍ

Rád bych poděkoval všem, kteří mi během tvorby této práce poskytli podporu, rady a cenné připomínky.

Velké poděkování patří panu doktorovi Janu Mertovi za odborné vedení, konstruktivní zpětnou vazbu a cenné rady, které mi pomohly lépe uchopit a realizovat tento projekt.

ANOTACE

Tato bakalářská práce se zabývá návrhem a implementací multiplayerové 3D hry v žánru CO-OP RPG s prvky inspirovanými sériemi Diablo, KingsRoad a pravidly Dungeons & Dragons. Cílem práce bylo vytvořit funkční prototyp hry s možností registrace a přihlášení hráče, lobby systémem pro vytváření part, systémem dovedností, instancováním výprav a základním bojovým systémem proti nepřátelům. Práce popisuje použitou technologii Unreal Engine 5. Součástí je také řešení síťové synchronizace a komunikace mezi klientem, serverem a databází prostřednictvím REST API.

KLÍČOVÁ SLOVA

hra, multiplayer, Unreal Engine, CO-OP RPG, REST API, databáze, síťová synchronizace

TITLE

Multiplayer 3D game using a modern game engine

ANNOTATION

This bachelor thesis focuses on the design and implementation of a multiplayer 3D game in the CO-OP RPG genre, inspired by the Diablo and KingsRoad series as well as Dungeons & Dragons mechanics. The goal was to create a working prototype of a game that includes user registration, a lobby system for party creation, a skill and specialization system, instanced levels with enemies, and a basic combat system. The thesis describes the Unreal Engine 5 technology. It also addresses challenges of network synchronization and communication between the game client, server, and database via a REST API.

KEYWORDS

game, multiplayer, Unreal Engine, CO-OP RPG, REST API, database, networking

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

UE / UE5 – Unreal Engine 5

API – Application Programming Interface

REST API – Representational State Transfer API

CO-OP – Cooperation

RPG – Role-Playing Game

FPS – First Person Shooter

D&D – Dungeons & Dragons

EF – Entity Framework

SQL – Structured Query Language

JSON – JavaScript Object Notation

UI – User Interface

LAN – Local Area Network

NPC – Non Playable Character

TERMINOLOGIE

Party – Skupina hráčů, kteří se spojí v lobby a vyrážejí společně do výpravy.

Výprava – Herní úroveň, ve které hráči bojují proti nepřátelům, získávají zkušenosti a materiály.

Specializace – Systém zaměření hráče, který ovlivňuje jeho atributy.

Replikace – Proces synchronizace dat mezi serverem a klienty v síťové hře.

Synergie schopností – Vzájemné doplňování účinků schopností mezi hráči, podporující týmovou spolupráci.

Subclass – Dílčí zaměření hráčské třídy, např. válečník → tank nebo barbar.

Boss – Silný nepřítel na konci instance, obvykle s unikátními schopnostmi a vyšší obtížností.

Threat systém – AI logika určující, na kterého hráče se nepřítel zaměří podle toho, koho vnímá jako největší hrozbu.

Blueprints – Vizuální skriptovací systém v UE5 umožňující rychlé prototypování bez nutnosti psaní kódu.

Lobby – Místo, kde se hráči scházejí před začátkem výpravy a sestavují party.

Cel-shaded – Styl vykreslování grafiky, který napodobuje vzhled ručně kreslených animací nebo komiksů. [1]

Low-Poly – Grafický styl charakterizovaný nízkým počtem polygonů (trojúhelníků), což vede k jednoduššímu a často stylizovanému vzhledu modelů. [2]

Blocking – Fáze raného vývoje, kdy se vytváří hrubé 3D modely a rozmístění prvků (např. stěn, překážek, důležitých bodů) za účelem otestování hratelnosti, rozložení a měřítka úrovně před přidáním detailů. Často se používají jednoduché tvary jako krychle a válce.

ÚVOD

Herní průmysl je jedním z nejrychleji se rozvíjejících odvětví zábavního průmyslu a multiplayerové hry se staly klíčovým prvkem moderní herní scény. Vývoj takových her však přináší řadu technických výzev, včetně síťové synchronizace, správy serverů a návrhu herních mechanik. Tato práce se zaměřuje na návrh a implementaci kooperativní RPG hry s důrazem na multiplayerovou komunikaci, herní mechaniky a interakci hráčů v týmovém prostředí. V rámci této práce budou popsány teoretické základy žánru CO-OP RPG, historie a technické aspekty multiplayerových her, role herních enginů a specifika Unreal Engine 5. Praktická část se pak bude věnovat návrhu a implementaci samotné hry, včetně herních mechanik a serverové architektury.

Videohry mě provázejí od dětství a postupně se z koníčku staly i profesním zájmem. Fascinuje mě, jak herní technologie umožňují vytvářet interaktivní světy a propojit hráče napříč celým světem. Vývoj multiplayerové hry představuje komplexní výzvu, která spojuje programování, herní design a síťové technologie. Cílem této práce je nejen vytvořit funkční prototyp, ale také prozkoumat osvědčené postupy při vývoji síťových her a aplikovat je v praxi. Hlavním cílem je návrh a implementace prototypu kooperativní RPG hry s podporou multiplayeru. Mezi hlavní úkoly patří vytvoření autentizačního systému, správa hráčských instancí a návrh herních mechanik, jako je soubojový systém, dovednostní specializace nebo výroba. Důraz bude kladen také na synchronizaci dat mezi klientem a serverem, optimalizaci výkonu a zabezpečení komunikace.

Práce se věnuje charakteristice a historii žánru CO-OP RPG, přehledu multiplayerových her, jejich významu a technickým výzvám spojeným s jejich vývojem. Zaměřuje se také na roli herních enginů, zejména Unreal Engine 5, jeho klíčové vlastnosti, síťovou komunikaci a integraci s databázemi. Dále popisuje návrh herního systému, včetně herních mechanik, soubojového systému, systému výroby a party systému, a věnuje se struktuře hry i architektuře serveru. Součástí práce je implementace autentizace, správy relací a synchronizace objektů. Závěr shrnuje dosažené výsledky, přínos práce a možnosti dalšího rozšíření. Tento dokument poskytuje ucelený pohled na návrh a implementaci multiplayerové RPG hry a nabízí řešení klíčových problémů spojených s vývojem síťových her.

1 Žánr CO-OP RPG

1.1 Definice a charakteristika žánru CO-OP RPG

Žánr kooperativních RPG her, známý pod zkratkou CO-OP RPG, vychází z principu spolupráce mezi hráči. Název „CO-OP“ pochází z anglického „cooperation“, tedy spolupráce, a tento prvek je klíčovou součástí hratelnosti. Hráči společně řeší herní výzvy, bojují proti nepřátelům, plní úkoly nebo sdílí herní svět. Tento žánr nevznikl až s digitálními hrami, ale má své kořeny už v tradičních deskových hrách. Mezi nejvýznamnější předchůdce patří Dungeons & Dragons, který položil základy nejen kooperativního hraní, ale také samotného RPG systému.

Obrázek 1 – Logo Dungeons & Dragons [3]

RPG neboli Role-Playing Game, znamená hru na hrdiny, kde hráč přebírá roli postavy a rozhoduje o jejím vývoji. To zahrnuje volbu povolání, specializací a dovedností, stejně jako možnost upravovat vzhled nebo ovlivňovat příběhové linie. Ve zmíněném Dungeons & Dragons si hráč může zvolit, zda se stane rytířem, mágem, lučištníkem či jiným povoláním, přičemž každá volba zásadně ovlivňuje styl hraní a možnosti v herním světě. Tento princip se přenesl i do digitálních her, kde hráči často vytvářejí unikátní postavy a v rámci kooperativní hry se podílejí na společném plnění úkolů či taktické spolupráci v boji.

1.2 Historický vývoj žánru, příklady her

Historie kooperativních RPG sahá až do 80. let, kdy se začaly objevovat první digitální adaptace stolních her, často inspirované klasickými tituly jako Dungeons & Dragons. Významný rozmach nastal v 90. letech s nástupem lokálního multiplayeru, a především s rozvojem LAN hraní, které umožnilo více hráčům spolupracovat v rámci jedné sítě. S příchodem rychlejšího internetu na přelomu tisíciletí se žánr přesunul do online prostoru, což otevřelo dveře rozsáhlejším kooperativním zážitkům a vzdálené spolupráci mezi hráči.

Mezi známé tituly, které výrazně ovlivnily tento žánr, patří například Diablo II, které hráčům umožnilo spojit síly v boji proti démonickým hordám. Později následovala série Borderlands, jež kombinuje RPG prvky s akční kooperativní hratelností v unikátním cel-shaded vizuálním stylu.

Dalším významným příkladem je Divinity: Original Sin 2, které přineslo tahový soubojový systém, důraz na volnost v řešení situací a možnost hrát celou kampaň ve dvou až čtyřech hráčích. Tato hra se stala duchovním předchůdcem novějšího titulu Baldur's Gate 3, který přejímá jádro deskové hry Dungeons & Dragons v její páté edici a věrně implementuje pravidla, včetně tahových soubojů, systému dovedností a kostkových hodů.

Baldur's Gate 3 umožňuje hráčům společně procházet rozsáhlý příběhový svět, interagovat s NPC postavami a ovlivňovat děj svými rozhodnutími, přičemž každý hráč může hrát unikátní roli ve skupině. Díky kombinaci hlubokého příběhu, strategických soubojů a komplexních možností roleplaye (hraní role, kdy hráč rozhoduje o chování a vývoji své postavy) se hra zařadila mezi nejvýznamnější tituly žánru a stala se milníkem v moderním herním designu.

Obrázek 2 – Screenshot ze hry Baldur's Gate 3 (zdroj vlastní)

2 Multiplayerové hry

Multiplayerové hry prošly od svého vzniku v 70. letech výrazným technologickým i designovým vývojem. První formy víceuživatelského hraní se objevily v podobě textových her známých jako MUD (Multi-User Dungeon), které hráčům umožňovaly sdílet virtuální světy a komunikovat prostřednictvím textových příkazů.

V 80. letech se multiplayer stal běžnou součástí arkádových automatů a domácích konzolí. Hráči často sdíleli jednu obrazovku a ovladače, ať už v kooperativních nebo versus režimech. S příchodem 90. let a rozvojem osobních počítačů i lokálních sítí (LAN) se prosadily první akční tituly, jako například Doom nebo Quake, které umožnily hráčům soupeřit v reálném čase na lokální síti. Zároveň se začaly objevovat první online multiplayerové hry, z nichž některé, jako Ultima Online (1997) nebo EverQuest (1999), položily základy pro vznik a rozvoj žánru MMORPG.

V období let 2000–2010 se online multiplayer stal standardní součástí herního světa. Významný vliv na to měly nové generace konzolí a služby jako Xbox Live a PlayStation Network, které umožnily hráčům snadné a stabilní připojení k online zápasům. Masovou popularitu si získal také žánr MMORPG – zejména World of Warcraft (2004), který definoval herní a sociální standardy moderních online her.

Od roku 2010 se multiplayerové hraní neustále rozšiřuje. Na významu získaly kompetitivní eSporty s profesionálními ligami a milionovými turnaji, v nichž dominují tituly jako League of Legends, Dota 2 či Counter-Strike: Global Offensive. Zároveň vznikl nový a velmi populární žánr Battle Royale, který reprezentují hry jako Fortnite a PUBG. Tyto tituly umožňují stovkám hráčů soupeřit o přežití v jednom herním kole. Díky rostoucí podpoře tzv. cross-platform hraní je dnes běžné, že hráči z různých zařízení – ať už PC, konzolí nebo mobilních telefonů – mohou hrát společně bez ohledu na platformu.

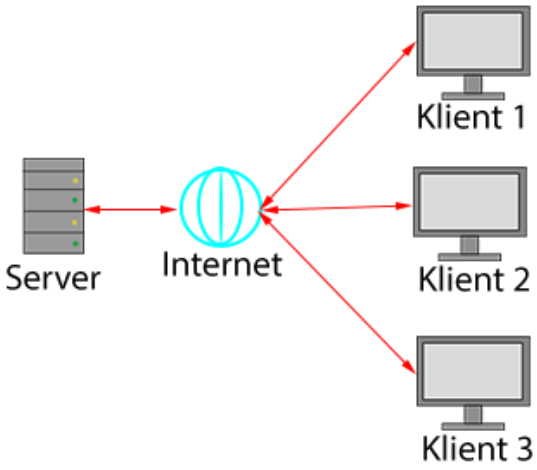
Multiplayerové hry se tak z původně jednoduchých experimentálních projektů proměnily v komplexní virtuální světy s propracovanou infrastrukturou, které tvoří nedílnou součást moderní digitální zábavy. [4, 5, 6]

2.1 Druhy multiplayeru

Multiplayer může být realizován několika způsoby, z nichž nejčastější jsou:

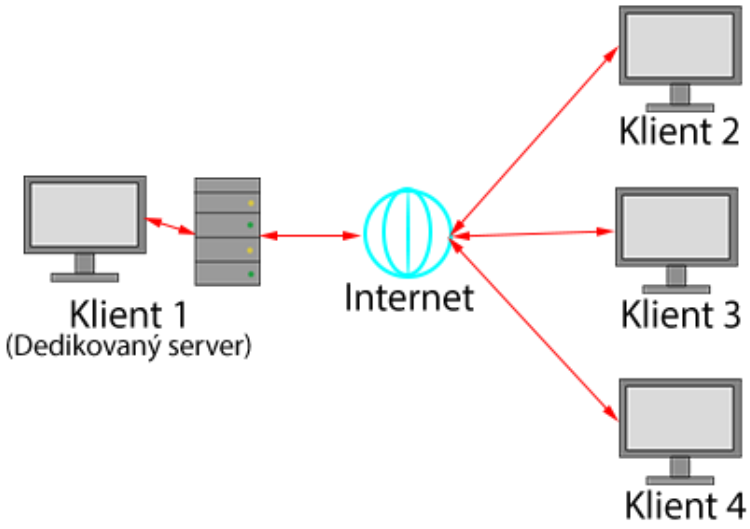
- **Server-klient**
- **Dedikovaný server-klient**
- **LAN (Local Area Network)**
- **Splitscreen (rozdělená obrazovka)**
- **Hotseat (střídavý režim)**

V modelu server-klient hra běží na vyhrazeném serveru, který není řízen žádným z hráčů, ale je umístěn v datových centrech provozovatele hry. Tento model poskytuje stabilní připojení, efektivní synchronizaci dat a lepší ochranu proti podvodům, což je ideální pro velké online hry s množstvím hráčů. [7]



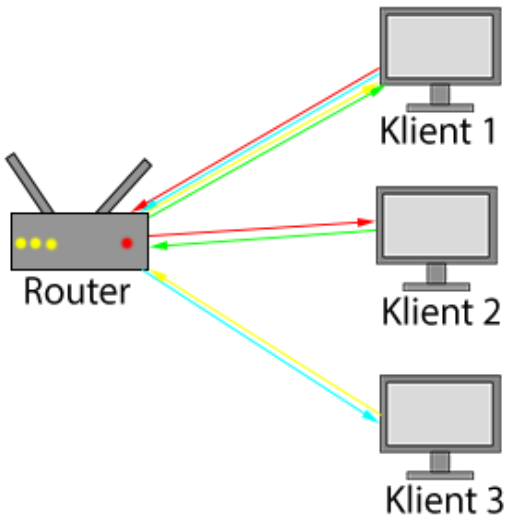
Obrázek 3 – Připojení server x klient (zdroj vlastní)

Na druhé straně dedikovaný server-klient znamená, že jeden z hráčů funguje jako hostitel (server) a ostatní se k němu připojují jako klienti. Tento model se často používá u menších her nebo tam, kde si hráči chtějí vytvořit vlastní soukromou hru. Nevýhodou je, že pokud hostitel hru opustí nebo má nestabilní připojení, hra se může přerušit nebo být ukončena.



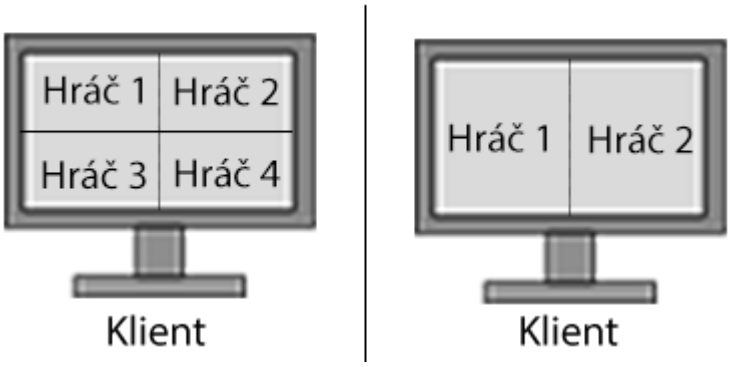
Obrázek 4 – Připojení dedikovaný server x klient (zdroj vlastní)

LAN multiplayer umožňuje hráčům připojit se ke hře v rámci jedné lokální sítě, například v domácnosti nebo na LAN party. Tento typ připojení obvykle nevyžaduje internet a poskytuje velmi nízkou latenci a stabilní připojení. LAN režim je populární zejména u herních komunit nebo v prostředí, kde není dostupné kvalitní internetové připojení.



Obrázek 5 – Připojení LAN (zdroj vlastní)

Specifickou formou multiplayeru je splitscreen, kdy se obrazovka rozdělí na několik částí a umožňuje více hráčům hrát na jednom zařízení. Například u dvou hráčů je obrazovka rozdělena na levou a pravou polovinu, u čtyř hráčů bývá rozdělena do čtyř menších sekcí.



Obrázek 6 – Připojení splitscreen (zdroj vlastní)

Hotseat je typ lokálního multiplayeru, kdy se více hráčů střídá u jednoho zařízení – například po provedení tahu nebo akce. Tento způsob je typický pro tahové strategie, kde každý hráč odehraje svůj tah a poté předá ovládání dalšímu hráči.

2.2 Příklady multiplayerových her podle způsobu připojení

- **Server–Klient**
    - **World of Warcraft** – Masivně multiplayerová online RPG, která využívá centralizované servery Blizzardu. Tento model zajišťuje stabilní propojení desítek tisíc hráčů současně.
    - **Fortnite** – Rychlá battle royale hra, kde servery spravují matchmaking i průběh každé hry.
- **Dedikovaný server–Klient**
    - **Dead by Daylight** – Hororová hra, kde se hráči připojují k centrálním dedikovaným serverům kvůli spravedlivému hostování.
    - **Rust** – Survival sandbox, kde běh a persistenci světa zajišťují dedikované servery.
    - **Valheim** – Hráči si mohou spustit vlastní dedikovaný server pro survival kooperaci až 10 hráčů.

- **LAN**
    - **Heroes of Might and Magic III** – Umožňuje připojení přes lokální síť pro souboje nebo kooperaci bez nutnosti internetu.
    - **Age of Empires II** – Klasická RTS, která se často hrála na LAN party pro soutěžní 1v1 nebo týmové bitvy.
    - **Minecraft** – Umožňuje hostování vlastního světa v místní síti s plnou kontrolou.
- **Splitscreen**
    - **Overcooked! 2** – Kooperativní chaos v kuchyni, ideální pro hraní na jednom zařízení.
    - **Rocket League** – Nabízí možnost hrát až ve čtyřech hráčích na jedné obrazovce.
    - **A Way Out** – Hra navržená primárně pro splitscreen kooperaci dvou hráčů.
- **Hotseat**
    - **Heroes of Might and Magic III** – Typická tahová strategie, kde se hráči střídají u jednoho zařízení.
    - **Worms Armageddon** – Tahová akce, kde každý hráč ovládá svůj tým červíků v kole.
    - **Civilization VI** – Tahová strategie, která podporuje hotseat multiplayer až pro 12 hráčů.

2.3 Výzvy spojené s vývojem multiplayerových her

Vývoj multiplayerových her přináší několik technických výzev, které je třeba efektivně řešit, aby byla zajištěna plynulost a férovost herního zážitku. Mezi klíčové aspekty patří latence, tedy zpoždění v komunikaci mezi hráči a serverem. Příliš vysoká latence může způsobit nesoulad mezi akcemi hráče a tím, co vidí ve hře. Pro minimalizaci latence se využívají optimalizované síťové protokoly, prediktivní algoritmy kompenzující zpoždění a geograficky distribuované servery (tzv. edge computing), které umožňují hráčům připojit se k nejbližšímu dostupnému serveru (pro Českou republiku se tyto servery nejčastěji nachází v Německu). [8]

Další výzvou je synchronizace dat, tedy zajištění toho, aby všichni hráči viděli stejný stav hry v reálném čase. K tomu se využívají různé metody replikace objektů a optimalizace síťového provozu, například komprese dat, delta komprese (přenášení pouze změn místo celých stavů) nebo přednostní synchronizace klíčových herních prvků. Některé hry také aplikují techniku lag compensation, která analyzuje a zpětně upravuje herní stavy na základě latence hráčů, aby minimalizovala nesrovnalosti.

Významným aspektem multiplayerových her je bezpečnost a ochrana proti podvodům. Online hry jsou často terčem hackerů a cheaterů, kteří se snaží získat neférovou výhodu pomocí externích nástrojů, modifikací klienta nebo manipulace se síťovou komunikací. K ochraně proti těmto hrozbám se využívají metody jako šifrování datových přenosů, ověřování integrity souborů hry, server-side validace důležitých akcí (aby se zabránilo manipulaci na straně klienta), systémy detekce cheatů, které monitorují neobvyklé vzorce chování hráčů anebo manuální kontroly, kdy administrátor kontroluje akce uživatele a případné podvodníky zabanuje. Některé moderní hry rovněž využívají systémy založené na AI, které jsou schopné identifikovat podezřelé aktivity a automaticky zabanovat podvodníky. Velice známým způsobem prevence botování v počátcích hry World of Warcraft byl takzvaný "rock anticheat". Tento mechanismus spočíval v tom, že administrátor, který měl podezření na přítomnost bota, umístil do herního světa kámen. Pokud se jednalo o bota, ten pokračoval ve své naskriptované cestě a stále běhal přímo do umístěného kamene, protože jeho chování bylo automatizované a nedokázal se přizpůsobit. Naopak skutečný hráč by kámen oběhl, čímž bylo možné snadno rozpoznat, zda se jedná o lidského hráče nebo o bota. Tento jednoduchý, velice kreativní, ale účinný způsob sloužil jako metoda detekce a ochrany proti automatizovaným programům, které obcházejí pravidla hry. [9, 10, 11]

Další důležitou oblastí je matchmaking a balancing, tedy hledání vhodných soupeřů a vyvažování herního prostředí tak, aby byla zajištěna férová a zábavná hra pro všechny účastníky. Systémy matchmakingu obvykle pracují na základě hodnotících algoritmů, které analyzují výkon hráčů a přiřazují je k protivníkům odpovídající úrovně. Pro zajištění vyváženého zážitku se také implementují mechaniky dynamického přizpůsobování obtížnosti, kdy hra upravuje určité prvky na základě dovedností hráče, například změnou chování nepřátel nebo přizpůsobením herních odměn. [12]

S rostoucím počtem hráčů se stává klíčovou otázkou škálovatelnost serverů, tedy schopnost herní infrastruktury zvládnout zvýšenou zátěž bez poklesu výkonu. Moderní hry často využívají cloudová řešení a dynamickou alokaci serverových zdrojů podle aktuální potřeby. Jednou z účinných technik je instance-based hosting, kdy se servery spouští pouze pro aktivní hráče a po opuštění hry se automaticky uvolňují, čímž se optimalizuje využití hardwaru. Další metodou je load balancing, který rovnoměrně rozkládá zatížení mezi více serverů, aby se zabránilo přetížení jednoho uzlu.

3 Herní Engine

Herní engine je softwarový framework, který poskytuje vývojářům nástroje a funkce potřebné pro tvorbu videoher. Zahrnuje různé moduly, jako je renderování 2D a 3D grafiky, simulace fyziky, zvukový systém, správu animací, detekci kolizí, řízení kamery, síťovou komunikaci a správu herních objektů. Díky hernímu enginu mohou vývojáři soustředit svůj čas na samotný design a herní mechaniky, aniž by museli od základu programovat všechny technologické části hry. Kromě toho herní enginy často obsahují vizuální editory, které umožňují návrh prostředí, úpravy materiálů a testování přímo v editoru. To výrazně zkracuje vývojový cyklus a zlepšuje spolupráci mezi vývojáři, designéry a animátory. Některé enginy poskytují také nástroje pro tvorbu uživatelského rozhraní, správu assetů a buildování projektu pro různé platformy.

3.1 Role herního enginu při vývoji her

Role herního enginu při vývoji her je naprosto klíčová. Umožňuje efektivní práci s grafikou, zvukem, umělou inteligencí, fyzikálními simulacemi a multiplayerovou komunikací. Moderní herní enginy navíc často nabízejí pokročilé funkce jako je real-time ray tracing, globální iluminace, simulace kapalin, částicové systémy nebo nástroje pro machine learning, které se dají použít například pro chytré AI. [13]

Důležitou roli hraje také optimalizace – enginy se starají o efektivní využití paměti, vykreslování jen potřebných objektů (culling), level streaming a další techniky, které umožňují hru spustit na různých typech zařízení – od výkonných herních počítačů až po mobilní telefony nebo VR headsety.

Některé enginy, jako např. Unity nebo Unreal Engine, jsou univerzální a podporují široké spektrum herních žánrů i platforem. Jiné jsou naopak specializované – např. Godot je oblíbený mezi indie vývojáři pro svou jednoduchost a otevřenost, zatímco CryEngine se zaměřuje na realistické vizuály.

3.2 Využití herních enginů mimo herní průmysl

Herní enginy, jako je Unreal Engine, nacházejí široké uplatnění i mimo oblast videoher. Níže jsou uvedeny některé z oblastí, kde se tyto technologie úspěšně využívají:

- **Filmový průmysl** – Unreal Engine byl klíčovým nástrojem při výrobě seriálu The Mandalorian. Tvůrci využili technologii StageCraft, která umožnila promítat realistické digitální prostředí na obrovské LED stěny, čímž eliminovali potřebu natáčení na skutečných lokacích. Tato metoda umožnila hercům interagovat s dynamickým prostředím v reálném čase, což zefektivnilo produkční proces. [14, 15]

- **Architektonické vizualizace** – Architekti využívají Unreal Engine pro tvorbu interaktivních 3D modelů budov a interiérů. Tento nástroj umožňuje realistické zobrazení materiálů, osvětlení a prostředí, což klientům poskytuje detailní představu o finálním vzhledu projektů. Funkce jako Lumen pro realistické osvětlení a Nanite pro detailní geometrii usnadňují tvorbu vysoce kvalitních vizualizací. [16]

- **Automobilový průmysl** – Společnosti v automobilovém průmyslu využívají Unreal Engine pro simulace a testování vozidel v různých prostředích. Například platforma SCANeR od AVSimulation integruje Unreal Engine pro vytváření realistických jízdních simulací, což pomáhá při vývoji a testování autonomních vozidel. [17]

- **Vojenský výcvik** – Unreal Engine je využíván pro tvorbu taktických vojenských simulací, které umožňují realistický výcvik vojáků v bezpečném virtuálním prostředí. Společnost SimCentric například vyvinula aplikace pro výcvik v oblasti společných paleb a blízké letecké podpory, které jsou postaveny na Unreal Engine a poskytují flexibilní a škálovatelná řešení pro vojenský výcvik. [18]

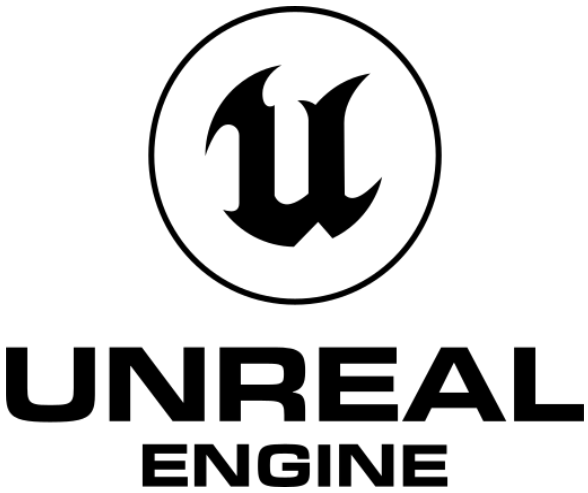
3.3 Příklady nejznámějších enginů a her v nich vytvořených

- **Unreal Engine** – Výkonný engine vhodný pro fotorealistickou grafiku a komplexní hry. Používá jazyk C++ pro implementaci herní logiky a Blueprints pro vizuální skriptování a interakce.
    - Fortnite, Gears of War, Final Fantasy VII Remake
- **Unity** – Oblíbený engine pro širokou škálu her, od 2D až po 3D tituly. Využívá programovací jazyk C#.
    - Hollow Knight, Subnautica, Cuphead

- **Godot** – Open-source engine s flexibilním nástrojem pro skriptování a nízkými systémovými nároky. Používá programovací jazyk C# pro komplexní skriptování a GodotScript pro jednodušší implementaci herní logiky.
    - Brotato
- **CryEngine** – Engine zaměřený na realistickou grafiku s pokročilými možnostmi vykreslování. CryEngine využívá C++ pro implementaci herní logiky a náročné grafické funkce, jako je simulace fyziky a rendering. Lua a Python jsou používány pro skriptování herních událostí a AI
    - Crysis, Hunt: Showdown
- **REDengine** – Engine vyvinutý studiem CD Projekt Red pro jejich RPG tituly. Další vývoj tohoto engine byl přerušen a společnost CDPR pro své budoucí herní tituly přechází na Unreal Engine. REDengine využívá C++ pro vykreslování grafiky a implementaci herní logiky. Pro skriptování herních událostí a interakcí je používán vlastní skriptovací jazyk.
    - Zaklínač 3: Divoký hon, Cyberpunk 2077
- **Divinity Engine** – Engine vyvinutý studiem Larian Studios, přizpůsobený pro tvorbu izometrických RPG s propracovaným systémem interakcí a tahových soubojů. Pro skriptování je využíván jazyk Lua, který je flexibilní a umožňuje rychlé úpravy a ladění. C++ je používán pro optimalizaci výkonu a náročné herní mechaniky.
    - Divinity: Original Sin 2, Baldur's Gate 3
- **RAGE (Rockstar Advanced Game Engine)** – Engine používaný studiem Rockstar Games pro otevřené světy s detailní simulací prostředí. RAGE engine používá C++ pro základní herní logiku a pokročilou grafiku. C# a Lua se používají pro skriptování herních mechanik, ovládání postav a interakci s prostředím.
    - Grand Theft Auto V, Red Dead Redemption 2

4 Unreal Engine 5

Unreal Engine 5 je jedním z nejpokročilejších herních enginů současnosti, vyvinutý společností Epic Games. Díky moderním technologiím poskytuje vývojářům nástroje pro tvorbu vizuálně působivých a technicky vyspělých her, a to jak pro singleplayer, tak i multiplayer tituly.



Obrázek 7 – Unreal Engine 5 logo [19]

4.1 Klíčové funkce a výhody

Mezi klíčové funkce Unreal Engine patří vysoká úroveň flexibility a podpora multiplatformního vývoje. Engine je schopen běžet na širokém spektru platforem, od PC, konzolí až po mobilní zařízení, což vývojářům umožňuje snadnou adaptaci a optimalizaci her pro různé platformy. Skriptovací jazyk Blueprints je silný nástroj, který zjednodušuje vývoj pro začínající programátory nebo pro rychlé prototypování, což umožňuje i nezkušeným vývojářům rychle vytvářet komplexní herní logiku bez potřeby psát složitý kód.

Jedním z nejvýznamnějších vylepšení oproti předchozím verzím enginu jsou technologie Nanite a Lumen. Nanite je virtuální mikropolygonová geometrie, která umožňuje vykreslování extrémně detailních modelů bez výrazného dopadu na výkon. Díky této technologii lze přímo ve hře používat vysoce detailní modely, aniž by bylo nutné je složitě optimalizovat. Lumen je dynamický osvětlovací systém, který umožňuje realistické globální osvětlení a odrazy světla v reálném čase.

Další výhodou je integrovaná podpora pro VR a AR (virtuální a rozšířenou realitu), která umožňuje vývojářům snadno vytvářet aplikace pro nové a inovativní zážitky v těchto oblastech. Unreal Engine také nabízí silné nástroje pro animaci a fyzikální simulace, což usnadňuje práci s realistickým pohybem postav, objektů a prostředí.

4.2 Nevýhody

Jedním z hlavních problémů je vyšší náročnost na hardware. Technologie jako Nanite a Lumen, které umožňují vykreslování extrémně detailních modelů a dynamické osvětlení v reálném čase, mohou být velmi náročné na výkon. To znamená, že pro plné využití všech funkcí enginu je potřeba výkonný hardware, což může být problém pro menší studia nebo vývojáře, kteří nemají přístup k silné počítačové infrastruktuře. Další nevýhodou je, že Unreal Engine, i přes své pokročilé technologie, není vždy optimalizovaný pro všechny platformy. Tento problém s optimalizací může způsobit, že hry běží hůř na slabších systémech, i když jsou správně nastavené, což může ovlivnit celkový výkon a stabilitu hry.

Další nevýhodou je komplexita pro začátečníky. I když Blueprints usnadňuje práci pro začátečníky, samotný engine a jeho možnosti mohou být pro nové vývojáře poměrně složité. Učení se kódu v C++ v Unreal Engine může být výzvou, zejména kvůli rozsáhlým API a struktuře projektu, což může vést k delší době na osvojení si enginu a efektivní práci s ním.

Unreal Engine je známý tím, že vytváří projekty s poměrně velkými velikostmi souborů, což může být problém zejména při distribuci nebo optimalizaci pro zařízení s omezeným úložným prostorem. Hry vytvořené v Unreal Engine často obsahují velké množství textur, modelů, zvukových souborů a dalších assetů, které mohou výrazně zvětšit velikost finálního projektu. To vede k delším časům stahování a instalace, což může být překážkou pro hráče s pomalejším internetovým připojením nebo omezeným úložným prostorem.

Pro příklad, můj projekt na disku má velikost 73 GB (z toho pouze kolem 100MB tvoří mé assety do hry), což je běžné pro středně velký projekt v Unreal Engine. Když se podíváme na samotný source code Unreal Engine, ten zabírá až 324 GB, což zahrnuje kompletní kód enginu, všechny jeho nástroje a knihovny. Většinu místa zabírá cache. Po zkompilování hry a vytvoření finálního build souboru velikost výsledné hry na disku činí přibližně 400MB (debug build 2,5 GB).

Další nevýhodou je licencování. Ačkoliv Unreal Engine je zdarma pro vývoj, pro komerční využití je nutné platit poplatky z příjmů z prodeje hry, pokud hra překročí určitý zisk (v současnosti 1 milion USD). To může být nevýhodné pro některé vývojáře nebo malé studia, která chtějí zachovat větší část svého výdělku. [20]

Pro menší projekty může být Unreal Engine také nadbytečně složitý. Je navržen pro tvorbu velkých, komplexních her, takže pro menší 2D hry nebo jednoduché mobilní aplikace mohou být jeho pokročilé funkce nevyužité. V takových případech mohou vývojáři sáhnout po jednodušších enginech, jako je Unity.

4.3 Blueprints vs C++

Unreal Engine 5 nabízí dva hlavní způsoby programování herní logiky – vizuální skriptovací systém Blueprints a klasické programování v C++.

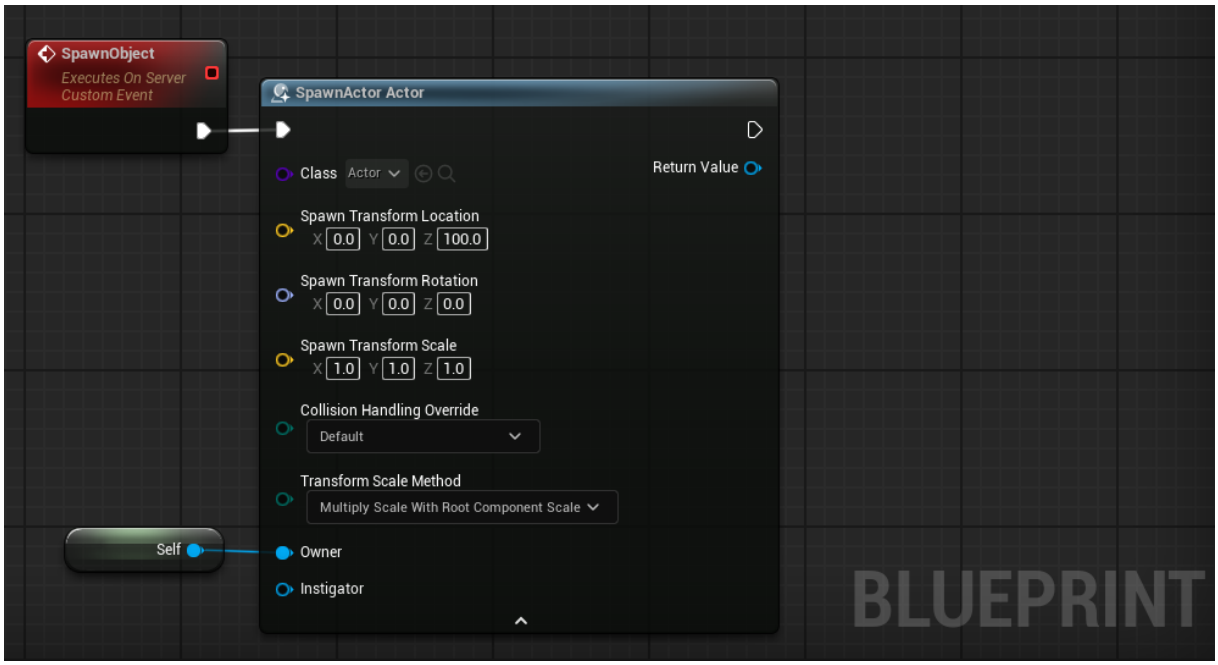
Blueprints jsou vizuální nástroj umožňující rychlé prototypování herních mechanik bez nutnosti psát kód. Jsou intuitivní, snadno pochopitelné i pro začátečníky a umožňují rychlé iterace v návrhu hry. V Blueprints je také podstatně jednodušší práce s uživatelským rozhraním. Ve skutečnosti však nejsou plnohodnotným programovacím jazykem, ale spíše vizuální reprezentací kódu, který Unreal Engine interně překládá do nativního C++.

C++ poskytuje větší flexibilitu a výkon. Používá se zejména pro optimalizaci náročnějších částí hry, správu paměti nebo práci s jádrem enginu. Výhodou je také možnost integrace s dalšími systémy, například databázemi nebo síťovými službami. Nicméně i zde je třeba zmínit, že Unreal Engine si vlastní C++ kód zpracovává svým způsobem – například generuje vlastní reflektivní systém a speciální makra, která umožňují lepší propojení s enginem.

Na obrázcích níže je ukázán příklad dynamického vytvoření objektu a následné umístění do herního světa v C++ a Blueprint.

```cpp
void AGameServer::SpawnObject()
{
    // Definice místa, kde se nová instance objeví - v tomto případě 1 metr nad bodem (0,0,0)
    const FVector SpawnLocation( InX: 0.0f,  InY: 0.0f,  InZ: 100.0f);

    // Nastavení rotace instance - žádná rotace
    const FRotator SpawnRotation( InPitch: 0.0f,  InYaw: 0.0f,  InRoll: 0.0f);

    // Parametry pro spawnování instance
    FActorSpawnParameters SpawnParams;

    // Nastavíme tohoto server jako vlastníka nově vytvořené instance
    SpawnParams.Owner = this;

    // Řešení kolizí při spawnování - vždy vytvořit, i pokud by kolidoval s jiným objektem
    SpawnParams.SpawnCollisionHandlingOverride = ESpawnActorCollisionHandlingMethod::AdjustIfPossibleButAlwaysSpawn;

    // Samotné vytvoření instance ve světě - použije se výchozí třída AActor
    GetWorld()->SpawnActor<AActor>(
        AActor::StaticClass(), // Třída, které chceme vytvořit instanci
        SpawnLocation, // Pozice ve světě
        SpawnRotation, // Rotace ve světě
        SpawnParams // Další parametry (kolize, vlastník, atd.)
    );
}
```

Obrázek 8 – Vytvoření instance třídy AActor a umístění do světa pomocí C++ (zdroj vlastní)



Obrázek 9 – Vytvoření instance třídy AActor a umístění do světa pomocí Blueprint (zdroj vlastní)

Kombinace Blueprints a C++ je běžným přístupem – základní logika a rychlé prototypování se často realizují v Blueprints, zatímco náročnější výpočty a optimalizace probíhají v C++. Obojí tak lze vnímat jako určitý druh „pseudokódu“, který Unreal Engine dále transformuje do své vnitřní optimalizované podoby.

4.4 Verze Unreal Engine: Source Code vs. Editor

Unreal Engine nabízí dvě hlavní verze – verzi se zdrojovým kódem (Source Code) a verzi s pouze editorem (Binary). Tyto dvě verze se liší nejen způsobem distribuce, ale i v možnostech, které vývojářům poskytují, a mohou mít zásadní dopad na vývojový proces, zejména pokud jde o programování, flexibilitu a multiplayerové možnosti.

4.4.1 Source code verze

Verze Unreal Engine se zdrojovým kódem je dostupná na GitHubu a umožňuje vývojářům přístup k plnému kódu enginu, který mohou upravovat a přizpůsobit podle potřeb svého projektu. Tato verze je ideální pro vývojáře, kteří chtějí mít plnou kontrolu nad samotným enginem a potřebují implementovat specifické funkce, optimalizace nebo dokonce měnit základní chování enginu. Výhodou je, že v této verzi můžete používat C++ kód pro vývoj herní logiky, což poskytuje vyšší výkon a flexibilitu. C++ je zvláště výhodný pro složité herní mechaniky, optimalizaci náročných částí hry, nebo pro implementaci složitějších síťových a multiplayerových funkcí, kde je potřeba detailní kontrola nad chováním klienta a serveru.

Přístup k zdrojovému kódu znamená, že můžete upravit jakýkoli aspekt enginu, což je zvláště užitečné v případě potřeby přizpůsobení specifických nástrojů nebo funkcí pro váš projekt. Tato verze také umožňuje plnou kontrolu nad multiplayerovými mechanismy, kde si můžete přizpůsobit serverovou logiku, optimalizovat síťovou komunikaci, nebo upravit základy synchronizace mezi klienty a servery. Každá aktualizace enginu vyžaduje, abyste opětovně zkompilovali kód, což může být časově náročné a složité. Pro menší týmy nebo jednotlivce může být správa této verze výzvou.

4.4.2 Editor verze

Verze s již zkompilovaným editorem je jednodušší variantou Unreal Engine, která je distribuována jako hotová aplikace, kterou si můžete stáhnout a nainstalovat. Tento editor je vlastně výstup zkompilovaného zdrojového kódu Unreal Engine. Tato verze je určena pro vývojáře, kteří se nechtějí zabývat modifikacemi samotného enginu, ale soustředí se především na tvorbu herní logiky, designu a obsahu prostřednictvím vizuálního nástroje Blueprints. Blueprints je intuitivní skriptovací jazyk, který nevyžaduje znalosti tradičního programování a je ideální pro rychlé prototypování herních mechanik, interakcí a uživatelského rozhraní.

Tato verze neumožňuje přímý přístup k C++ kódu, což znamená, že všechny komplexní herní mechaniky nebo optimalizace je nutné realizovat pomocí Blueprints. I když Blueprints jsou výkonné a umožňují rychlý vývoj, nemají stejnou úroveň kontroly a výkonu jako C++, což může být omezení pro náročnější projekty nebo pro implementaci specifických multiplayerových funkcí.

Pokud jde o multiplayer, verze s pouze editorem (Binary) může mít omezené možnosti, zejména pokud jde o správu složitějších síťových funkcí. I když Unreal Engine poskytuje určité nástroje pro práci s Blueprints v síťovém prostředí (například pro synchronizaci stavů mezi klienty a servery), pro větší kontrolu nad síťovou logikou a optimalizací výkonu bude opět nutné přejít na verzi se zdrojovým kódem a využívat C++.

4.5 Unreal Engine Networking

Síťová komunikace je klíčovým prvkem multiplayerových her, a Unreal Engine 5 poskytuje robustní nástroje pro její implementaci, čímž umožňuje vývojářům snadno vytvářet stabilní a efektivní multiplayerové zážitky. Engine podporuje jak model server-klient, kde hlavní server spravuje herní svět a klienti se k němu připojují, tak model dedikovaných serverů, které běží samostatně a nemusí být hostovány aktivním hráčem. Tento model je vhodný pro hry s velkým počtem hráčů, kde je kladeno důraz na stabilitu a nízkou latenci.

4.5.1 Replikace

Jedním z hlavních mechanismů pro synchronizaci dat mezi klienty a serverem v Unreal Engine je replikace. Tento systém umožňuje definovat, které objekty a proměnné mají být synchronizovány mezi serverem a všemi připojenými klienty. Replikace je velmi důležitá pro zajištění konzistentního herního zážitku, protože se stará o to, že každý hráč vidí stejný stav světa, i když se ve skutečnosti nachází na různých počítačích. Například, pokud hráč vystřelí, replikace zajistí, že ostatní hráči na serveru budou vidět, že došlo k akci, i když to byla událost, která se stala na jiném počítači.

4.5.2 Serverové a klientské funkce v Unreal Engine

Ve vývoji multiplayerových her je klíčové správně rozlišovat mezi tím, co se vykonává na serveru, a co na klientovi. Toto rozdělení má zásadní vliv nejen na výkon a plynulost hry, ale především na její bezpečnost a konzistentní synchronizaci mezi hráči.

- **Server** – Má v síťové architektuře Unreal Engine plnou autoritu nad herním světem. Vykonává veškerou důležitou logiku – zpracovává pohyb postav, kolize, bojové mechaniky, správu herního stavu a další klíčové operace. Výsledky výpočtů serveru jsou považovány za směrodatné a jsou replikovány na klienty. Klienti tyto informace přijímají a přizpůsobují jim svůj lokální stav.

- **Klient** – Obstarává především vizuální a zvukovou prezentaci – například vykreslování efektů, přehrávání animací nebo zvuků. Lokálně také může provádět predikce některých akcí, jako je pohyb postavy, aby hra působila svižně a bez zpoždění. Při nesouladu mezi predikcí klienta a autoritativní odpovědí serveru dojde ke korekci – například efektu známému jako *rubberbanding*, kdy je hráčova pozice upravena podle skutečného stavu hry na serveru.

4.5.3 Remote Procedure Calls (RPCs)

Pro efektivní komunikaci mezi klientem a serverem Unreal Engine využívá tzv. Remote Procedure Calls (RPCs), což jsou speciální síťové funkce, které umožňují volání metod mezi zařízeními. RPCs zajišťují, že akce provedené na jednom zařízení mohou být synchronizovaně vykonány na jiných – například pokud hráč vystřelí, tato informace musí být doručena ostatním klientům i serveru. Existují tři hlavní typy RPC funkcí:

1. **Server RPC** – Funkce, která je vykonána na serveru. Používá se pro validaci vstupu, změny herního stavu nebo logiku, která nesmí být ponechána klientům.

2. **Client RPC** – Funkce, která je vykonána na konkrétním klientovi. Využívá se například pro spuštění animace nebo vizuální efekt, který se má zobrazit jen určitému hráči.

3. **Multicast RPC** – Volána na serveru, vykonána na všech připojených klientech. Slouží k synchronizaci událostí, které musí vidět všichni hráči – výbuchy, animace objektů, změny prostředí apod.

Důležitým atributem RPCs je také možnost označení funkce jako "Reliable" nebo "Unreliable", které se vztahují k metodě přenosu dat:

- **Reliable RPCs** – Zaručují, že volání bude doručeno na cílový stroj, i když dojde k problémům v síti. To je ideální pro akce, které jsou kritické pro správnou synchronizaci mezi klienty a serverem (například zásahy nebo důležité herní události). Nicméně, reliable RPCs mohou mít vyšší latenci, protože systém musí zajistit jejich doručení. Tento typ RPC se obvykle používá s TCP (Transmission Control Protocol), což je protokol zajišťující spolehlivý přenos dat s potvrzením o doručení, ale s vyšší latencí, protože čeká na potvrzení přijetí každého balíčku.
- **Unreliable RPCs** – Jsou rychlejší, ale nejsou garantovány. Měly by se používat pro neklíčové události, kde je důležitější výkon než 100% spolehlivost, například aktualizace polohy postavy nebo efektů, které neovlivní základní hratelnost, pokud se někdy neprojeví. Tento typ RPC se často používá s UDP (User Datagram Protocol), který je rychlý, ale nezajišťuje doručení dat. UDP je ideální pro aplikace, kde je kladeno důraz na nízkou latenci, například pro real-time herní mechaniky, kde ztráta některých dat není kritická.

4.5.4 Matchmaking a session management

Pro správu připojení hráčů do her a jejich organizaci nabízí Unreal Engine vestavěnou podporu pro matchmaking a session management. Tento systém umožňuje vývojářům snadno integrovat funkce pro vytváření herních relací, připojování hráčů a správu těchto relací. Unreal Engine podporuje integraci s externími službami pro správu online herních relací, jako jsou Steam nebo Epic Online Services (EOS). To hráčům umožňuje snadno najít a připojit se k dostupným serverům a nabízí pokročilé možnosti pro organizaci zápasů (například párování hráčů podle jejich dovedností).

Unreal Engine rovněž umožňuje snadno implementovat funkce jako lobby, přizpůsobení herních session parametrů (např. maximální počet hráčů, mapy, herní režimy), a také poskytuje nástroje pro správu připojení hráčů (například detekci odpojení nebo správa chyb při připojování).

4.5.5 Latence a optimalizace

Unreal Engine 5 rovněž poskytuje nástroje a systémy pro optimalizaci síťové komunikace a minimalizaci latence, což je klíčové pro dosažení plynulého a konzistentního herního zážitku v multiplayer prostředí. V rámci replikace herních objektů využívá engine techniku network relevancy, která zajišťuje, že se na klienta posílají pouze data relevantní jeho aktuální pozici nebo akci – tím se výrazně snižuje množství přenášených dat. Kromě toho Unreal umožňuje používat frequency throttling, tedy omezení frekvence zasílání aktualizací pro určité objekty, což opět pomáhá s optimalizací přenosu. Celý síťový model je navíc rozšiřitelný pomocí C++ a je možné přepsat základní replikace i routing systém, pokud vývojář potřebuje např. složitější prioritizaci nebo vlastní model synchronizace.

Pro minimalizaci viditelných dopadů latence jsou v enginu implementovány techniky jako client-side prediction (predikce pohybu na straně klienta), interpolation (plynulé dopočítání polohy mezi dvěma stavovými aktualizacemi), a také reconciliation, což je mechanismus pro korekci pozice objektu v případě nesrovnalosti mezi klientem a serverem. To se často využívá např. u pohybu postav – klient „předpoví“, kam se hráč hýbe, ale jakmile dorazí potvrzení ze serveru, případné odchylky se opraví, většinou nepostřehnutelně pro hráče. [21]

4.6 Integrace s databázemi / web API

Pro moderní multiplayerové hry je klíčové efektivní propojení s databázemi a webovými službami, například pro autentizaci hráčů nebo ukládání herních dat. Unreal Engine nabízí několik metod pro komunikaci s REST API a databázemi:

- **HTTP požadavky** – Unreal Engine podporuje odesílání HTTP požadavků, což umožňuje snadnou komunikaci s REST API. V C++ je možné použít třídu FHttpModule, FHttpRequestPtr a zpracovávat odpovědi v asynchronních callback funkcích.

- **WebSockets** – Pro real-time komunikaci lze využít WebSockets, které poskytují obousměrné spojení mezi klientem a serverem. Na rozdíl od klasických HTTP požadavků jsou WebSockety ideální pro přenos dat v reálném čase, například pro herní chat, notifikace nebo dynamické události. Unreal Engine má pro WebSockety připraven modul, který umožňuje snadné napojení klienta na server.

- **Databázové konektory** – Unreal Engine umožňuje přímé propojení s databázemi pomocí pluginů třetích stran. Například SQLite je podporován jako lokální úložiště, které je vhodné pro menší projekty nebo ukládání dat na klientovi. Pro přímé propojení s většími databázemi, jako je MySQL nebo PostgreSQL, je nutné využít specializované pluginy nebo middleware (např. použitím již zmíněných HTTP požadavků na vlastní REST API jako zprostředkovatele).

4.7 Struktura objektů v rámci UE

UE5 pracuje s objektově orientovanou architekturou, která je založena na několika základních třídách:

- **Actor** – základní třída pro všechny objekty ve světě hry.
- **Pawn** – speciální typ třídy Actor, který může být ovládán hráčem nebo AI.
- **Character** – rozšířená verze třídy Pawn obsahující pohybový systém s animacemi.
- **Controller** – objekt, který řídí chování třídy Pawn, rozdělený na PlayerController (pro hráče) a AIController (pro umělou inteligenci).
- **GameMode** – definuje pravidla hry, například jak hra začíná, kdy hra končí nebo hráč bude chtít uzavřít obchod s NPC. Funguje jako typický server.
- **GameInstance** – objekt přetrvávající mezi úrovněmi, který lze využít pro uchování globálních dat, například přihlášení hráče nebo nastavení hry.

4.8 Viditelnost objektů mezi sebou

V multiplayerových hrách se ne všechny objekty vidí navzájem. Každý hráč má vlastní instanci svého PlayerControlleru, což znamená, že PlayerController jednoho hráče nevidí PlayerController jiného hráče. Tento design zajišťuje, že každý klient spravuje pouze svůj vlastní vstup a interakci s herním světem, zatímco server řídí celkový stav hry.

Naopak GameMode běží pouze na serveru a vidí všechny objekty ve hře, včetně všech hráčských instancí, AI a jejich příslušným Controllerům. Díky tomu je GameMode často používán k řízení pravidel hry, například počítání skóre, správu časovačů nebo vyhodnocení podmínek pro vítězství.

4.9 Detailnější příklady her vytvořených v UE

Unreal Engine je široce využívaný herní engine, který umožňuje vývoj her napříč různými žánry – od stylizovaných akčních střílečeк až po realistické RPG s pokročilou grafikou. Díky své flexibilitě a výkonným nástrojům, jako jsou Lumen, Nanite a pokročilé fyzikální simulace, se stal oblíbeným mezi vývojáři po celém světě. Mezi známé tituly, které využívají Unreal Engine, patří:

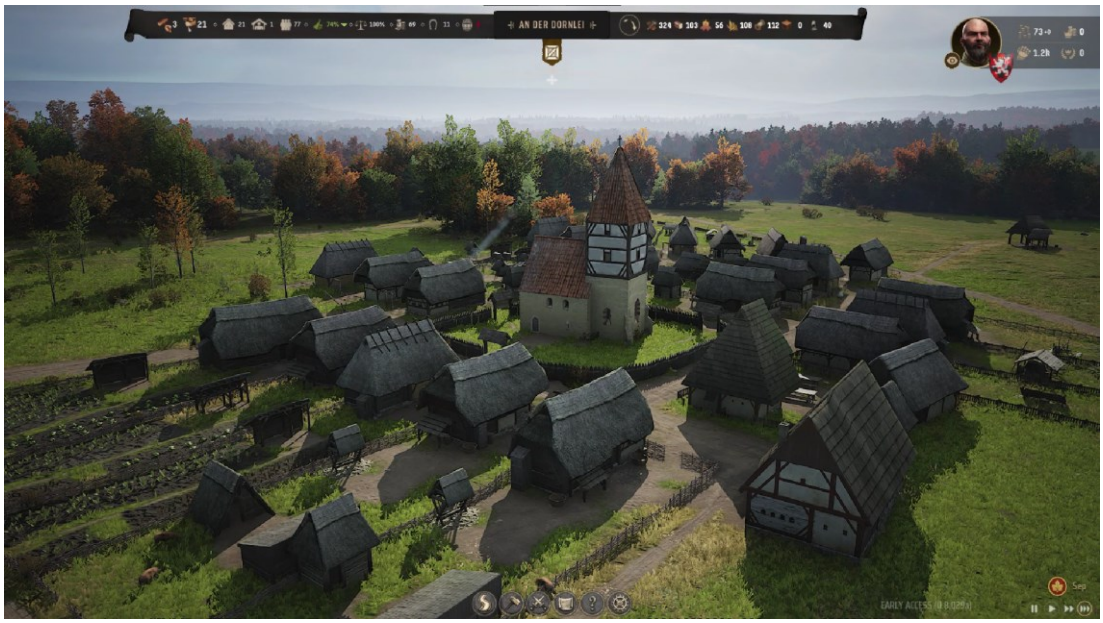
Black Myth: Wukong (2024, UE5, Akční RPG, Soulslike) – Jeden z moderních titulů využívajících Unreal Engine 5 a jeho nejpokročilejší technologie, jako je Lumen pro realistické globální osvětlení a Nanite pro detailní prostředí. Hra nabízí vizuálně působivé souboje, plynulé animace a vysoký stupeň realismu, který posouvá hranice herní grafiky.

Obrázek 10 – Screenshot ze hry Black Myth: Wukong (zdroj vlastní)

S.T.A.L.K.E.R. 2: Heart of Chornobyl (2024, UE5, FPS, Survival Horror) – Pokračování této série přináší otevřený svět, intenzivní atmosféru a dynamický ekosystém s propracovanou umělou inteligencí. Unreal Engine 5 umožňuje realistické nasvícení, detailní prostředí a pokročilé fyzikální simulace, čímž vytváří pohlcující zážitek.

Obrázek 11 – Screenshot ze hry S.T.A.L.K.E.R. 2: Heart of Chornobyl (zdroj vlastní)

Manor Lords (2024, UE5, Strategická hra, City Builder) – Realistická středověká budovatelská strategie kombinující správu města s taktickými bitvami. Díky Unreal Enginu dosahuje hra vizuální autenticity s propracovanými animacemi, dynamickým počasím a detailním zpracováním středověkých osad.



Obrázek 12 – Screenshot ze hry Manor Lords (zdroj vlastní)

Borderlands 3 (2019, UE4, Looter Shooter, Co-op FPS) – Akční střílečka kombinující dynamickou hratelnost s charakteristickou cel-shading grafikou. Unreal Engine zde zajišťuje plynulý průběh akce, optimalizaci rozsáhlých prostředí a propracované vizuální efekty.

Obrázek 13 – Screenshot ze hry Borderlands 3 (zdroj vlastní)

Life is Strange (2015, UE3, Interaktivní příběhová adventura) – Příběhově zaměřená hra kladoucí důraz na rozhodnutí hráče, která ovlivňují vývoj děje. Unreal Engine zde umožňuje filmovou atmosféru s detailním nasvícením a plynulými animacemi postav, které podporují pohlcující narativní zážitek.

Obrázek 14 – Screenshot ze hry Life is Strange (zdroj vlastní)

Schválně jsem zvolil různé žánry her, abych ukázal, jak všestranný je Unreal Engine a jak široké má uplatnění napříč nejrůznějšími typy her – od akčních RPG až po simulátory a strategické tituly.

5 Návrh hry

5.1 Seznam požadavků

1 Požadavky na architekturu a technologii

   a. Použití Unreal Engine 5 a programování v jazyce C++.

   b. Architektura klient-server

   c. Dynamické vytváření více instancí herních serverů

   d. Modularita systému – snadné přidání nových výprav, nepřátel

2 Požadavky na hratelnost a herní mechaniky

   a. Výběr specializace postavy ovlivňující rozvoj postavy

   b. Tvorba party hráčů ve městě a následné vytvoření nové instance

   c. Souboje s nepřáteli, sbírání zkušeností a materiálů

   d. Náhodné generování událostí v instanci

   e. Systém výroby

3 Požadavky na správu dat a bezpečnost

   a. Ukládání hráčských dat (zkušenosti, atributy) do databáze

   b. Ověřování přihlašovacích údajů a ochrana proti neoprávněnému přístupu

   c. Validace všech důležitých změn dat na serverové straně (nejen na klientovi)

   d. Automatické odpojení hráče při detekci podezřelého chování

4 Požadavky na grafiku a uživatelské rozhraní

   a. Jednotný grafický styl prostředí a postav (low-poly styl)

   b. Minimální set animací pro hráče i nepřátele

   c. Jednoduché uživatelské rozhraní pro správu postavy

   d. Zobrazování důležitých herních stavů (zdraví, zkušenosti).

   e. Notifikace (např. důvody selhání akce / potvrzení o úspěšném vykonání akce).

5.2 Jádro hry

Hra začíná na přihlašovací obrazovce, kde má uživatel možnost registrace nebo přihlášení. Na pozadí v tomto kroku běží autentizační server, který zajišťuje ověření přihlašovacích údajů. Po úspěšném přihlášení je hráč přepojen do herního lobby, které představuje centrální prostor mezi jednotlivými výpravami.

V lobby se hráč může volně pohybovat a interagovat s různými objekty. Například u kovadliny může vylepšovat své atributy a u nástěnky spustit výpravu. Lobby také umožňuje pozvání dalších hráčů do skupiny (party), aby mohli společně absolvovat výpravy.

Při spuštění výpravy je hráč (spolu s celou party) přenesen na dynamicky vytvořený server, kde se načte mapa výpravy. Terén výpravy je vždy předem definovaný, ale v herním světě se při každém spuštění generují náhodné události, které mohou průběh výpravy ovlivnit. Výprava dále obsahuje nepřátele, jejichž rozmístění je předem určeno a poražení všech těchto nepřátel je hlavním cílem mise. Za každého poraženého nepřítele získává hráč zkušenosti a zlaťáky. Pokud během výpravy zemře nebo ji opustí před jejím dokončením, přichází o všechny získané odměny. Naopak, pokud výpravu úspěšně dokončí, obdrží navíc i náhodný počet materiálů, které může později využít. Po návratu do lobby může hráč pokračovat v dalším cyklu hry.

5.3 Systém boje a interakce s nepřáteli / NPC

Bojový systém je založen na reálném čase a využívá základních RPG principů, jako je cooldown mechanika mezi útoky nebo atributy, které určují výkon v boji (např. síla, obratnost). Nepřátelé jsou řízeni jednoduchou umělou inteligencí (AI), která se náhodně pohybuje po mapě. Pokud se hráč přiblíží, nepřítel začne po něm útočit. Nepřítel vždy útočí po prvním hráči, který vstoupí do jeho aggro hitboxu.

Všechny útoky jsou validovány na serveru, včetně generování hodnot poškození a provedení dalších akcí. To znamená, že jakékoli interakce v boji jsou ověřeny na serverové straně, aby se předešlo podvodům nebo desynchronizaci mezi hráči. Například vzdálenost mezi hráčem a nepřítelem je pravidelně kontrolována, aby útoky probíhaly korektně. Po každém úspěšném zásahu se na serveru vypočítá poškození na základě atributů hráče, a příslušná hodnota se hráči zobrazí na UI.

NPC (Non Playable Characters) jsou v aktuálním stavu hry reprezentovány pouze objekty jako kovadlina a nástěnka, které mají interaktivní funkce. Ačkoli tyto objekty technicky nesplňují klasickou definici NPC (jelikož to nejsou charaktery), jejich funkčnost odpovídá názvu NPC ve smyslu poskytování užitečných interakcí pro hráče.

Pro splnění definice NPC by bylo do budoucna možné přidat například postavu kováře, který by obsluhoval hráče u kovadliny. Dále například obchodníka, který by umožnil hráčům nakupovat a prodávat předměty. Tento prvek by podpořil dynamiku hry a rozšířil možnosti interakce s herním světem.

5.4 Systém atributů a specializace

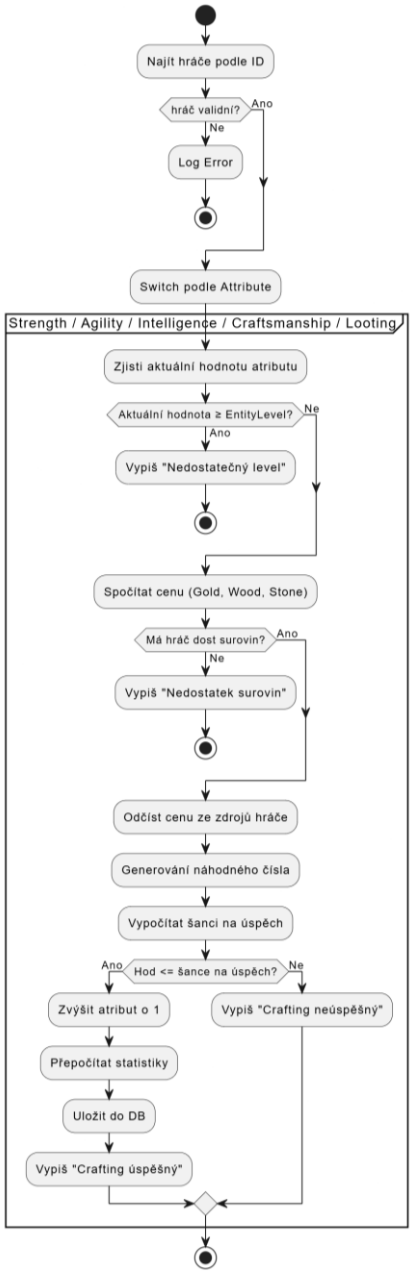
Při registraci si hráč vybírá specializace ve dvou atributech, a naopak slabinu ve dvou dalších atributech. V aktuální implementaci tento systém nemá zásadní vliv na herní zážitek, jelikož se specializace projevuje pouze přidáním +1 nebo –1 k hodnotám vybraných atributů.

V současnosti je ve hře implementováno pět základních atributů, které mají vliv na různé aspekty hratelnosti:

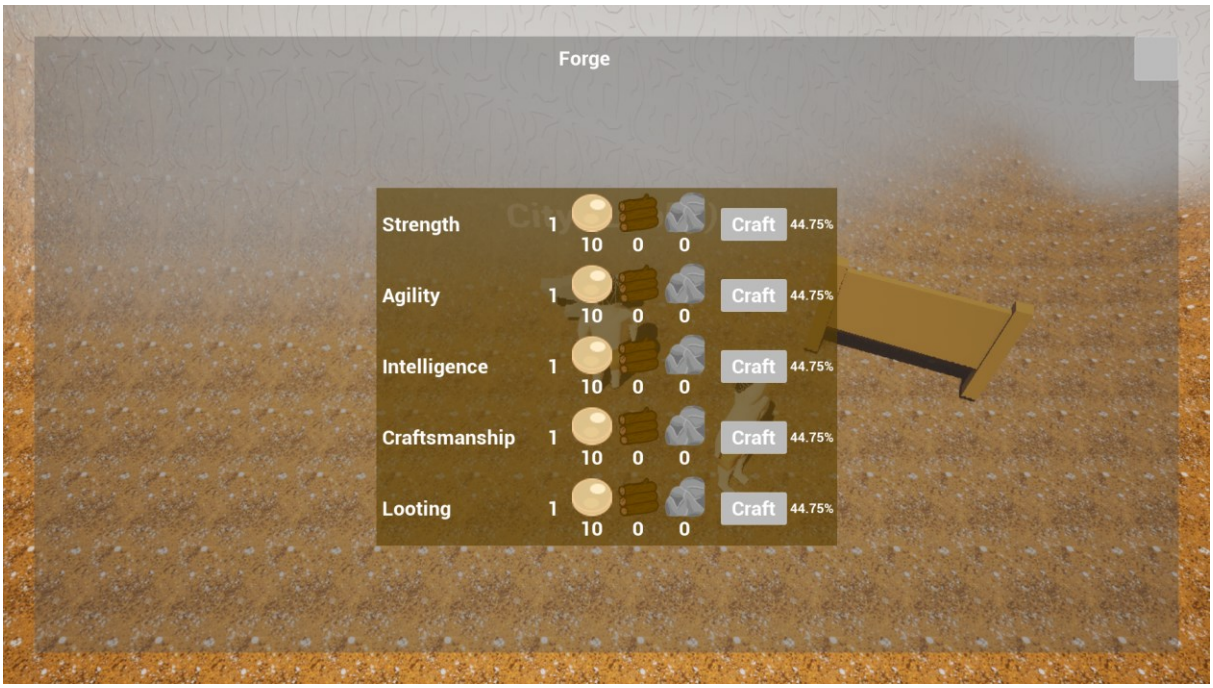
- **Strength (Síla)** – Určuje maximální počet životů hráče a zvyšuje fyzickou sílu útoků. Hráči s vysokou sílou mohou způsobit větší poškození v boji zblízka a vydržet více zásahů od nepřátel.
- **Agility (Obratnost)** – Ovlivňuje frekvenci útoků. Vyšší obratnost znamená kratší prodlevu mezi údery, což umožňuje rychlejší a plynulejší provádění akcí v boji. Hráči s vysokou obratností mohou provádět více útoků za jednotku času, což je užitečné pro rychlé útoky na nepřátele.
- **Intelligence (Inteligence)** – Zvyšuje množství získaných zkušeností za porážení nepřátel. Postavy s vysokou inteligencí mají výhodu při rychlém postupu hrou, protože mohou rychleji dosahovat vyšších úrovní a tím zvyšovat svou herní sílu.
- **Looting (Sběr kořisti)** – Ovlivňuje množství odměn získaných při zabití nepřítele, otevření truhel nebo dokončení výpravy. Vyšší hodnota znamená lepší kořist, což je užitečné pro hráče, kteří se zaměřují na sběr surovin.
- **Craftsmanship (Řemeslné dovednosti)** – Zvyšuje šanci na úspěšnou výrobu. Tento atribut je důležitý pro hráče, kteří se chtějí soustředit na vylepšování s vyšší efektivitou.

5.5 Výroba a suroviny

Ve hře je momentálně jediný způsob, jak vylepšit svou postavu, formou výroby. Pro výrobu je potřeba určité množství zlaťáků a na vyšších úrovních i surovin. Výroba není zaručená – základní šance na úspěch je 50 %, která se dále upravuje na základě hodnoty rozvíjeného atributu a atributu Craftsmanship. Čím vyšší je hodnota rozvíjeného atributu (např. Intelligence nebo Agility), tím větší je negativní vliv na šanci na úspěch. Naopak čím vyšší je hodnota atributu Craftsmanship, tím větší bonus je přidělen k úspěchu. Atributy je možné rozvíjet maximálně do hodnoty úrovně hráče – například hráč s úrovní 5 může mít atribut Looting pouze na úrovni 5.



Obrázek 15 – UML diagram výroby (zdroj vlastní)



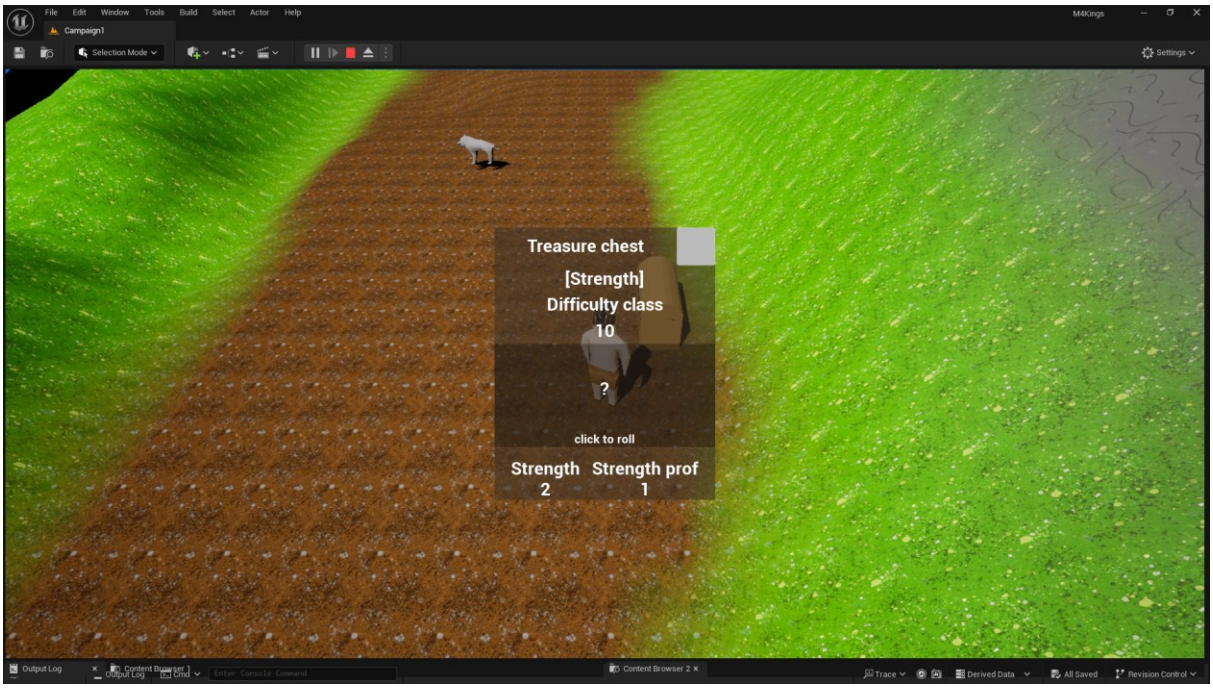
Obrázek 16 – Screenshot okna výroby (zdroj vlastní)

5.6 Náhodné události

Mechanika náhodných událostí vnáší do hry prvek nepředvídatelnosti, což nutí hráče přemýšlet o strategických rozhodnutích a přizpůsobit své chování aktuální situaci. Během výpravy se hráči mohou setkat s různými událostmi, jejichž výsledek závisí na jejich schopnostech a specializaci. Například zamčená truhla může být pro hráče s vysokou Strength snadným úkolem, zatímco pro hráče, kteří se specializují na Looting, je odměna větší a hodnotnější. Tento prvek umožňuje hráčům využít své atributy k dosažení výhod nebo k vyhnutí se penalizacím.

V současné implementaci existuje pouze jedna událost – zamčená truhla. Pokud hráč neuspěje při jejím otevření, může ji nenávratně poškodit a ztratit tak odměnu. Události jsou zatím volitelné a hráči se mohou rozhodnout, zda je využijí, aniž by to mělo trvalé negativní důsledky. Tento mechanismus přidává do hry příležitosti k získání dodatečných odměn, ale hráči se mohou rozhodnout je ignorovat.

Pokud je hráč v partě, odemčení truhly je zpřístupněno pro všechny členy skupiny. Odměna, kterou truhla obsahuje, se však odvíjí od hodnoty atributu Looting hráče, který truhlu otevřel. Hráči s vyšší hodnotou tohoto atributu získají lepší odměny pro celou skupinu. Pokud však hráč neuspěje v odemčení truhly, poškodí ji, čímž znemožní jejímu otevření komukoli dalšímu. Tento prvek podporuje týmovou spolupráci a nutí hráče přemýšlet, kdo by měl jaké akce provádět, aby maximalizovali zisky celé party.



Obrázek 17 – Screenshot okna náhodné události (zdroj vlastní)

5.7 Party systém

Party systém je klíčovým prvkem hry, jelikož se jedná o kooperativní titul zaměřený na týmovou spolupráci. Tento systém umožňuje hráčům spojit se do skupiny a společně se vydat na výpravu, čímž se otevírají možnosti pro kombinaci různých specializací a strategií.

Party se vytváří v hlavní lobby, kde hráči mohou pozvat ostatní nebo se připojit k již existující skupině. Jakmile je party sestavena, hráči společně vstupují do výpravy, kde sdílejí některé herní mechaniky – například při boji mohou využívat synergii svých specializací, sdílet suroviny nebo si vzájemně pomáhat při řešení náhodných událostí.

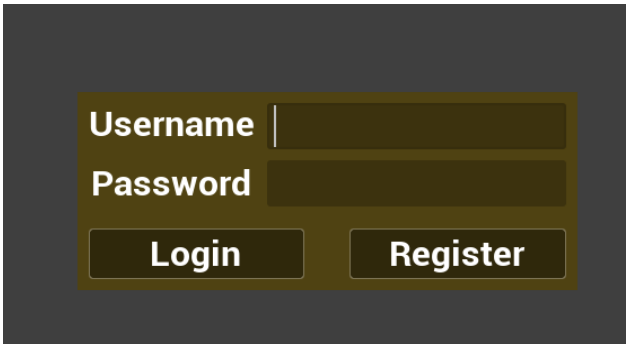
Některé herní mechaniky dynamicky reagují na přítomnost party. Například náhodné eventy se vyhodnocují na základě atributů jednotlivých členů party, a jejich úspěšnost může ovlivnit celou skupinu. Tento systém podporuje týmovou hratelnost a zajišťuje, že hráči nejsou na výpravě odkázáni jen sami na sebe, ale mohou si vzájemně pomáhat a společně překonávat výzvy.

6 Struktura hry

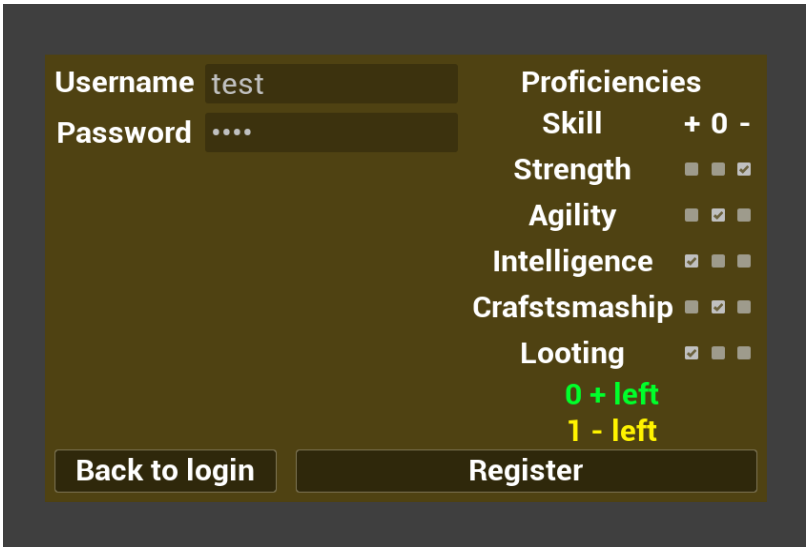
Hra je rozdělena do třech hlavních částí (login, lobby, instance výpravy), které společně tvoří její základní strukturu. Každá z těchto částí má svou specifickou roli a funkcionalitu, přičemž mezi nimi dochází k plynulému přechodu.

6.1 Login

Login je klíčovým prvním krokem při vstupu do hry. V této fázi se hráč buď registruje, nebo přihlašuje prostřednictvím svého účtu. Proces začíná tím, že uživatel zadá své přihlašovací údaje (uživatelské jméno a heslo). Tyto údaje jsou odeslány na server a následně na REST API, kde se provádí ověření identity hráče. Pokud jsou údaje správné, server vygeneruje session token, který je následně zaslán zpět klientovi. Tento token slouží jako bezpečný prostředek pro ověření uživatele v dalších interakcích s hrou a servery. Tento token je platný pouze dokud hráč nevypne hru a před jakýmkoli požadavkem, který manipuluje s daty musí být předložen k ověření. Pokud je token platný, hráč je považován za autentizovaného a je přesměrován do hlavního lobby, kde může pokračovat v dalším průběhu hry.



Obrázek 18 – Přihlašovací okno (zdroj vlastní)



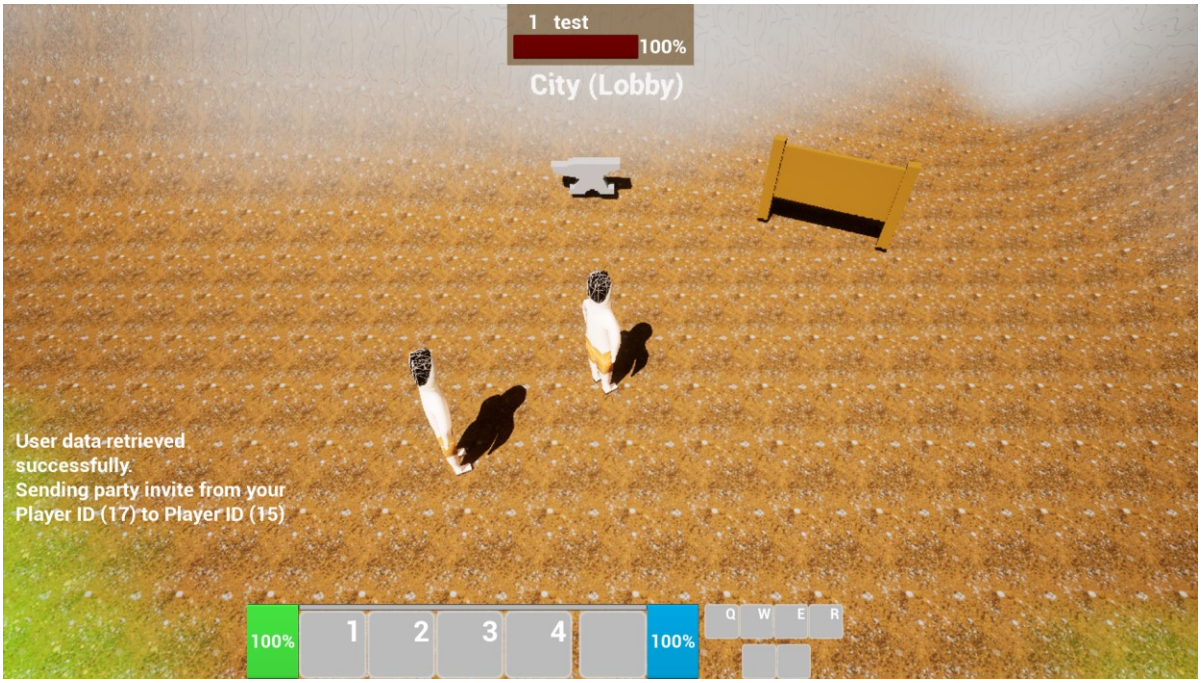
Obrázek 19 – Registrační okno (zdroj vlastní)

6.2 Lobby

Lobby slouží jako centrální rozbočovací bod hry. Hráči zde mohou:

- Vytvářet a připojovat se do party
- Přistupovat k crafting systému
- Vybrat a vstoupit do instance výpravy

V této části hry se hráči mohou setkávat a připravovat se na nadcházející výzvy.



Obrázek 20 – Screenshot lobby (zdroj vlastní)

6.3 Instance levelů

Instance levelu představuje samotnou výpravu, do které hráči vstupují buď sami, nebo jako součást party. Jedná se o uzavřený herní svět s omezenou dobou trvání (po opuštění všemi hráči se smaže), ve kterém hráči bojují s nepřáteli, získávají suroviny a řeší náhodné události. Každá instance je generována dynamicky na základě předem definovaných parametrů. To znamená, že zatímco rozložení mapy a základní struktura zůstávají stejné, konkrétní náhodné události, které hráči potkají, se generují náhodně. Tento systém zajišťuje mírnou variabilitu průběhu jednotlivých misí a zvyšuje znovuhratelnost.

Hlavní prvky instance:

- **Soubojový systém** – hráči čelí nepřátelům
- **Systém odměn** – hráči získávají suroviny a zlaťáky podle svého výkonu a rozhodnutí během instance
- **Náhodné události** – interaktivní prvky, které mohou hráčům usnadnit nebo zkomplikovat postup instancí

Po dokončení instance se hráči vracejí zpět do lobby, kde si mohou spravovat získané suroviny, zlepšovat svoji postavu prostřednictvím výroby a připravit se na další výpravu.

Obrázek 21 – Screenshot výpravy (zdroj vlastní)

6.4 Ukládání a získávání dat

Hra pracuje s perzistentními daty, která jsou ukládána a načítána prostřednictvím databázového systému. Tento systém zajišťuje, že hráčský postup je vždy bezpečně uložen a dostupný při dalším přihlášení.

Mezi hlavní ukládané informace patří:

- **Herní účet a přihlašovací údaje** – jméno postavy (a zároveň účtu) a argon2 otisk hesla
- **Statistiky postavy** – úroveň, zkušenosti, atributy a specializace
- **Inventář** – zlaťáky a suroviny

Data jsou uchovávána na serveru a v databázi, přičemž veškerá komunikace mezi serverem a databází probíhá prostřednictvím zabezpečených REST API požadavků. Tím je zajištěna konzistence a bezpečnost uložených dat, čímž se předchází ztrátě postupu nebo neoprávněným úpravám hráčských statistik.

6.5 Architektura serveru

Serverová část hry je rozdělena do několika komponent, které spolu vzájemně komunikují, aby zajistily plynulý a bezpečný průběh hry. Hlavní komponenty serveru jsou:

- **Auth Server** – Tento server je zodpovědný za autentizaci hráčů. Při připojení se hráč přihlásí pomocí svých přihlašovacích údajů, které jsou ověřeny na serveru. Po úspěšném přihlášení je hráči vygenerován session token, který je následně použit k ověření identity hráče při dalších operacích ve hře. Implementován v Unreal Engine čistě v C++.

- **Lobby Server** – Lobby server slouží jako centrální bod pro správu herních session a interakce mezi hráči. Hráči se zde mohou spojit do skupin, spravovat své postavy a připravovat se na mise. Tento server zajišťuje synchronizaci a správu dat mezi všemi hráči v lobby. Implementován v Unreal Engine čistě v C++.

- **Instance Servery** – Instance servery jsou zodpovědné za hostování dynamických instancí výprav. Každý hráč, nebo skupina hráčů, je připojena k odpovídající instanci, kde probíhá samotná hra. Servery pro instance jsou dynamicky vytvářeny při zahájení nové výpravy a po jejím ukončení jsou ukončeny, čímž se šetří prostředky serveru. Implementován v Unreal Engine, C++ kód je převzatý z Lobby serveru, ale je obohacen o některé funkce v Blueprints.

- **Database Server** – Tento server zajišťuje uložení perzistentních herních dat, jako jsou přihlašovací údaje hráčů, statistiky postav a související herní data. Komunikace s databází probíhá prostřednictvím REST API implementovaného v C# ASP.NET Core, který zajišťuje bezpečnou a efektivní výměnu dat mezi serverem a databází.

6.6 Použité technologie a nástroje

- **Docker** – Použit pro zajištění snadné a efektivní správy serverových komponent a jejich izolaci do kontejnerů. Databáze a REST API běží v samostatném Docker kontejneru, což zjednodušuje nasazení a správu serverů.

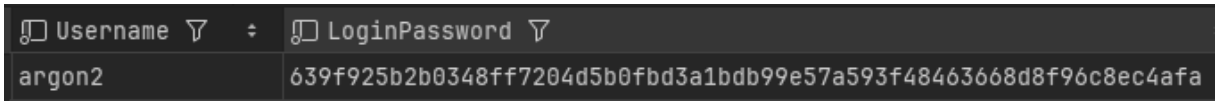
- **MySQL** – Databáze, která uchovává perzistentní herní data. MySQL je spolehlivá a výkonná relační databáze, která umožňuje efektivní uložení a načítání dat hráčů, jejich postav a dalších informací.
- **ASP.NET Core + Entity Framework** – Použito pro vývoj REST API, které komunikuje mezi serverovými komponentami a databází. Entity Framework slouží k jednoduchému mapování objektů v aplikaci na relační databázi a poskytuje moderní nástroje pro práci s databázovými daty.
- **Unreal Engine + Blueprints** – Unreal Engine je použit jako hlavní nástroj pro vývoj herní logiky, vizuálních efektů a mechanik, přičemž jazyk C++ slouží jako základ pro implementaci klíčových systémů, jako je síťová komunikace, správa dat, bojový systém, replikace nebo autentizace, kde je potřeba výkon a plná kontrola nad chováním. Blueprints naopak využívám pro práci s uživatelským rozhraním a vizuální prezentaci herních mechanik, jako jsou animace, widgety nebo jednoduché interakce, které vyžadují rychlou iteraci bez nutnosti zdlouhavé kompilace – což výrazně zrychluje vývoj a ladění.
- **Visual Studio + JetBrains Rider** – Přešel jsem z Visual Studia na JetBrains Rider kvůli otřesnému indexování a špatnému našeptávání kódu ve Visual Studiu, což výrazně zpomalovalo vývoj. Rider nabízí mnohem lepší výkon a stabilitu při práci s Unreal Engine. Navíc má zabudovanou podporu pro Unreal Engine, což znamená, že při každém spuštění si naindexuje celý engine a tím pádem je vývoj rychlejší a efektivnější, protože následující práce s kódem je plynulá a bez zbytečných prodlev.
- **Blender** – Blender byl použit pro tvorbu 3D modelů, animací a jednoduchých textur. Tento nástroj, který je dostupný zdarma, je velmi silný a hojně používaný v oblasti modelování a animace.
- **Photoshop** – Photoshop byl využit pro tvorbu uživatelského rozhraní (UI). Vytvářel jsem v něm grafické prvky (převážně ikony), které tvoří vzhled hry.
- **Git + GitHub** – Git a GitHub byly použity pro verzování a zálohování kódu, což je nezbytné pro správu velkých projektů. Git umožňuje efektivní sledování změn v kódu, práci s větvemi a návrat k předchozím verzím, pokud dojde k chybám. GitHub slouží jako cloudová platforma pro ukládání a sdílení kódu, což usnadňuje zálohování celého projektu.

7 Implementace

7.1 Registrace a přihlášení

Autentizace uživatelů přes REST API funguje velmi přímočaře.

1. Po spuštění hry se hráči zobrazí přihlašovací nebo registrační okno, do kterého zadá své údaje.
2. Po odeslání klient nejprve ověří, že pole nejsou prázdná, odstraní přebytečné mezery a odešle požadavek na server.
3. Server přijatá data znovu očistí a ověří jejich validitu.
4. Heslo je poté osoleno SHA1 otiskem uživatelského jména a zahashováno algoritmem Argon2 a spolu s uživatelským jménem odesláno na REST API. [22]
5. REST API tato data rovněž zkontroluje a provede SQL dotaz do databáze, kde hledá záznam s odpovídajícím uživatelským jménem a otiskem hesla.
6. Pokud se žádný záznam nenajde, REST API vrátí HTTP kód 404, čímž klient obdrží zprávu o neplatných údajích.
7. Pokud je uživatel nalezen, REST API odpoví HTTP kódem 200.
8. Server následně vygeneruje náhodný session token, který kontroluje proti databázi session tokenů (dokud nevygeneruje unikátní).
9. Tento token se pak odešle klientovi, který si ho uloží do instance třídy GameInstance – což je objekt Unreal Engine, který uchovává data napříč úrovněmi a servery.
10. Jakmile je token uložen, klient odešle požadavek na přesun na lobby server, přičemž token připojí k požadavku. Pokud token není validní nebo chybí, server to vyhodnotí jako bezpečnostní hrozbu a okamžitě ukončí klientskou instanci hry.



Username	LoginPassword
argon2	639f925b2b0348ff7204d5b0fbd3a1bdb99e57a593f48463668d8f96c8ec4afa

Obrázek 22 – Ukázka otisku hesla (zdroj vlastní)

```
[HttpPost("LoginUser")]
public async Task<IActionResult> LoginUser([FromBody] LoginRequest request)
{
    try
    {
        if (string.IsNullOrEmpty(request.Username) || string.IsNullOrEmpty(request.Password))
            return StatusCode(400, new { ServerMessage = "Invalid request data", ClientMessage = "Please provide valid credentials." });

        var user = await _context.Players
                                 .AsNoTracking()
                                 .FirstOrDefaultAsync(u => u.Username == request.Username && u.LoginPassword == request.Password);
        return user == null ? StatusCode(400, new { ServerMessage = "Invalid username or password", ClientMessage = "Login failed. Please check your credentials." })
                            : StatusCode(200, new { ServerMessage = "Login successful", ClientMessage = "Welcome back!", user.PlayerId });
    }
    catch (Exception e)
    {
        return StatusCode(500, new { ServerMessage = e.Message, ClientMessage = "An unexpected error occurred. Please try again later." });
    }
}
```

Obrázek 23 – REST API zdrojový kód pro přihlášení (zdroj vlastní)

7.2 Správa relace

Session token je v aktuální implementaci využíván jako náhrada za klasický identifikátor hráče – místo ukládání a předávání ID hráče slouží token jako klíč pro přístup k jeho údajům v databázi. Při každém požadavku klienta, který vyžaduje práci s daty (např. načtení postavy, úprava inventáře), se token odešle na REST API, které na základě něj vyhledá příslušného hráče. Tento přístup zvyšuje bezpečnost tím, že na klientské straně nikdy nefiguruje skutečné ID hráče, čímž se snižuje riziko neoprávněných zásahů do dat.

V aktuální verzi nejsou k session tokenu přiřazena žádná metadata – například čas vytvoření, expirace nebo IP adresa zařízení. I když systém takto funguje, absence těchto údajů snižuje možnosti pro správu životního cyklu relace, detekci podezřelého chování nebo automatické odhlašování po delší době nečinnosti. Do budoucna by bylo vhodné metadata k session tokenům přidat, a tím zvýšit bezpečnost i flexibilitu celého systému.

7.3 Lobby systém

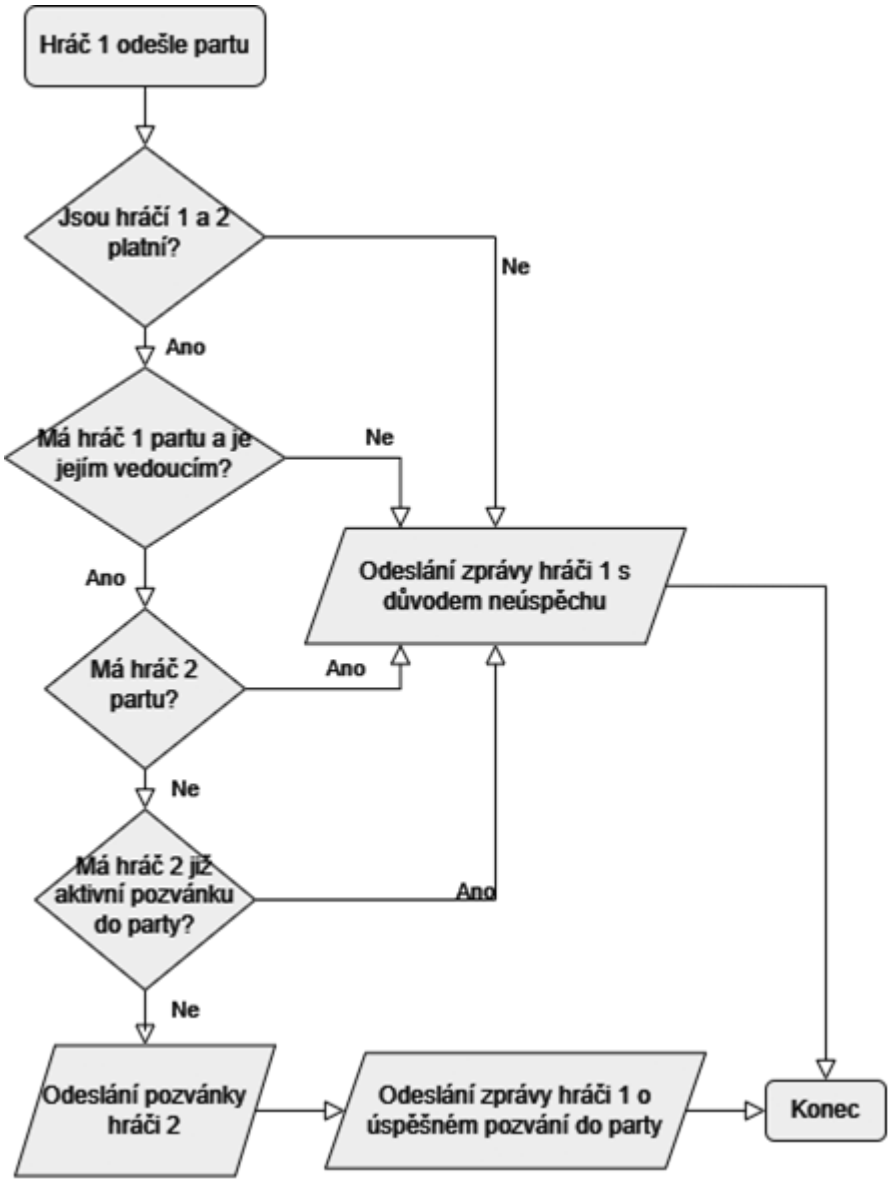
7.3.1 Správa hráčů

V lobby není veden žádný centrální seznam hráčů na úrovni herní logiky – připojení nového hráče je řešeno pomocí replikace v rámci Unreal Engine. Když se hráč připojí na server (lobby nebo instance výpravy), jeho data (např. jméno, životy, atributy) se automaticky replikují na ostatní klienty, což zajišťuje, že každý hráč vidí ostatní připojené hráče a jejich stavy bez potřeby ruční synchronizace.

7.3.2 Vytváření party

Hráči mohou vytvářet party, do kterých mohou pozvat ostatní hráče z lobby. Každá party má své unikátní ID a je spojena s databázovou strukturou, která uchovává členy party a identifikaci vůdce (leader). Vůdce party má plná práva k správě členů, což zahrnuje pozvání nových hráčů nebo jejich vyhození z party. Tento mechanismus je implementován pomocí REST API, které provádí operace v databázi, kde jsou uchovávány vztahy mezi členy party a jejich role.

Pozvání do party je implementováno pomocí herní interakce – hráč, který chce někoho pozvat, najede kurzorem na hráče v herním světě a následně použije pravé tlačítko myši. Tento proces spustí žádost, která je zaslána na server, jenž ověří, zda je hráč oprávněn k pozvání (např. zda není již v jiné partě). Po ověření je pozvání odesláno cílovému hráči, který může rozhodnout, zda pozvánku přijme.



Obrázek 24 – Vývojový diagram pozvání do party (zdroj vlastní)

Každá party je izolována od ostatních, což znamená, že hráči mohou být maximálně právě v jedné partě. Tato izolace zajišťuje, že každá party může mít své vlastní aktivity, aniž by došlo k interferencím mezi různými skupinami. Když hráč přijme pozvánku, je přidán k odpovídající party a související informace o něm jsou aktualizovány v databázi.

Hráč může kdykoliv partu opustit. Tento proces je opět spravován přes server, který aktualizuje databázi a vyřazuje hráče z příslušné party. Pokud je hráč vůdcem party, systém automaticky přiřadí nového leadera, nebo parta zanikne, pokud není k dispozici jiný hráč. Tento systém umožňuje flexibilní správu hráčů a party v reálném čase, přičemž se všechny změny synchronizují mezi serverem a klienty pomocí replikace.

7.4 Instance levelu (výprava)

V lobby má hráč možnost spustit výpravu. Po odeslání požadavku klientem na vytvoření instance dojde ke spuštění nové instance výpravy na novém serveru, který se následně postará o správu dané výpravy. Nejprve lobby server ověří platnost session tokenu, který je odeslán z klienta, a následně generuje náhodný port pro novou herní instanci v určitém rozsahu. Tento port je generován, dokud není nalezený neobsazený port. Jakmile je port úspěšně vygenerován, server spustí novou instanci hry s mapou dané výpravy.

Jakmile je server připraven, hráč (spolu s případnou partou) je přesměrován na tento server. Při připojení na server dochází k inicializaci hráče a případně celé party. V tomto kroku se na pozadí načítají herní objekty a textury, které jsou replikovány mezi serverem a klientem. Zároveň se z databáze načítá objekt hráče, což zahrnuje jeho atributy, zkušenosti a další herní data.

V rámci výpravy se nachází několik nepřátel a náhodně generovaných událostí, které mohou ovlivnit průběh výpravy. Po zneškodnění všech nepřátel hráč může opustit výpravu, což iniciuje požadavek na návrat do lobby. Tento požadavek server zpracuje a před tím, než hráče vrátí zpět, ověří, zda je výprava dokončena. Pokud jsou všechny nepřátelé poraženi, server uloží nové hodnoty do databáze, včetně získaných zkušeností, zlaťáků a materiálů.

Pokud výprava není dokončena (když některý z nepřátel zůstává naživu), hráč přijde o veškeré získané věci. To samé platí i v případě, že hráč zemře – pokud dojde k smrti, jakékoli získané zkušenosti nebo materiály jsou ztraceny. Tento systém funguje na základě dočasného ukládání herních dat do paměti serveru. Pouze pokud hráč splní všechny podmínky úspěchu výpravy, jsou tato data následně zapsána do databáze a využita ve zbytku hry.

7.4.1 Náhodné události

Systém náhodných událostí slouží ke zvýšení variability a znovuhratelnosti výprav. Tyto události jsou generovány automaticky při spuštění serveru. Ve světě jsou předem rozmístěny instance třídy typu Actor, které mají definovaný tag EventSpawnLocation. Tyto instance slouží jako možné pozice, na kterých se mohou náhodné události objevit.

Po spuštění serveru je vygenerován náhodný počet událostí – v aktuální implementaci se počet pohybuje v rozsahu 0–2. Následně jsou náhodně vybrány příslušné lokace z dostupných EventSpawnLocation a na každé z nich je vytvořena instance konkrétní události.

V současnosti je implementována pouze truhla, která demonstruje fungování celého systému – od generování a umístění události až po interakci hráčů a zpracování výsledku na serveru.

Při interakci hráče s událostí se na klientovi zobrazí grafické rozhraní, které simuluje hod dvacetistěnnou kostkou (D20). Tato simulace funguje na principu generování náhodného čísla v rozmezí 1–20 každou půl sekundu a zobrazení dané hodnoty na UI. Celý vizuální efekt trvá minimálně 3 sekundy + dobu odezvy odeslání požadavku na server a získání odpovědi, tedy celková doba může být mírně proměnlivá v závislosti na aktuální síťové latenci.

Současně s animací na pozadí klient odešle požadavek na server pro vygenerování skutečné hodnoty hodu. Tato hodnota je určena výhradně na serveru, aby se předešlo pokusům o podvodné ovlivnění výsledku na straně klienta. Server tak zajišťuje férovost a důvěryhodnost výsledku každé události.

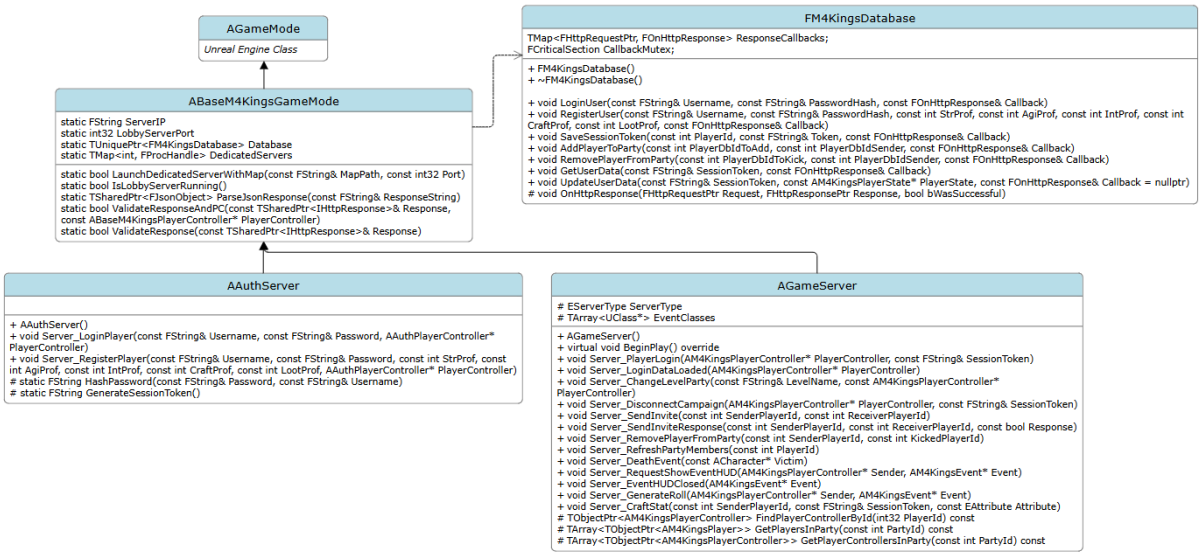
Po uplynutí 3 sekund je výsledek odeslán zpět na klienta. Následně je pomocí multicast RPC spuštěna funkce vyhodnocující výsledek – např. v případě úspěšného otevření truhly obdrží všichni hráči v instanci odměnu ve formě herních surovin a je spuštěna odpovídající animace.

Tento systém kombinuje vizuální efekty a plynulý uživatelský zážitek s technickým zabezpečením pomocí serverově řízené logiky, čímž zajišťuje konzistenci, bezpečnost a férovost v rámci multiplayerového prostředí.

7.5 Diagramy vybraných tříd

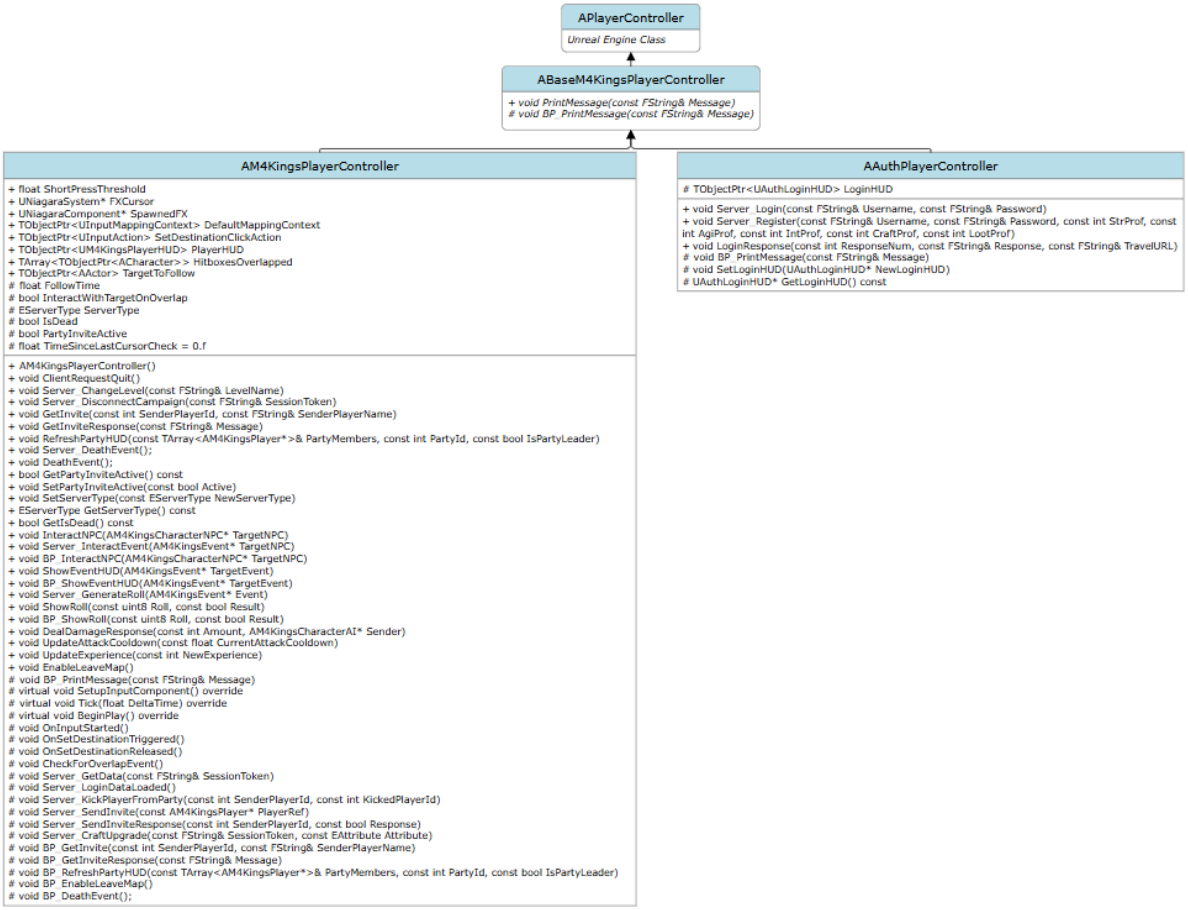
Projekt obsahuje rozsáhlé množství tříd, pro zlepšení čitelnosti jsou v této části uvedeny pouze diagramy stěžejních částí serverové a klientské části.

Na následujícím obrázku je znázorněno jádro serverové architektury. Základní serverová třída dědí z třídy AGameMode, která je součástí Unreal Engine. Tato základní třída nese název ABaseM4KingsGameMode a obsahuje společné funkcionality, které dále využívají její potomci – konkrétně AAuthServer a AGameServer. Třída ABaseM4KingsGameMode zároveň obsahuje vazbu na třídu FM4KingsDatabase, která zajišťuje komunikaci s REST API (s databází).



Obrázek 25 – Diagram tříd hlavní serverové části (zdroj vlastní)

Druhý diagram znázorňuje strukturu klientské části zaměřené na ovládání postavy a interakci se světem. Základním prvkem je třída ABaseM4KingsPlayerController, která dědí z APlayerController, základní třídy Unreal Engine pro ovládání hráče. Z této třídy následně vycházejí specializovaní potomci – třída AM4KingsPlayerController, která se stará o samotné herní ovládání, a třída AAuthPlayerController, která řeší proces přihlašování na straně klienta.



Obrázek 26 – Diagram tříd hlavní klientské části (zdroj vlastní)

7.6 Problematika síťové synchronizace

Jelikož se jedná o multiplayer server-klient, síťová synchronizace představuje nevyhnutelný problém. Unreal Engine většinu optimalizací a synchronizací řeší automaticky, ale ne všechno je tak jednoduché. Jeden z problémů nastává při replikaci dat, zejména když hráč přechází mezi servery. Jakmile se připojí, pošle serveru požadavek na načtení svých hráčských dat. Server tato data načte z databáze a nastaví je na serveru. Ovšem, tato data se na klienta dostanou až prostřednictvím replikace, která není okamžitá. Server sice po nastavení dat informuje klienta, že vše je připraveno, ale klient ještě nemá data, protože replikace ještě probíhá. Klient tedy čeká a neustále kontroluje, zda už jsou data zreplikovaná. Jakmile dojde k replikaci, hráči se konečně zobrazí hra místo zobrazeného "Loading data".

7.7 Technické neúspěchy

S takto rozsáhlým projektem byla větší četnost technických problémů prakticky nevyhnutelná. První komplikace nastala již při vytváření samotného projektu – na disku nebyl dostatek volného místa, což znemožnilo dokončení iniciální konfigurace a stažení potřebných knihoven. Dalším problémem byla kompilace samotného Unreal Enginu ze zdrojových kódů. Ta vyžaduje specifické závislosti a vývojářské nástroje (např. .NET SDK, Visual Studio s konkrétními workloady a Windows SDK), které jsem neměl správně nainstalované, což vedlo k opakovaným chybám při sestavení.

Největší komplikace však přišla při návrhu systému pro ukládání herních dat. V první verzi jsem se pokusil uchovávat veškeré informace o hráči (např. atributy, zkušenosti) přímo v paměti serveru. Již od začátku mi bylo jasné, že tato strategie je nepoužitelná – po vypnutí serveru by hráč přišel o veškerý postup.

Následoval přechod na SQLite – databázi uloženou v jediném souboru, který je snadno přenositelný a nevyžaduje žádný další server. Brzy se však ukázal její zásadní nedostatek: problém se současným přístupem více instancí serveru. Pokud se pokusily o paralelní zápis, mohlo dojít ke kolizi a selhání transakce. V praxi by to znamenalo nutnost ošetřit výjimky u každého dotazu do databáze, což je dlouhodobě neudržitelné řešení.

Bylo proto rozhodnuto o přechodu na robustnější databázové řešení – MySQL, které běží jako samostatná služba a zvládá více paralelních dotazů. I zde ale následovala komplikace: Unreal Engine nepodporuje přímou komunikaci s MySQL databázemi, alespoň ne bez externích knihoven. Pokusil jsem se implementovat oficiální C++ MySQL Connector od vývojářů MySQL, ale i po stovkách pokusů se jej nepodařilo správně integrovat do Unreal Engine.

Z tohoto důvodu bylo nakonec zvoleno elegantnější a rozšířenější řešení – REST API postavené na ASP.NET Core, které zprostředkovává komunikaci mezi hrou a databází. Tato architektura umožňuje snadnou správu požadavků, autentizaci, škálování i napojení dalších služeb bez nutnosti přímé integrace databázových knihoven do enginu. REST API se od té doby stalo stabilním a spolehlivým pilířem serverové komunikace mé hry.

Během vývoje se také přešlo z doporučeného Visual Studia na Rider od společnosti JetBrains – vývoj v Rideru je oproti Visual Studiu výrazně plynulejší a přehlednější. Rider nabízí výrazně rychlejší indexaci, inteligentní navigaci a našeptávání ve složitých C++ projektech a celkově příjemnější vývojářské prostředí, které významně urychlilo práci a zvýšilo komfort při práci s Unreal Enginem.

8 Návod na spuštění

Jelikož se jedná o klient-server aplikaci, je nejprve nutné spustit serverovou část, aby se k ní mohli klienti připojit. Systém je navržen pro běh v lokální síti, a proto je důležité mít nakonfigurovanou statickou IP adresu 192.168.0.252, na kterou se servery automaticky odkazují.

8.1 Spuštění serverové části

Pro spuštění herního serveru postačuje spustit pouze Auth server, který si dle potřeby automaticky vytvoří a spustí vlastní instanci Lobby serveru. Auth server se spouští pomocí následujícího příkazu v příkazové řádce:

%CESTA_K_EXE%\M4KingsServer.exe /Game/Auth/Login/LoginLevel -server -log -port=12345

%CESTA_K_EXE% označuje cestu ke složce, ve které se nachází spustitelný soubor M4KingsServer.exe. Tento příkaz spustí Auth server naslouchající na adrese 192.168.0.252 a portu 12345. Server implicitně očekává, že:

- Auth server běží na portu 12345
- Lobby server poběží na portu 12346 (automaticky spuštěný Auth serverem)

Dále je doporučeno využít Docker kontejnery pro jednodušší správu databáze a rozhraní REST API:

- mysql-container – obsahuje databázi s uživatelskými daty, informacemi o postavách a herními záznamy.
- m4kingsrestapi – REST API vytvořené v technologii ASP.NET Core, které slouží jako prostředník mezi herním serverem a databází.

Doporučený způsob spuštění těchto kontejnerů je pomocí příkazů uvedených v souboru DockerInit.txt, který je součástí projektu M4KingsRESTAPI. Tento soubor obsahuje definice i potřebné příkazy pro vytvoření a spuštění REST API kontejneru.

Po spuštění kontejnerů by mělo být zajištěno následující:

- REST API bude dostupné na adrese: http://localhost:8081
- MySQL databáze bude dostupná na portu: 3306

Po spuštění kontejnerů je ještě nutné vytvořit tabulky v databázi pomocí příkazů uvedených v souboru DBInit.sql, který je také součástí projektu M4KingsRESTAPI.

8.2 Spuštění klienta

Po spuštění serveru a backendových služeb se mohou připojit klienti. Klient se spouští prostřednictvím spustitelného souboru M4KingsClient.exe, který představuje kompletní herní klient.

Po spuštění se aplikace automaticky pokusí navázat spojení s autentizačním serverem, který je standardně dostupný na IP adrese 192.168.0.252 a portu 12345. V případě úspěšného připojení je klient připraven k přihlášení uživatele a následnému vstupu do herního lobby.

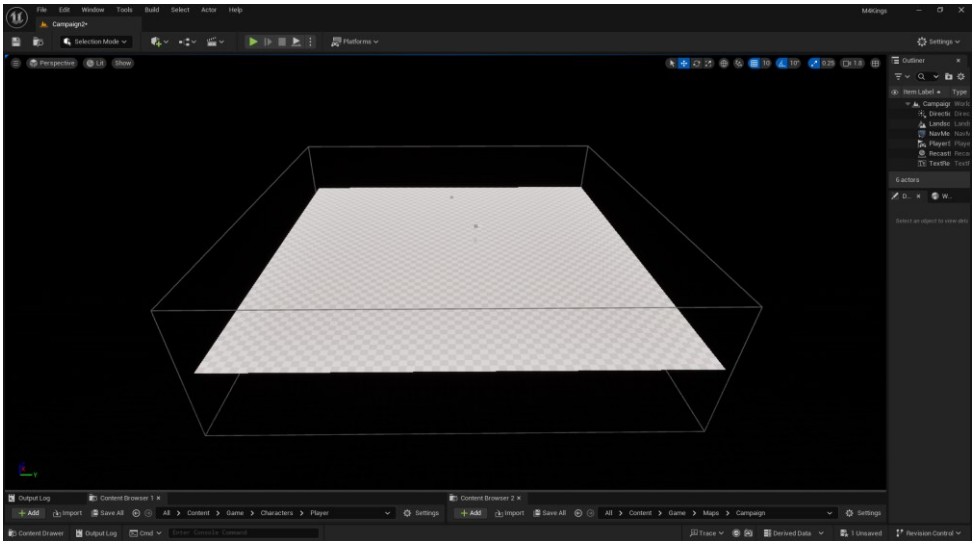
8.3 Doplňující poznámky

- IP adresa a port jsou pevně nastaveny ve zdrojovém kódu. Pro provoz na jiné adrese je potřeba tyto hodnoty upravit v projektu.
- V případě potíží doporučuji zkontrolovat firewall, správnost zadaných cest a případně log výstup serveru.
- REST API a MySQL musí být spuštěny před připojením hráče, jinak dojde k chybám při autentizaci nebo načítání dat.
- Klient i server je sestaven pouze pro OS Windows.

9 Grafický vývoj projektu

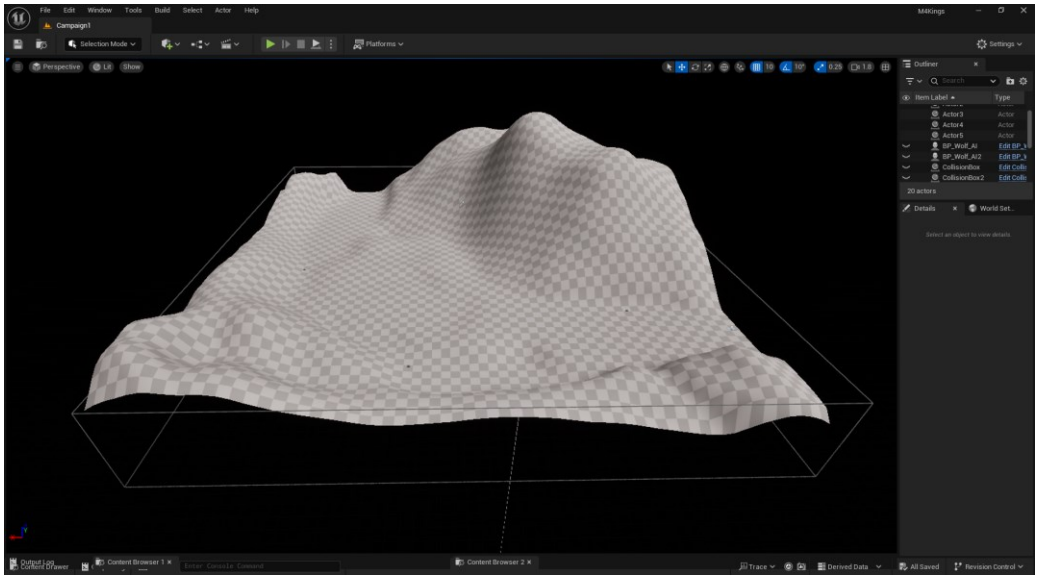
Tato kapitola dokumentuje průběh grafického vývoje projektu, který začal zcela od základu a postupně se vyvíjel až do současné podoby.

Na počátku vývoje se projekt skládal pouze z jednoduché ploché mapy bez jakýchkoliv modelů či textur. V tomto stádiu šlo primárně o ověření funkčnosti základní herní logiky a propojení klienta se serverem.



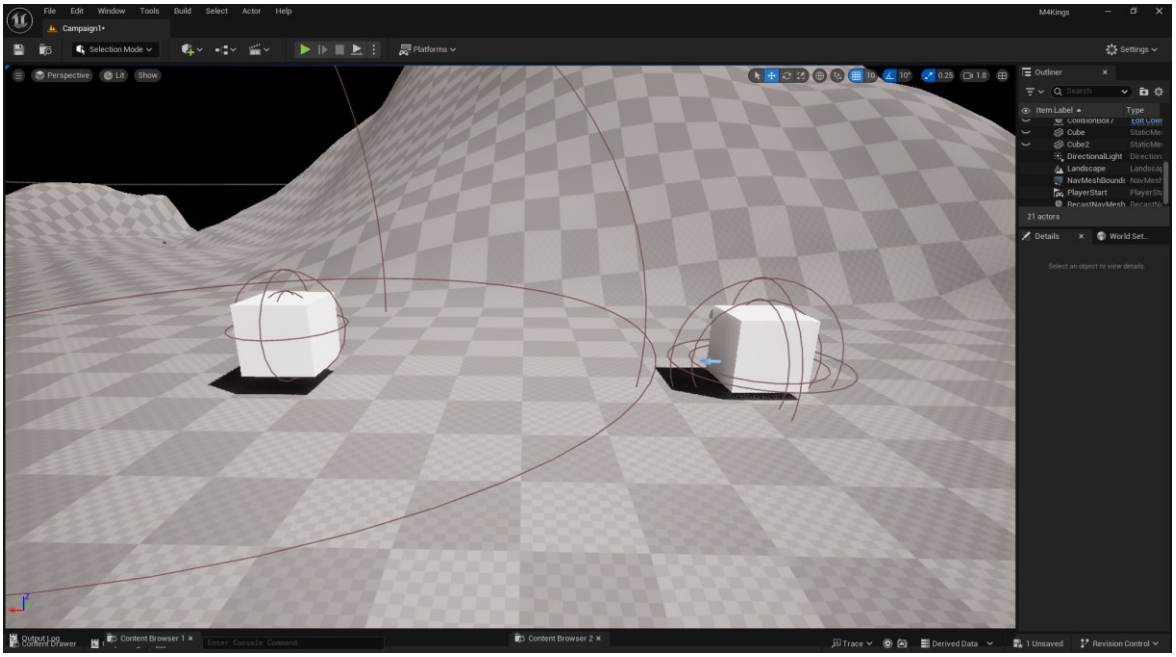
Obrázek 27 – Počátek projektu (zdroj vlastní)

Dalším krokem bylo vytvoření základního terénu pomocí nástroje Landscape v prostředí Unreal Engine. Jednoduchá krajina sloužila jako podklad pro testování pohybu postav, kolizí a základních interakcí.



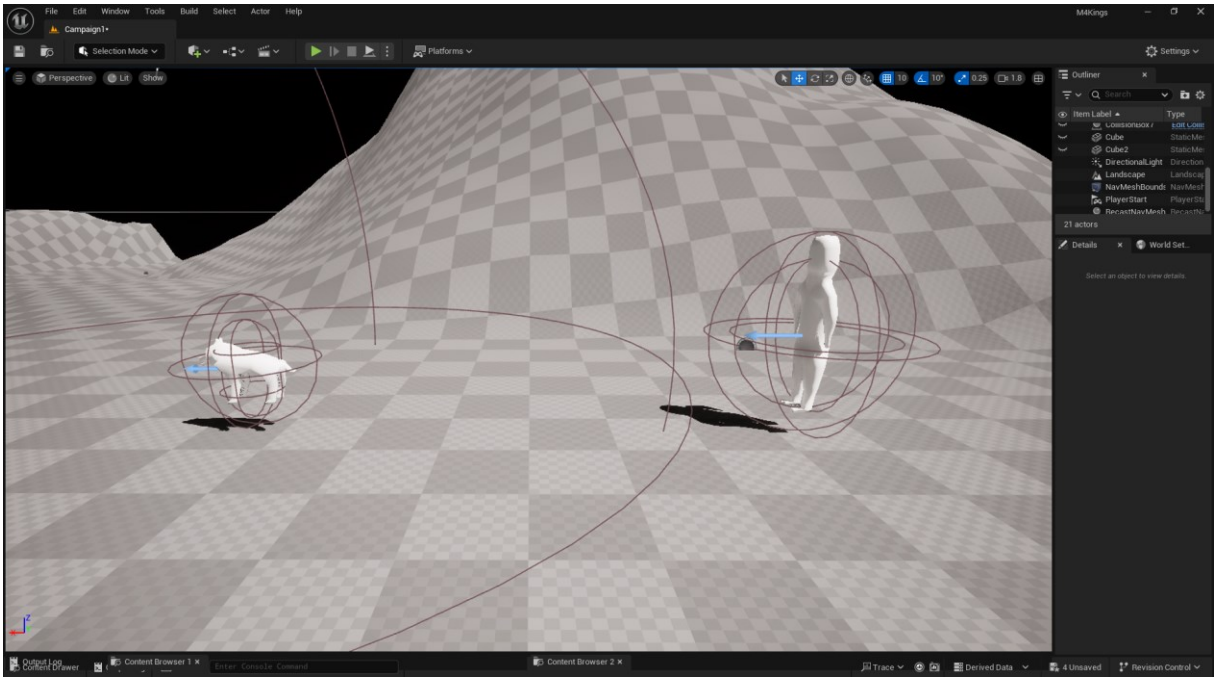
Obrázek 28 – Úprava terénu (zdroj vlastní)

Po vytvoření základního prostředí následoval tzv. blocking postav. Tento přístup umožnil testování funkčnosti a interakcí ještě před finálním vytvořením 3D modelů. Postavy byly zastoupeny jednoduchými objekty.



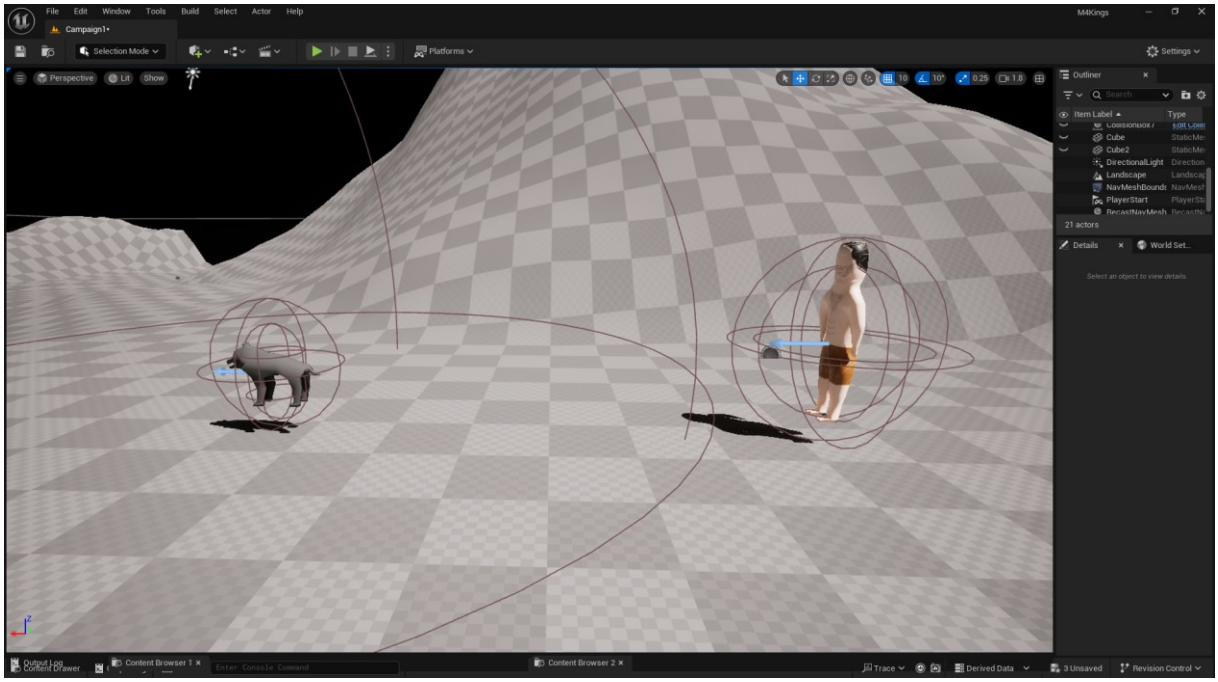
Obrázek 29 – Blocking postav (zdroj vlastní)

Po ověření funkčnosti jednotlivých herních mechanik následovala tvorba vlastních 3D modelů. V nástroji Blender byly vytvořeny modely hráče a nepřátelského vlka, které byly následně importovány do Unreal Engine.



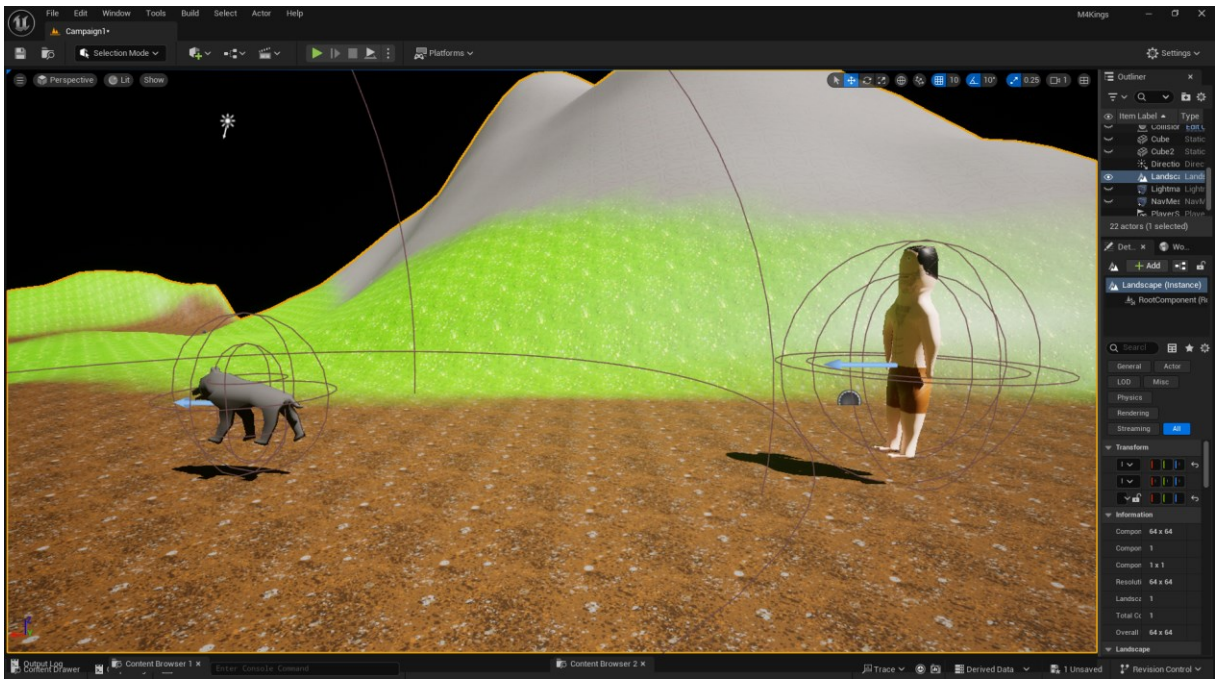
Obrázek 30 – Import modelů (zdroj vlastní)

Po importu byly tyto modely doplněny o vlastní textury, které rovněž vznikly v nástroji Blender.



Obrázek 31 – Textury modelů (zdroj vlastní)

Nakonec byl opět využit nástroj Landscape, tentokrát však pro aplikaci vlastních textur na prostředí. Byly použity vlastní textury trávy, cesty a skal, které vizuálně oživily herní svět a dodaly prostředí potřebnou atmosféru.



Obrázek 32 – Textury krajiny (zdroj vlastní)

10 Výsledná hra

Výsledkem projektu je funkční prototyp multiplayerové CO-OP RPG hry, který splňuje všechny stanovené požadavky a cíle práce. Hra byla vyvinuta v prostředí Unreal Engine 5 s použitím jazyka C++ a využívá architekturu klient-server, kdy server ověřuje všechny důležité operace hráče.

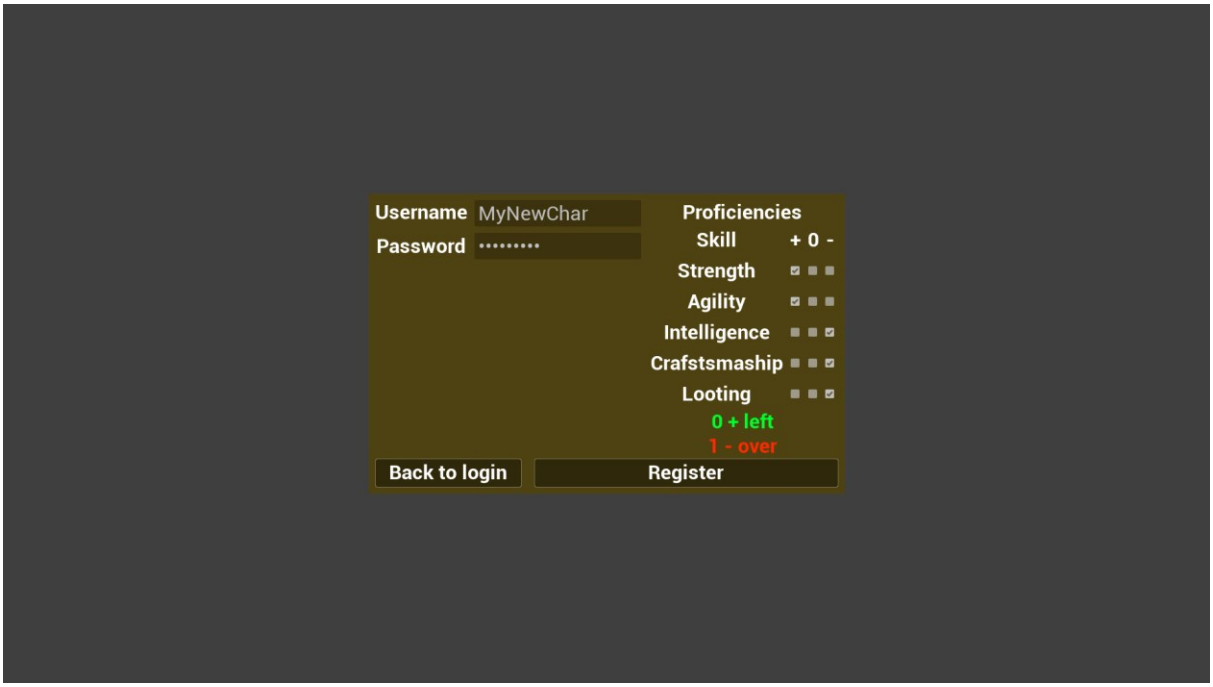
Po spuštění hry a následném přihlášení (registraci) se hráč dostane do města (lobby). V lobby má hráč možnost zakládat nebo se připojovat do skupin společně s ostatními hráči. Skupina poté může společně vstoupit do instance výpravy.

Instanční úrovně jsou zaměřeny na kooperativní boj s nepřáteli, sbírání zkušeností, získávání crafting materiálů a řešení náhodných událostí (otevírání truhel). Hráč postupně sbírá zkušenosti, zvyšuje svou úroveň a vylepšuje svoji postavu pomocí výroby. Po návratu z výpravy se data o dosaženém pokroku ukládají na server prostřednictvím REST API. Díky serverové validaci jsou eliminovány pokusy o neautorizovanou manipulaci s daty.

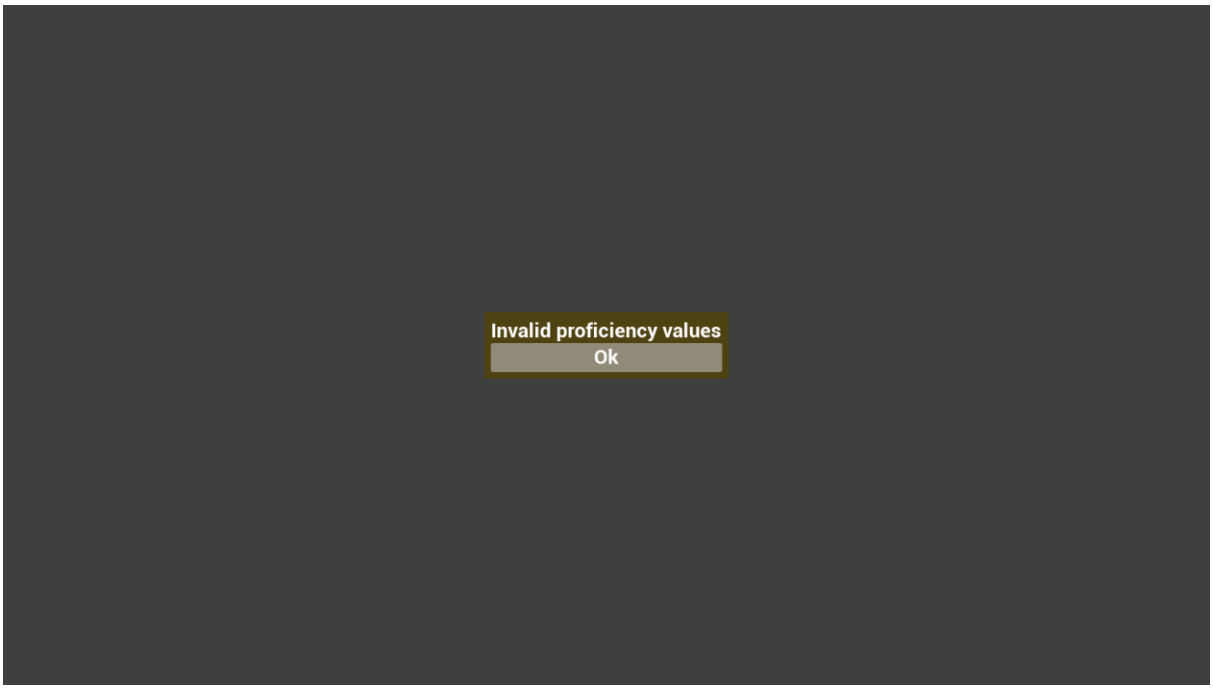
Grafické zpracování hry je laděno do low-poly stylizace. Hráči i nepřátelé mají základní animace pohybu. Uživatelské rozhraní zajišťuje přehledné zobrazování stavu hráče i důležitých herních událostí.

Součástí vývoje bylo také zajištění modularity a rozšiřitelnosti. Herní systém umožňuje snadné přidávání nových výprav či nepřátel bez nutnosti zásadních úprav stávající architektury.

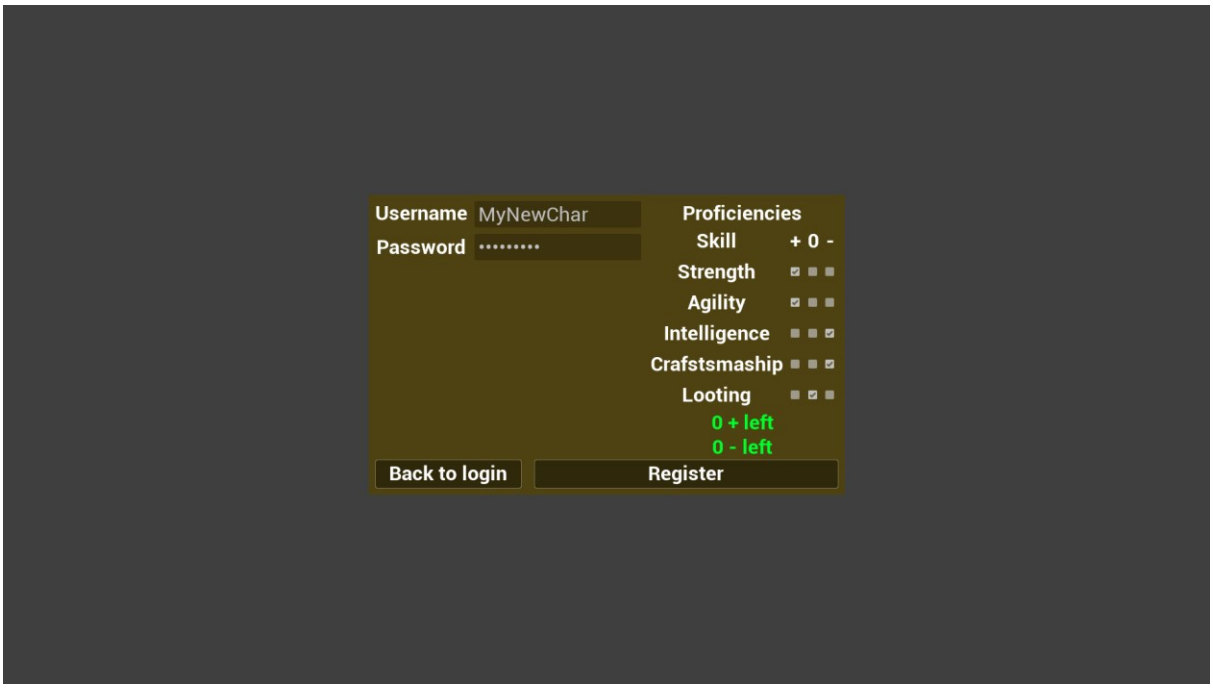
Dále je uvedeno několik ukázek z prostředí hry, zachycujících různé fáze hry:



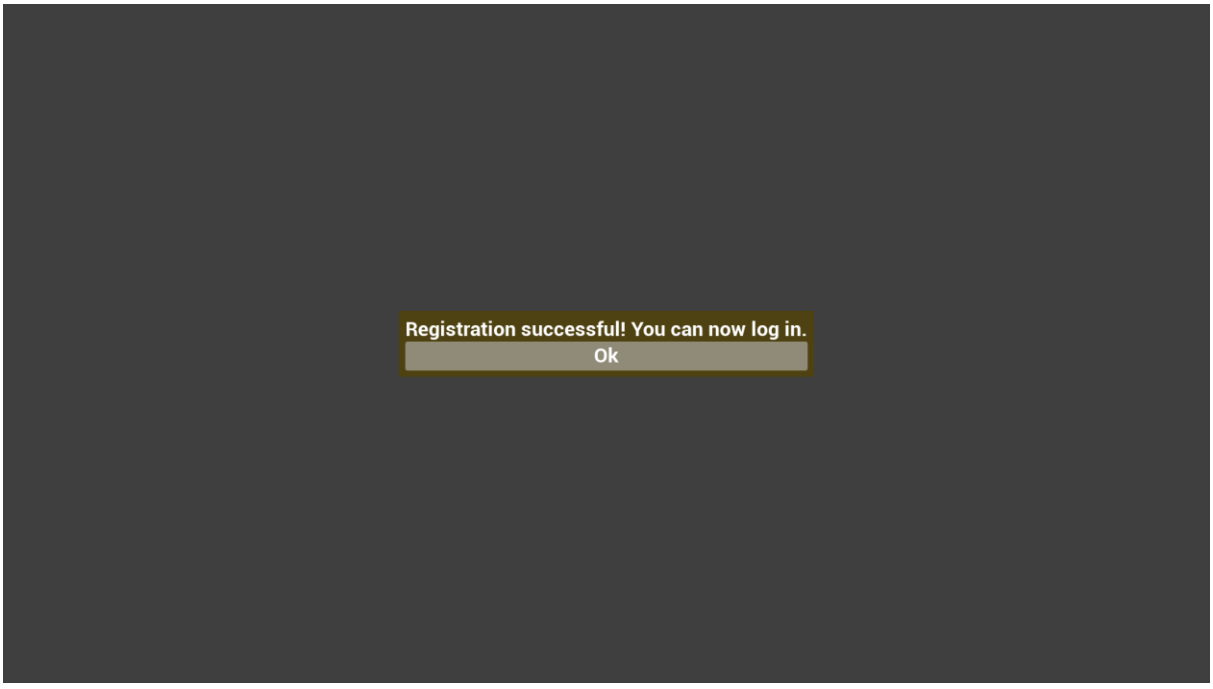
Obrázek 33 – Okno registrace s neplatnými hodnotami (zdroj vlastní)



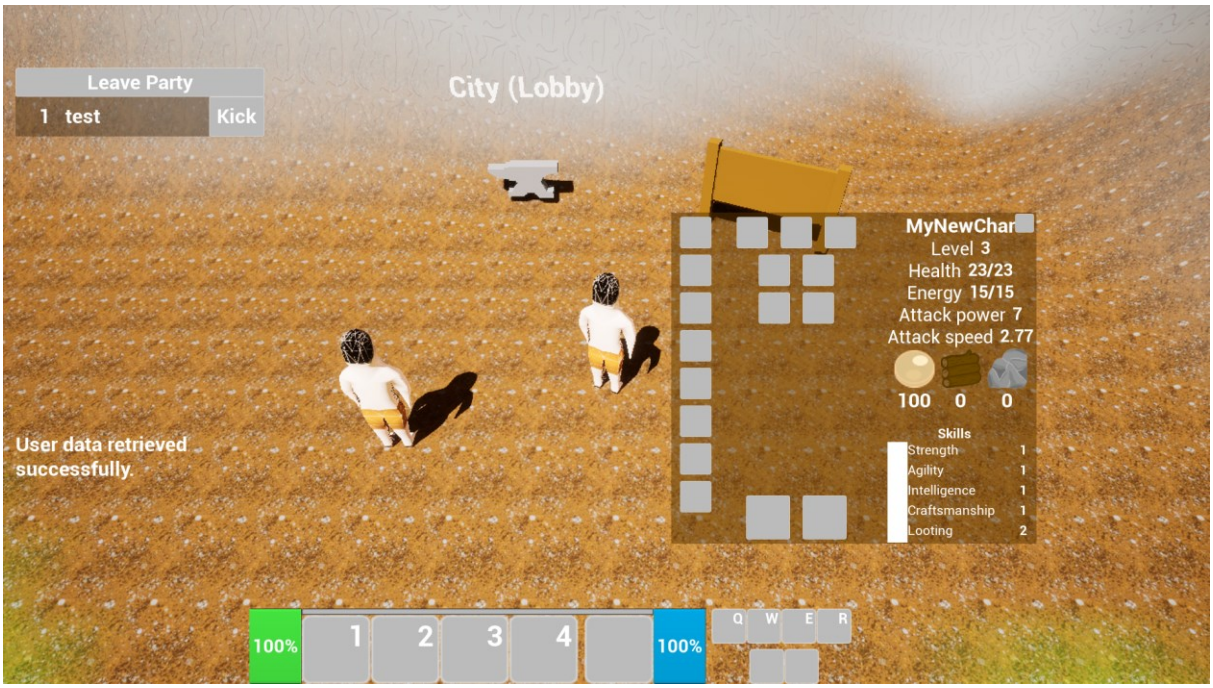
Obrázek 34 – Neúspěšná registrace (zdroj vlastní)



Obrázek 35 – Okno registrace s platnými hodnotami (zdroj vlastní)

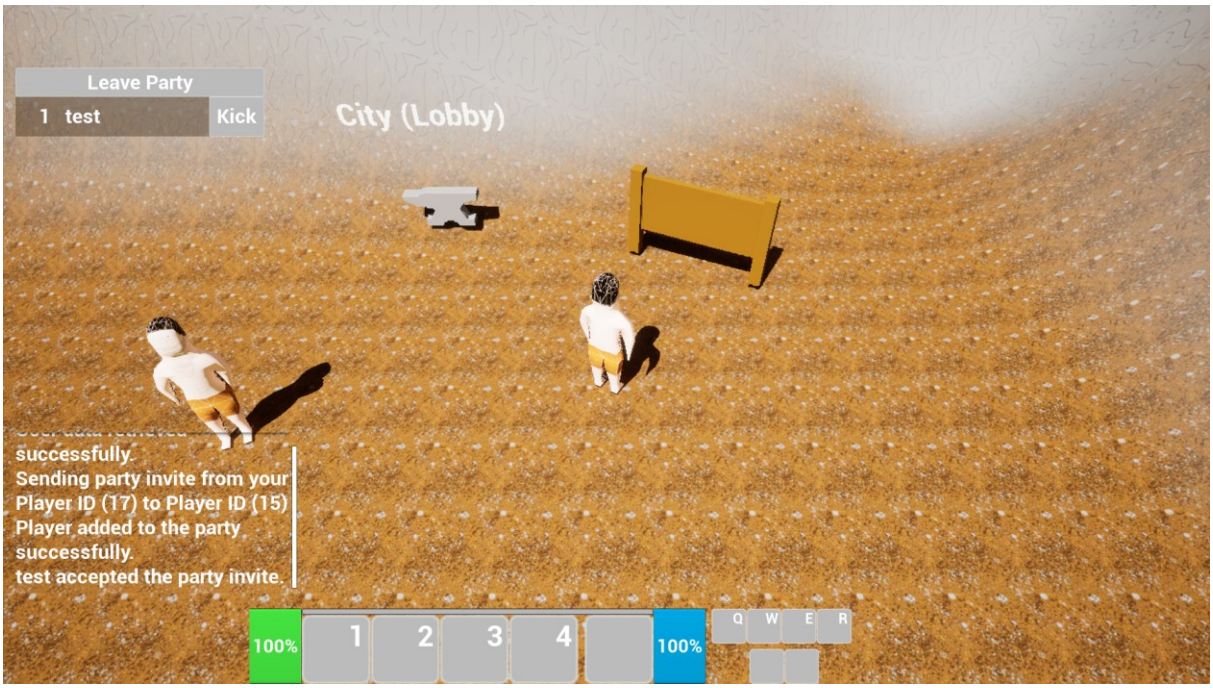


Obrázek 36 – Úspěšná registrace (zdroj vlastní)



Obrázek 37 – Lobby + inventář (zdroj vlastní)

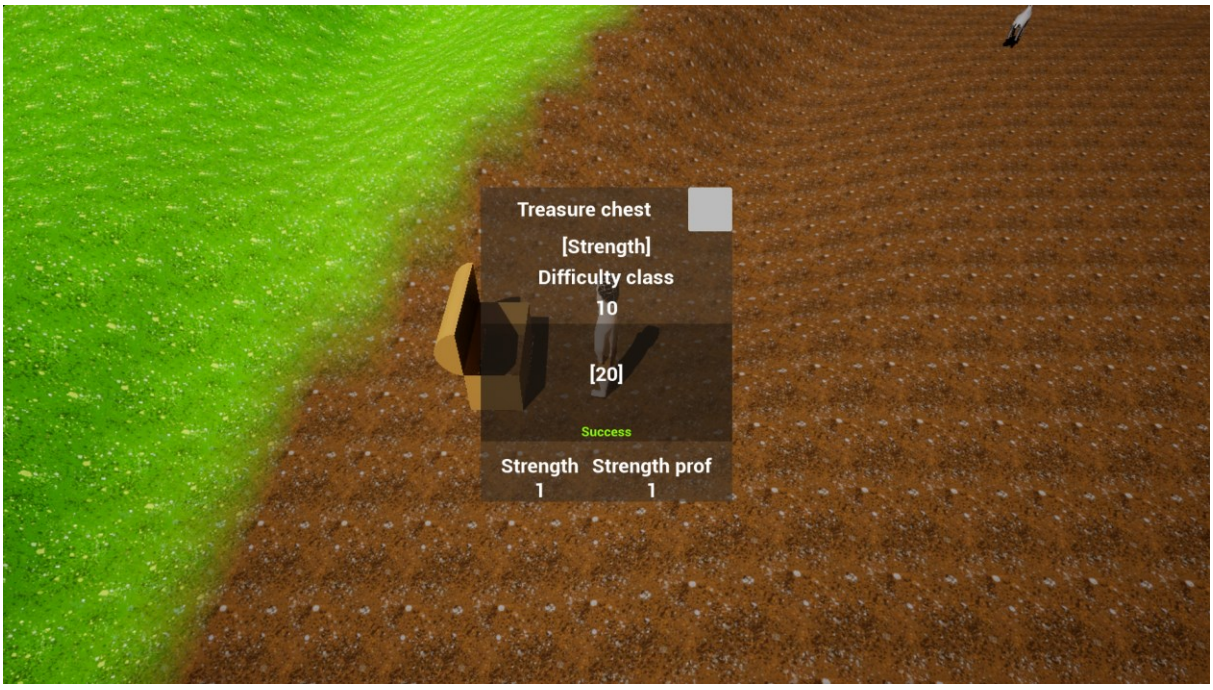
Obrázek 38 – Lobby s pozvánkou do party (zdroj vlastní)



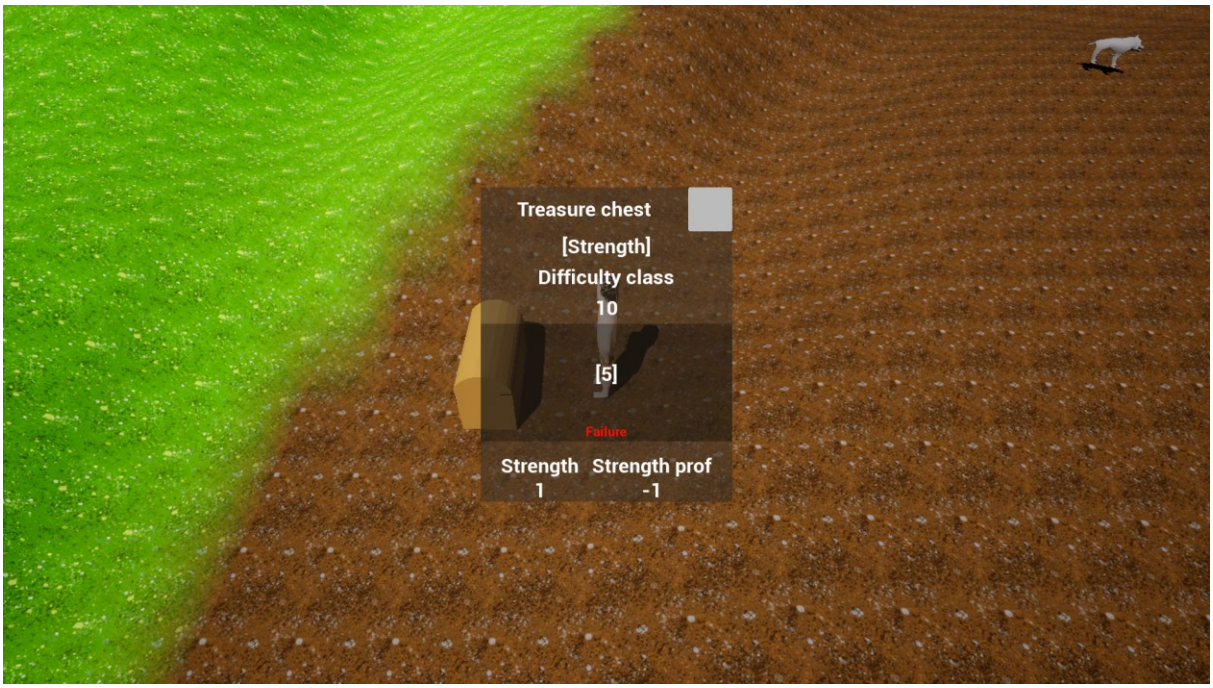
Obrázek 39 – Lobby s partou (zdroj vlastní)

Obrázek 40 – Výprava (zdroj vlastní)

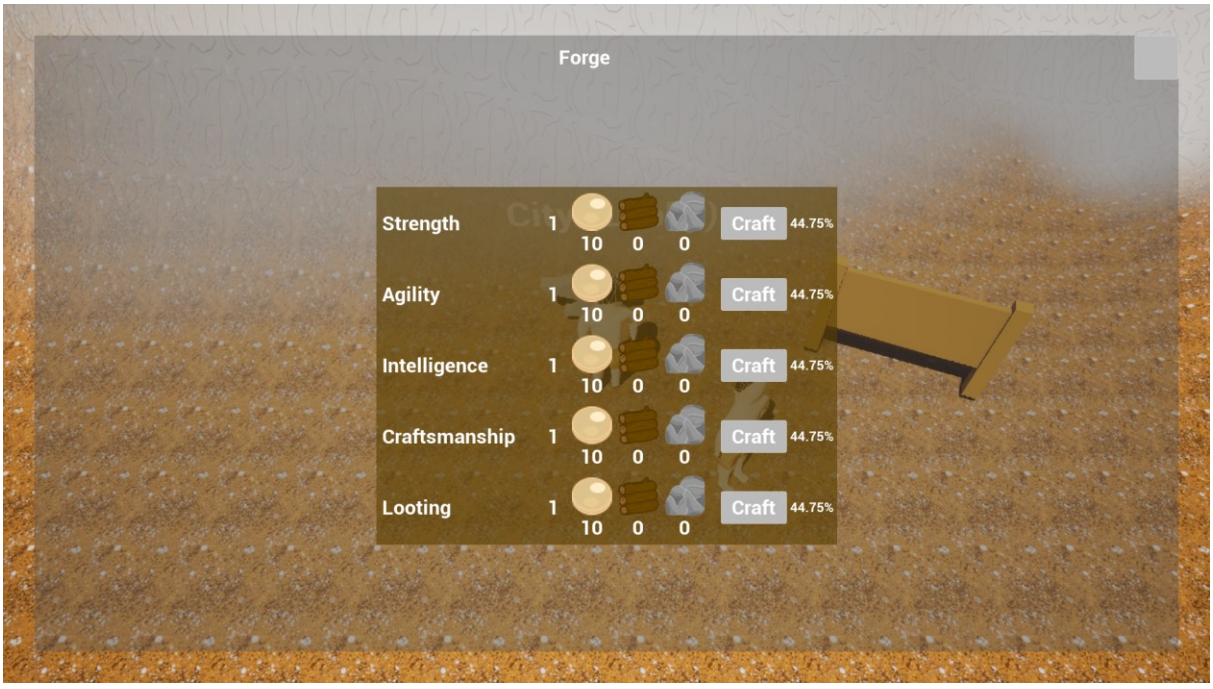
Obrázek 41 – Poškození nepřítele na výpravě (zdroj vlastní)



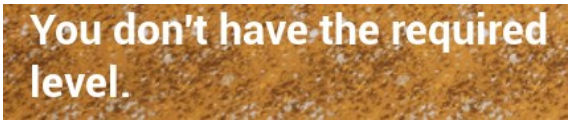
Obrázek 42 – Úspěšné otevření truhly se zvolenou specializací na atribut síly (zdroj vlastní)



Obrázek 43 – Neúspěšné otevření truhly se zvolenou nevýhodou na atribut síly (zdroj vlastní)



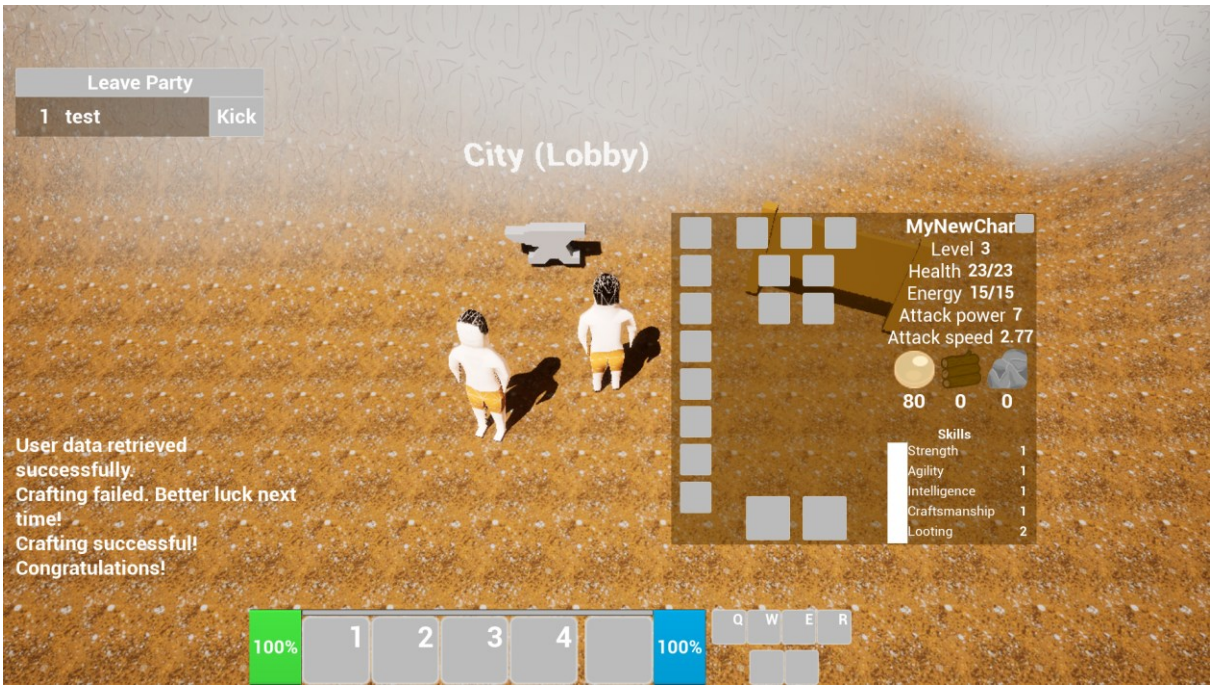
Obrázek 44 – Okno výroby (zdroj vlastní)



Obrázek 45 – Neúspěšná výroba (nedostatečná úroveň hráče) (zdroj vlastní)

Obrázek 46 – Neúspěšná výroba (nedostatek surovin) (zdroj vlastní)

Obrázek 47 – Neúspěšná výroba (šance) (zdroj vlastní)



Obrázek 48 – Úspěšná výroba atributu Looting (zdroj vlastní)

Vývoj výsledné hry probíhal postupně od základních testovacích prvků až po plnohodnotné herní funkce. Na počátku (Obrázek 27) byl projekt tvořen pouze prázdnou plochou bez jakékoliv interakce. Následně byl vytvořen terén pomocí nástroje Landscape (Obrázek 28) a přidány testovací modely pro základní ověřování herních mechanik (Obrázek 29). Po úspěšném otestování byly vytvořeny a importovány vlastní modely postav (Obrázek 30), které následně dostaly textury vytvořené v nástroji Blender (Obrázek 31). Finální úprava terénu byla provedena aplikací vlastních textur krajiny (Obrázek 32), čímž hra získala svůj vizuální styl. Dále byly implementovány klíčové herní funkce jako registrace a přihlašování (Obrázky 33–36), správa lobby a party systému (Obrázky 37–39), výpravy a interakce s herním světem (Obrázky 40–43) a výrobní systém s různými výstupy podle úspěchu či neúspěchu (Obrázky 44–48). Tyto kroky ukazují vývoj od technického základu až po komplexní herní systém s přehledným UI a síťovou logikou.

11 Možnosti rozšíření

Do budoucna se nabízí řada možností, jak hru dále rozšířit a obohatit herní zážitek. Mezi hlavní směry rozšíření patří:

1. Rozšíření hratelných tříd
    - V současnosti je dostupná pouze třída válečníka, nicméně nabízí se možnost přidat další třídy, jako například lučištníka zaměřeného na útoky na dálku, nebo mága specializovaného na kouzla a magii.
    - Takové rozšíření by umožnilo hráčům lépe přizpůsobit herní styl své preferenci a zvýšilo variabilitu týmových kombinací.
2. Zavedení systému subclass
    - Inspirováno pravidly Dungeons & Dragons, nabízí se možnost umožnit hráčům další specializaci v rámci hlavní třídy.
    - Například válečník by se mohl profilovat jako barbar zaměřený na maximální poškození, nebo jako rytíř se schopností chránit partu.
    - Tento systém by přispěl k větší hloubce a diverzitě vývoje postav.
3. Kooperativní schopnosti ovlivňující celou partu
    - Lze zavést schopnosti, které by například zvyšovaly rychlost útoků, regeneraci zdraví nebo obranu celé skupiny.
    - Tento prvek by podpořil týmovou spolupráci a strategické plánování.
4. Rozšíření typů nepřátel
    - Nabízí se přidání nových nepřátel s unikátními schopnostmi a taktikami.
    - Někteří nepřátelé by mohli být dominantní v boji na dálku, jiní by naopak preferovali boj zblízka, což by vyžadovalo od hráčů různorodé přístupy.
5. Zavedení systému aggro
    - Lze vylepšit umělou inteligenci nepřátel zavedením aggro systému – nepřátelé by dynamicky reagovali na míru hrozby, například podle způsobeného poškození.

- Tento systém by zvýšil potřebu taktické koordinace mezi hráči.

6. Strukturované mise s cílovými bossy
    - Je možné zavést mise zaměřené na porážku specifického bosse s unikátními schopnostmi.
    - Bossové by představovali významnou výzvu, která by vyžadovala koordinovanou týmovou přípravu.
7. Propracovanější systém vylepšování postavy
    - Nabízí se možnost ovlivňovat základní atributy zvolené specializace procentuálně místo pouhého +1 či -1, nebo zavést systém advantage/disadvantage inspirovaný D&D.
    - Alternativně lze rozšířit postup postavy prostřednictvím stromů dovedností, které by umožnily odemykání specializovaných schopností.
8. Systém výroby
    - Lze uvažovat o možnosti sbírat suroviny a vyrábět vlastní výbavu pomocí unikátních receptů.
    - Tento prvek by motivoval k průzkumu světa a hledání skrytých truhel.
9. Náhodné události
    - Nabízí se možnost inspirovat se roguelike žánrem a přidat náhodné environmentální překážky, například lávové kaluže, které by vyžadovaly testy atributů, kde neúspěch při překonávání těchto překážek by mohl trvale ovlivnit průběh výpravy.
    - Lze přidat například pasti, hádanky, setkání s NPC či environmentální výzvy.
    - Tyto prvky by významně zvýšily variabilitu výprav a nabídly hráčům nové možnosti interakce se světem.
10. Příběh a quest systém

ZÁVĚR

Tento projekt jsem založil již během léta 2024. Během několikaměsíčního vývoje jsem vytvořil plně funkční základ multiplayerové CO-OP RPG hry, která obsahuje kompletní systém registrace a přihlašování uživatelů přes REST API, správu relace pomocí session tokenu, lobby systém s podporou party mechanik, možnost spouštět instance kampaní a základní síťovou synchronizaci hráčských dat a herních objektů. Většina herní logiky běží na serveru, zatímco klient zajišťuje prezentaci a interakci. Celý projekt vznikal v prostředí Unreal Engine 5 za pomoci C++ a Blueprints, přičemž k backendové komunikaci s databází slouží ASP.NET Core aplikace s MySQL databází běžící v Dockeru.

Na vývoji jsem strávil stovky hodin – od návrhu architektury, přes řešení síťové replikace, až po debugging nezdokumentovaných problémů Unreal Enginu a hledání ideálních nástrojů a řešení. Například přechod z Visual Studia na JetBrains Rider byl jedním z klíčových rozhodnutí, které výrazně zrychlilo vývojový proces a zpříjemnilo práci v C++.

Tato práce pro mě znamenala zásadní přínos nejen po technické, ale i po osobní stránce. Získal jsem hluboké porozumění principům tvorby komplexních síťových her a osvojil si práci s celou řadou moderních technologií napříč různými vrstvami systému. Naučil jsem se navrhovat a implementovat architekturu klient-server, efektivně pracovat s REST API, spravovat session mechaniky, synchronizovat data mezi klientem a serverem v reálném čase a integrovat databázovou logiku do herního prostředí.

Vedle technických znalostí jsem si vyzkoušel také projektové řízení vlastního vývoje – od plánování jednotlivých komponent přes řešení složitých problémů až po testování a ladění. Vývoj takto komplexního systému mi ukázal, jak důležitá je schopnost hledat řešení napříč technologiemi a jak klíčové je porozumět jejich vzájemné souhře.

Celý projekt považuji za důležitý milník ve svém odborném růstu. Nejenže mi umožnil ověřit si nabyté znalosti v praxi, ale zároveň mě posunul dál v oblastech, které bych chtěl rozvíjet i do budoucna – ať už jde o herní vývoj, backendové systémy, nebo obecně výstavbu větších softwarových architektur. Výsledná hra představuje pevný základ, který lze dále rozšiřovat a na kterém je možné budovat rozsáhlejší funkce, obsah i multiplayer mechaniky.

POUŽITÁ LITERATURA

[1] CONTRIBUTORS, Wikipedia. Cel shading. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/Cel_shading.

[2] CONTRIBUTORS, Wikipedia. Low poly. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/Low_poly.

[3] Couldn't find publisher. Online. [cit. 2025-04-10].

[4] Historie multiplayerových her. Online. INDIAN. Dostupné z: https://indian-tv.cz/clanek/historie-multiplayerovych-her-7aomf4. [cit. 2025-04-11].

[5] CONTRIBUTORS, Wikipedia. History of video games. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/History_of_video_games.

[6] A.S., Alza. Multiplayer: Definice a historie. Online. Alza.cz. Dostupné z: https://www.alza.cz/slovnik/multiplayer-art16765.htm. [cit. 2025-04-11].

[7] TECNODIGITAL. Softwarová architektura klient-server: Kompletní průvodce. Online. Informática y Tecnología Digital. Dostupné z: https://informatecdigital.com/cs/architektura-softwaru-klientsk%C3%A9ho-serveru-kompletn%C3%AD-pr%C5%AFvodce/. [cit. 2025-04-11].

[8] KAIYUE. Why Is Edge Computing Essential for Reducing Latency in Online Gaming? Online. Edgenext - EdgeNext | Accelerate Your Digital Experience with Edge Computing, CDN, and Advanced DDoS Protection. Dostupné z: https://www.edgenext.com/why-is-edge-computing-essential-for-reducing-latency-in-online-gaming/. [cit. 2025-04-28].

[9] AI For Game Security And Anti-Cheat Systems. Online. Restackio. Dostupné z: https://www.restack.io/p/ai-for-surveillance-systems-answer-game-security-cat-ai. [cit. 2025-04-28].

[10] GOHARI, Pardis. Countering the ever-evolving scourge of cheating in games. Online. I3D.net. Dostupné z: https://www.i3d.net/countering-scourge-of-cheating-in-games/. [cit. 2025-04-28].

[11] PIRATE SOFTWARE. Pirate Software; YouTube. Online. Dostupné z: https://www.youtube.com/watch?v=nhoYSDGnXdM. [cit. 2025-04-09].

[12] CONTRIBUTORS, Wikipedia. Dynamic game difficulty balancing. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/Dynamic_game_difficulty_balancing.

[13] CONTRIBUTORS, Wikipedia. Game engine. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/Game_engine.

[14] Forging new paths for filmmakers on "The Mandalorian". Online. Unreal Engine. Dostupné z: https://www.unrealengine.com/en-US/blog/forging-new-paths-for-filmmakers-on-the-mandalorian. [cit. 2025-04-09].

[15] WOOD, Chandler. How The Mandalorian Used a Video Game Engine to Render Digital Locations in Real-Time. Online. PlayStation LifeStyle. Dostupné z: https://www.playstationlifestyle.net/2020/02/21/the-mandalorian-unreal-engine-render-real-time-led-screen/. [cit. 2025-04-09].

[16] ANDREW R. CHOW. Inside Epic's Unreal Engine 5—and What It Means for the Future of Gaming, Movies, and the Metaverse. Online. TIME. Dostupné z: https://time.com/6164332/epic-unreal-engine-5-launch/. [cit. 2025-04-09].

[17] Multi-purpose car simulation environment gets a boost from Unreal Engine. Online. Unreal Engine. Dostupné z: https://www.unrealengine.com/en-US/spotlights/multi-purpose-car-simulation-environment-gets-a-boost-from-unreal-engine. [cit. 2025-04-09].

[18] SimCentric scales up tactical military simulation training with Unreal Engine. Online. Unreal Engine. Dostupné z: https://www.unrealengine.com/en-US/spotlights/simcentric-scales-up-tactical-military-simulation-training-with-unreal-engine. [cit. 2025-04-09].

[19] Unreal Engine logo. Online. In: Wikipedia. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/d/da/Unreal_Engine_Logo.svg. [cit. 2025-04-28].

[20] Licensing. Online. Unreal Engine. Dostupné z: https://www.unrealengine.com/en-US/license. [cit. 2025-04-06].

[21] CONTRIBUTORS, Wikipedia. Client-side prediction. Wikipedia, The Free Encyclopedia. Dostupné také z: https://en.wikipedia.org/wiki/Client-side_prediction.

[22] Phc-winner-argon2. Online. GitHub. Dostupné z: https://github.com/P-H-C/phc-winner-argon2. [cit. 2025-04-28].

SEZNAM PŘÍLOH

Příloha A: src.zip (zdrojové kódy a assety hry)

Příloha B: rest_api_src.zip (zdrojové kódy REST API)

Příloha C: build_win_server.zip (spustitelný herní server)

Příloha D: build_win_client.zip (spustitelný herní klient)