Razvoj horor igre sa preživljavanjem za više igrača u Unreal Engine 4 razvojnom okruženju

Zeko, Karlo

Master's thesis / Diplomski rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:659328

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-16**

Repository / Repozitorij:

Digital Repository Juraj Dobrila University of Pula

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

KARLO ZEKO

RAZVOJ HOROR IGRE SA PREŽIVLJAVANJEM ZA VIŠE IGRAČA U UNREAL ENGINE 4 OKRUŽENJU

Diplomski rad

Pula, rujan, 2023.

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

KARLO ZEKO

RAZVOJ HOROR IGRE SA PREŽIVLJAVANJEM ZA VIŠE IGRAČA U UNREAL ENGINE 4 OKRUŽENJU

Diplomski rad

JMBAG: 0303069414**, redoviti student**
Studijski smjer: Informatika
Predmet: Dizajn i programiranje računalnih igara
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: izv.prof.dr.sc Tihomir Orehovački

Pula, rujan, 2023.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani **Karlo Zeko**, kandidat za magistra **Informatike** ovime izjavljujem da je ovaj Diplomski rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Diplomskog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

U Puli, rujan 2023. godine

IZJAVA
o korištenju autorskog djela

Ja, **Karlo Zeko** dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj diplomski rad pod nazivom **RAZVOJ HOROR IGRE SA PREŽIVLJAVANJE ZA VIŠE IGRAČA U UNREAL ENGINE 4 OKRUŽENJU** koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.
Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, rujan 2023. godine

Potpis

SAŽETAK

Igra opisana u ovom diplomskom radu spada pod žanr horor igre sa preživljavanjem za više igrača. Okruženje korišteno za izradu ove igre bilo je „Unreal Engine 4“, koje već ima ugrađene neke mehanike za implementaciju igre za više igrača. Glavni dio rada sastoji se od opisa izrade same igre pomoću nacrta i prikazivanje funkcionalnosti igre. Cilj igre je obaviti razne zadatke i preživjeti skupljajući hranu koja se nalazi na mapi.

KLJUČNE RIJEČI: Igrač, replikacija, klijent, ugostitelj, poslužitelj, Unreal Engine 4, instanca, akter, funkcija, Horor igra sa preživljavanjem za više igrača, umrežavanje, lik, korisničko sučelje, Multicast, ažuriranje, zvuk, scena, mapa, sesija

ABSTRACT

The game described in this master's thesis is a computer game that falls under the multiplayer survival horror game subgenre. The environment used for developing this game is the Unreal Engine 4 environment, which already has some mechanics built in for implementing a multiplayer game. The main part of the paper consists of describing the development of the game itself using blueprints and showing the functionality of the game. The goal of the game is to complete various tasks and survive by collecting food found on the map.

KEY WORDS: Player, Replication, Client, Host, Server, Unreal Engine 4, Instance, Actor, Function, Multiplayer Survival Horror Game, Networking, Character, User Interface, Multicast, Update, Sound, Scene, Map, Session

SADRŽAJ

1. UVOD 1
2. INSPIRACIJE 2
3. POSTAVLJANJE IGARA ZA VIŠE IGRAČA 4
4. TEMELJNI KONCEPTI UMREŽAVANJA 6
5. REPLIKACIJA 8
6. STVARANJE IGRE 9
   6.1 Stvaranje instance o igri 9
   6.2 Lobi „Game Mode“ i lobi „Player Controller“ 12
   6.3 Korisničko sučelje 18
   6.4 Akteri za interakciju sa igračima 28
   6.5 Događaji koji se aktiviraju direktno na mapi 29
   6.6 Glavni akter 31
   6.7 Dizajniranje mape za igru 32
   6.8 Primjeri koda 35
7. OPIS IGRE 42
8. ZAKLJUČAK 47
LITERATURA 48
PRILOZI 49

1. UVOD

Okruženje za razvoj računalnih igara (engl. Game Engine) je softver koji omogućuje izradu računalnih igara iz temelja. Njegove značajke obično omogućuju uvoz multimedijskih datoteka kao što su 3D modeli, slike, zvuk i video, i manipuliranje tim datotekama korištenjem programskih jezika. Neki od jezika koji se koriste su „C++“, „Python“, „Lua“ i mnogi drugi (Cordone, 2019). „Survival horror“ je podžanr horor i akcijsko-avanturističkih videoigara koji se fokusira na preživljavanje igrača dok ih igra pokušava prestrašiti horor grafikom ili zastrašujućim ambijentom (DE, 2021).
Motivacija za izradu videoigre dolazi od želje za boljim razumijevanjem razvoja videoigara za više igrača koje zahtijevaju znanje u mrežnoj komunikaciju između više korisnika te o tome kako tu komunikaciju prenijeti na izradu videoigara.

Prvu inačicu „Unreal Engine“ (UE) okruženja razvio je „Tim Sweeney“, osnivač „Epic Games-a“. Neke od poznatih igara koje su razvijene u prvoj generaciji „UE-a“ su „Unreal Tournament“ i „Harry Potter and the Sorcerer's Stone“. Druga generacija zvana „UE2“ prvi se put pojavila 2002. godine sa igrom „America's Army“ koja je služila kao sredstvo novačenja za američku vojsku (Grove, 2019). Treća generacija zvana “UE3“ je prezentirana 2004. godine, a do tada je bila u razvoju 18 mjeseci, prva igra izdana pomoću „UE3“ bila je „Gears of War“. Četvrta generacija zvana „UE4“ službeno je objavljena 2014. godine kroz novi model licenciranja za mjesečnu pretplatu od 19 američkih dolara. Nakon godinu dana „Epic“ je pustio „UE4“ sa svim budućim ažuriranjima besplatno za sve korisnike. U zamjenu su tražili 5 % prihoda za proizvode koji zarađuju više od 3000 dolara po četvrtini godine. Peta generacija zvana „UE5“ izdana je 2022. godine za sve korisnike. Jedna od glavnih značajki je „Nanite“ sustav virtualizirane geometrije koji omogućuje uvoz visoko detaljnog izvornog fotografskog materijala u igre (Tarantola, 2020). Posljednja verzija izdana rujna 2023. godine je „UE5.3“ koja je donijela poboljšanja na glavnim alatima kao što su prikazivanje, izgradnja svijeta, alati za animacije i modeliranje, simulacije i više.

GitHub poveznica: https://github.com/Sofoklo/SurvivalHorrorMP

2. INSPIRACIJE

Neke od igara koje su korištene kao inspiracija pri izradi bile su „The Forest“ i „Outlast“. Sve te igre imaju nekakav dio koji je implementiran u ovu igru bila to atmosfera ili nešto u vezi sa radnjom u igri.

Igra The Forest je horor igra sa preživljavanjem koju su razvili „Endnight Games“ u „Unity“ okruženju. Igra se odvija na šumovitom i „praznom“ otoku gdje glavni igrač mora pronaći sina i pri tome se boriti sa kanibalističkim čudovištima. Ova igra sadrži nelinearnu radnju u okruženju otvorenog svijeta koja se igra iz perspektive prvog lica, bez ikakvih zadanih misija pružajući igraču da sam donosi odluke o svom preživljavanju. Neki od elemenata koji su iz ove igre integrirani u igru iz ovog rada su traka koja prati razne podatke o igraču, prikazivanje zadatka kako bih igrač znao što mu je trenutna misija te je također pružila je inspiraciju u smislu preživljavanja u divljini i u izgledu krajolika koji je zastrašujući po noći kao što je prikazano na slici 1. koja prikazuje snimku ekrana tokom igranja The Forest.

Slika 1 Snimka ekrana iz igre The Forest

Neke od razlika između The Forest i igre u ovom radu su te da u The Forest igrač može izgraditi sklonište i razne stvari za preživljavanje, također ima element borbe protiv čudovišta koji napadaju skloništa od igrača.

Outlast je igra sa preživljavanjem za jednog igrača koju je izradio studio „Red Barrels“ u UE3 okruženju. U Outlast-u igrač igra kao novinar koji je došao istražiti napuštenu psihijatrijsku bolnicu. Igra se odvija u prvom licu gdje se igrač mora šuljati kako bih izbjegao neprijatelje jer nije moguće napasti neprijatelje, već se mora osloniti na skrivanje od neprijatelja u ormariće, ispod krevete i šunjajući se pokraj njih. Većina bolnice je mračna pa je jednini način da igrač vidi preko noćnog vida preko kamere, korištenje noćnog vida koristi bateriju pa igrač moram tražiti baterije po cijeloj bolnici.

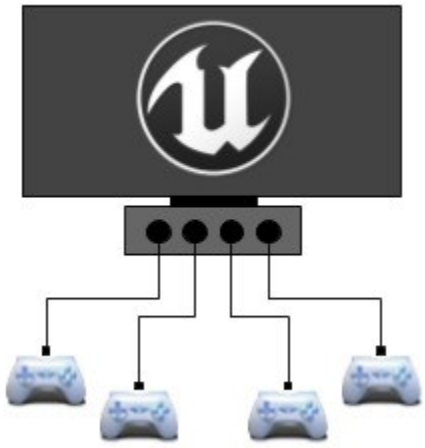
Outlast je igra koja se najviše oslanja na strašan ambijent, snažne zvukove i prepade(engl. Jumpscare), to je i ujedno najveća inspiracija kod igre u ovom radu. Na slici 2 prikazana snimka ekrana tijekom igranja igre Outlast.

Slika 2 Snimka ekrana iz igre Outlast

3. POSTAVLJANJE IGARA ZA VIŠE IGRAČA

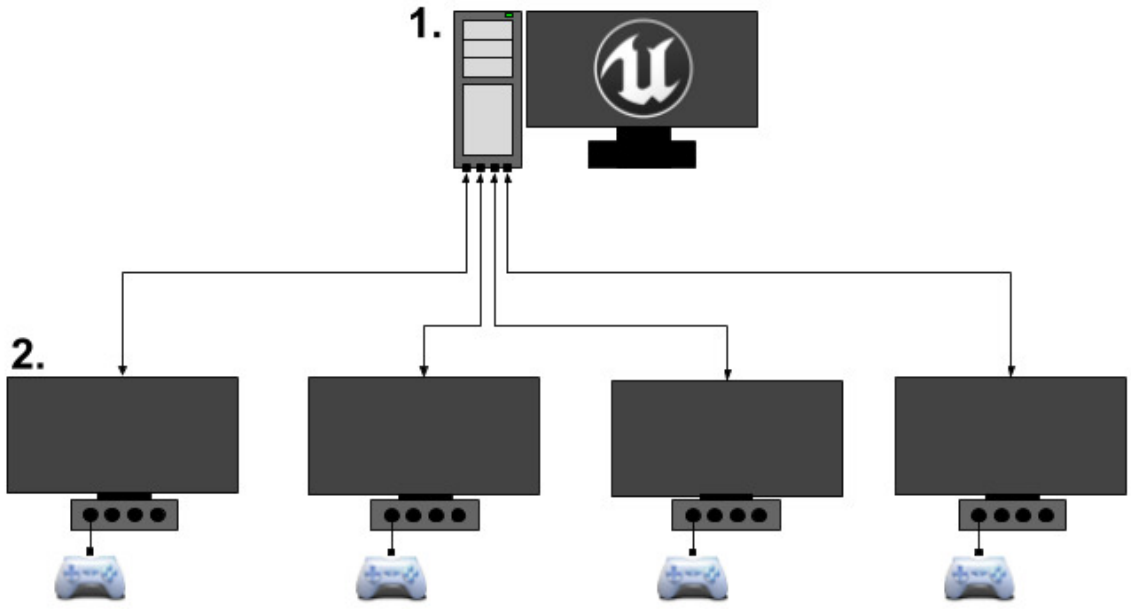
U sesiji za više igrača informacije se na nalaze samo na jednom računalu nego se o stanju igre komunicira između vise računala putem interneta. Iz tog je razloga programiranje za više igrača samo po sebi složenije od programiranja igre za jednog igrača jer je proces između igrača delikatan i uključuje mnoge dodatne korake. Kod planiranja izrade igre za više igrača važno je izgraditi temelje za igru s idejom da će igra biti za više igrača od samog početka. Može biti jako teško dodati način rada za više igrača u igru koja nije kreirana prema toj ideji od početka. U nekim slučajevima veliki dijelovi koda moraju biti potpuno iznova ispisani kako bi igra radila za više igrača.

U igri za jednog igrača ili lokanoj za više igrača, sesija je pokrenuta u samostalnoj igri. Igrači se povezuju na jedno računalo i izravno kontroliraju sve na njemu i sve u igri uključujući aktere, svijet i korisničko sučelje za svakog igrača. Takva igra postoji na lokalnom računalu kao što je prikazano na slici 3. (Networking Overview, n.d).



Slika 3. Prikaz igranja na lokalnoj mreži

Kada je u pitanju mrežna igra za više igrača „UE“ koristi model klijent-poslužitelj(engl. Server). Jedno računalo na mreži djeluje kao poslužitelj i stvara sesiju za više igrača kao što je prikazano na slici 4, dok se računala svih ostalih igrača povezuju kao klijenti. Poslužitelj zatim dijeli informacije o stanju igre sa svakim povezanim klijentom i pruža im mogućnost međusobne komunikacije. Poslužitelj, kao ugostitelj(engl. Host) igre drži jedno pravo autoritativno stanje igre. Drugim riječima, poslužitelj je mjesto na kojem se zapravo odvija igra za više igrača. Svaki klijent daljinski upravlja sa akterom kojeg posjeduju na strani poslužitelja, šaljući im signale procedura kako bi ih natjerao da izvrše radnje u igri. Međutim, poslužitelj ne prenosi vizualne sadržaje izravno na monitore klijenata. Umjesto toga, poslužitelj replicira informacije o stanju igre na svakom klijentu, govoreći im koji bi akteri trebali postojati, kako bi se ti akteri trebali ponašati i koje bi vrijednosti trebali imati različite varijable. Svaki klijent zatim koristi te informacije za simulaciju vrlo bliske aproksimacije onoga što se događa na poslužitelju (Networking Overview, n.d).



Slika 4. Poslužitelj obrađuje igru, a klijenti prokazuju igru korisnicima

4. TEMELJNI KONCEPTI UMREŽAVANJA

Mrežni način rada opisuje odnos računala i mrežne sesije za više igrača. Instanca igre može biti bilo koji od sljedećih mrežnih načina rada (Carnall, 2016):

Samostalan – igra radi kao poslužitelj koji ne prihvaća veze od udaljenih klijenata. Svi igraču koji sudjeluju u igri su isključivo lokalni igrači. Ovaj način se koristi za igre s jednim igračem i lokane igre za više igrača. Pokreće logiku na strani poslužitelja i logiku na strani klijenta prema potrebi za lokalne igrače.

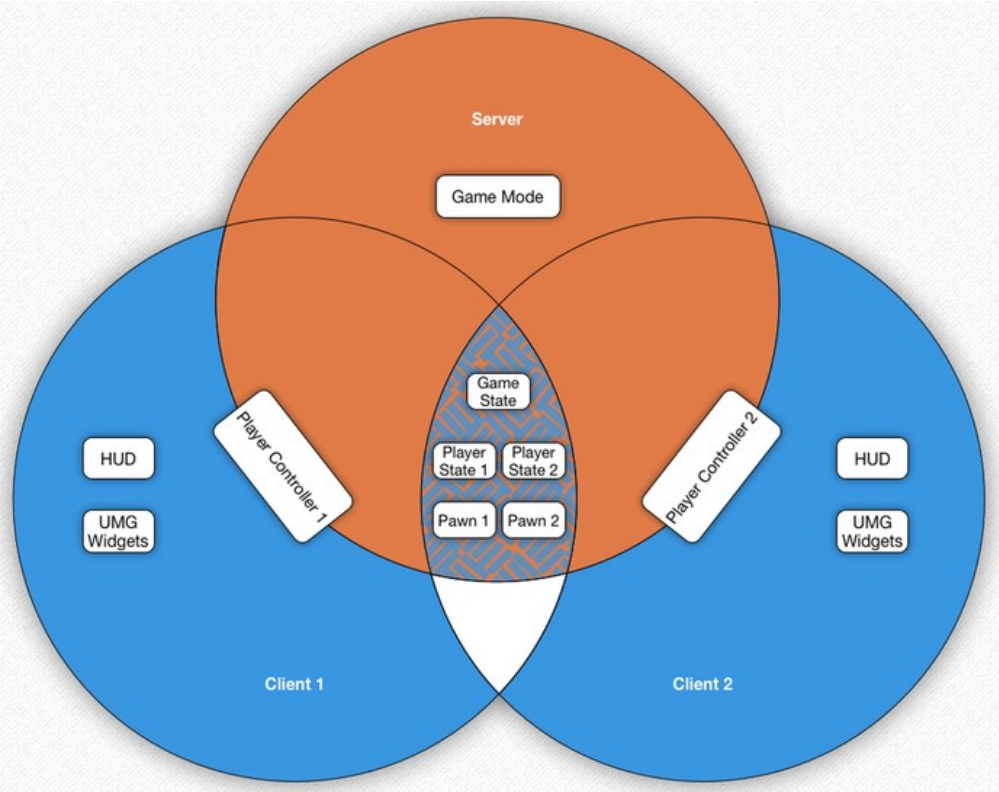
Klijent – igra radi kao klijent koji je povezan s poslužiteljem u mrežnoj sesiji za više igrača. Neće pokrenuti logiku na strani poslužitelja.

Klijent poslužitelj (engl. Listen server) – igra radi kao poslužitelj koji otvara mrežnu sesiju za više igrača. Prihvaća veze s udaljenih klijenata i ima lokane igrače izravno na poslužitelju. Ovaj način se često koristi za normalan i natjecateljski način igranja za više igrača.

Namjenski poslužitelj (engl. Dedicated server) – igra radi kao poslužitelj koji otvara mrežnu sesiju za više igrača. Prihvaća veze od udaljenih klijenata, ali nema lokane igrače, tako da odbacuje grafiku, zvuk, unos i druge značajke orijentirane na igrače kako bih radio učinkovitije. Ovaj se način rada često koristi za igre koje zahtijevaju ustrajniji, sigurniji ili masovniji način igranja.

Klijent poslužitelji su jednostavni za postaviti budući da svaki korisnik s kopijom igre može pokrenuti klijent poslužitelj i igrati na istom računalu. Igre koje koriste ovakav način umrežavanja često imaju korisničko sučelje u igri za pokretanje poslužitelja ili za traženje poslužitelja za spajanje. Međutim, budući da igrač koji drži klijent poslužitelj igra izravno na poslužitelju, on ima prednost u odnosu na druge igrače koji moraju koristiti mrežnu vezu za igranje, što izaziva zabrinutost oko pravednosti i varanja. Postoji i dodatno opterećenje obrade povezano s radom kao poslužitelj i održavanjem sustava relevantnih za igrače poput grafike i zvuka. Ovi čimbenici čine klijent poslužitelja manje prikladnim za igre u visoko konkurentnim okruženjima ili igre s vrlo velikim mrežnim opterećenjem, ali su vrlo prikladni za ležernu kooperativnu i natjecateljsku igru s više igrača između malog broja igrača.

Namjenski poslužitelji su skuplji i teže ih je konfigurirati, zahtijevaju odvojeno računalo od svih igrača koji sudjeluju u igri zajedno s vlastitom mrežnom vezom. Svi igrači koji se pridruže namjenskom poslužitelju doživljavaju igru s istom vrstom veze što osigurava pravednost. Budući da namjenski poslužitelj ne prikazuje grafiku niti izvodi bilo koju drugu logiku koja bi bila relevantna samo za lokalnog igrača on također mnogo učinkovitije upravljati igranjem i umrežavanjem. Zbog toga su namjenski poslužitelji poželjniji za igre koje zahtijevaju veliki broj igrača ili igre koje zahtijevaju pouzdan poslužitelj visokih performansi. Te igre uključuju žanrove kao što su „MMO“, natjecateljske „MOBA“ igre ili brze online pucačine. Na slici 5 prikazan je primjer kako namjenski poslužitelj funkcionira s dva klijenta i koje se funkcije obavljaju na poslužitelju, a koje na klijentu.



Slika 5. Primjer namjenskog poslužitelja s dva poslužitelja

5. REPLIKACIJA

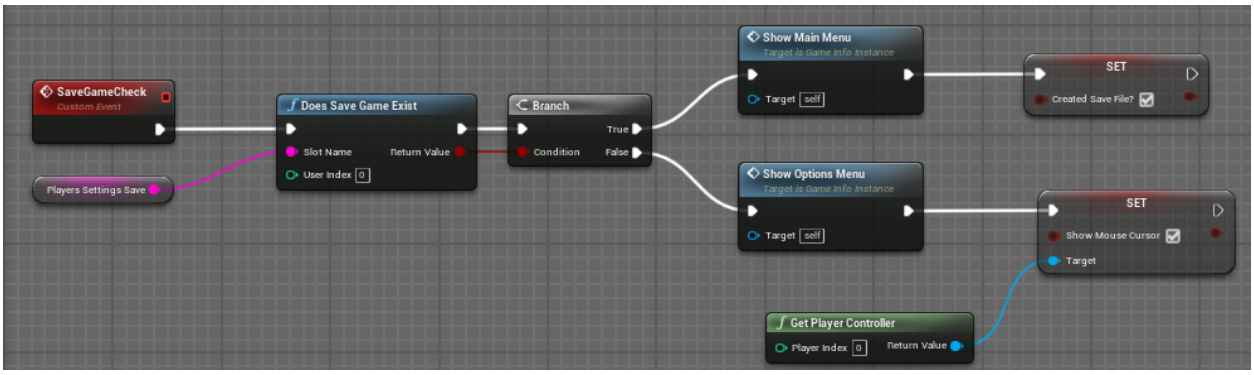
Replikacija je proces reprodukcije informacija o stanju igre između različitih računala u mrežnoj sesiji. Kada je replikacija ispravno konfigurirana, instance igre na različitim računalima postaju sinkronizirane. Prema zadanim postavkama većina aktera nema omogućenu replikaciju i sve će svoje funkcije obavljati lokano. Neke od najčešće korištenih značajki replikacije su (Carnall, 2016):

- **Stvaranje i uništavanje** – kada je autoritativna verzija repliciranog aktera stvorena na poslužitelju, on automatski generira svoje udaljene zastupnike na svim povezanim klijentima. Zatim će replicirati informacije tim udaljenim zastupnicima. Ako se uništi autoritativni akter, on će automatski uništiti svoje udaljene zastupnike na svim povezanim klijentima.
- **Replikacija kretanja** – ako autoritativni akter ima omogućenu replikaciju kretanja, automatski će replicirati svoju lokaciju, rotaciju i brzinu.
- **Replikacija varijabli** – sve varijable koje su označene kao repliciranje automatski će se replicirati od autoritativnog aktera do njegovih udaljenih zastupnika kad god se njihove vrijednosti promijene.
- **Udaljeni pozivi procedura (engl. Remote procedure calls, RPCs)** – „RPC“-ovi su posebne funkcije koje se prenose određenim računalima u mrežnoj igri. Bez obzira na kojem se računalu „RPC“ inicijalno poziva, njegova implementacija će se izvoditi samo na računalu za koje je on namijenjen. Oni mogu biti označeni kao poslužitelj (radi samo na poslužitelju), klijent (radi samo na klijentu koji posjeduje aktera) ili „Multicast“ (radi na svakom računalu povezanom na sesiji, uključujući poslužitelja).
- **Replikacija komponenti** - komponente aktera repliciraju se kao dio aktera koji ih posjeduje. Sve varijable unutra komponente koje su označene kao replicirane će se replicirati, a svi „RPC“-ovi povezani unutar komponente ponašati će se dosljedno „RPC“-ovima povezanim u klasi aktera.

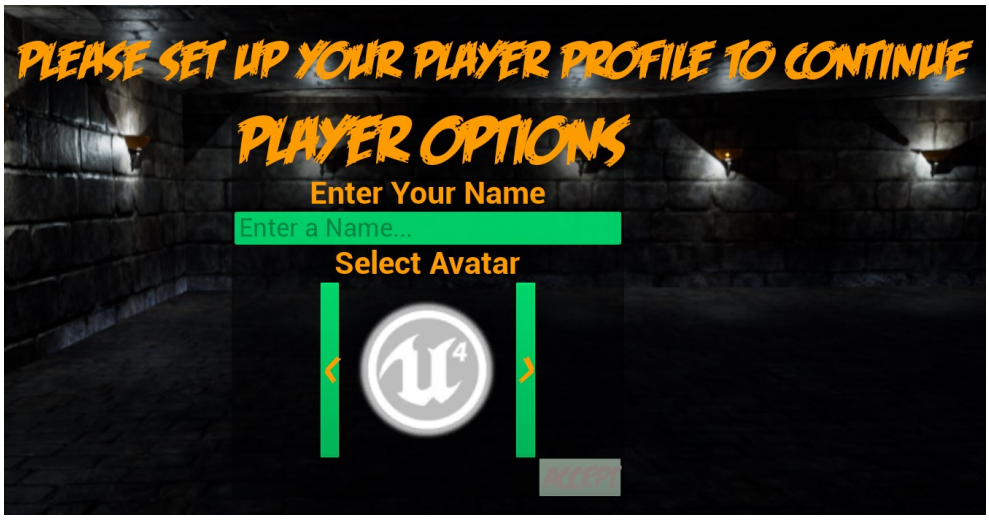
6. STVARANJE IGRE

6.1 Stvaranje instance o igri

Kada igrač ulazi u igru instanca igre provjerava dali igrač ima spremljene podatke, ako nema ništa spremljeno otvaraju se prvo opcije postavki igrača koji traži od igrača da unese podatke kao što su korisničko ime i ikona koje želi koristiti kao što je prikazano na slikama 6 i 7.

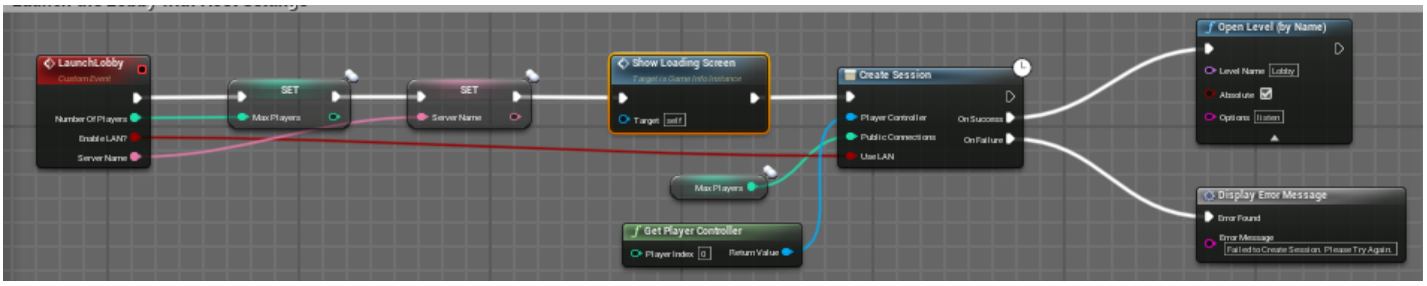


Slika 6. Provjera dali imaju spremljeni podaci o igri



Slika 7. Unos korisničkog imena i ikone

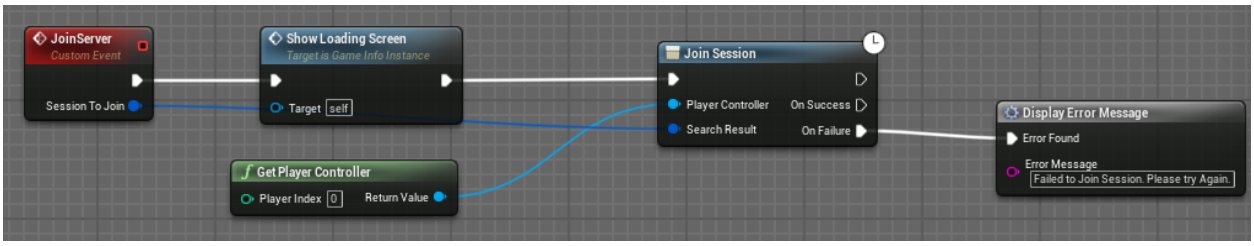
Slike 8 i 9 prikazuju funkciju koja otvara lobi s postavkama koje je igrač odredio a to su broj igrača, ime poslužitelja i da li je želi igrati preko lokalne mreže ili preko interneta. Broj igrača i naziv poslužitelja moraju biti replicirane kako bih informacija o poslužitelju bili prikazane isto svim igračima.



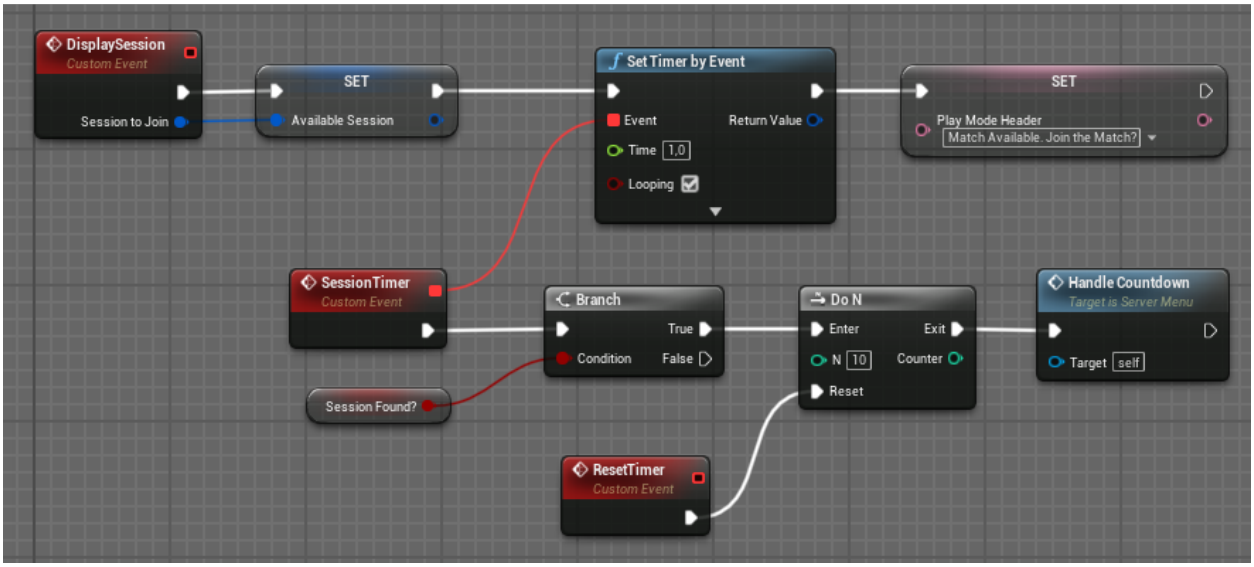
Slika 8 Otvaranje lobija sa odabranim postavkama

Slika 9 Otvoren lobi sa odabranim postavkama

Za ulazak u poslužitelj, klijent prvo provjerava dali ima dostupnih poslužitelja, ako ima dostupnih poslužitelja prikazuje se obavijest da ima dostupnih poslužitelja i pitanje da li se želi pridružiti sesiji s tajmerom od 10 sekundi da donese odluku. Na slikama 10, 11 i 12 prikazan je programski kod i mehanika ulaska na poslužitelj.



Slika 10. Ulaz u poslužitelj - pozvan i glavnog izbornika

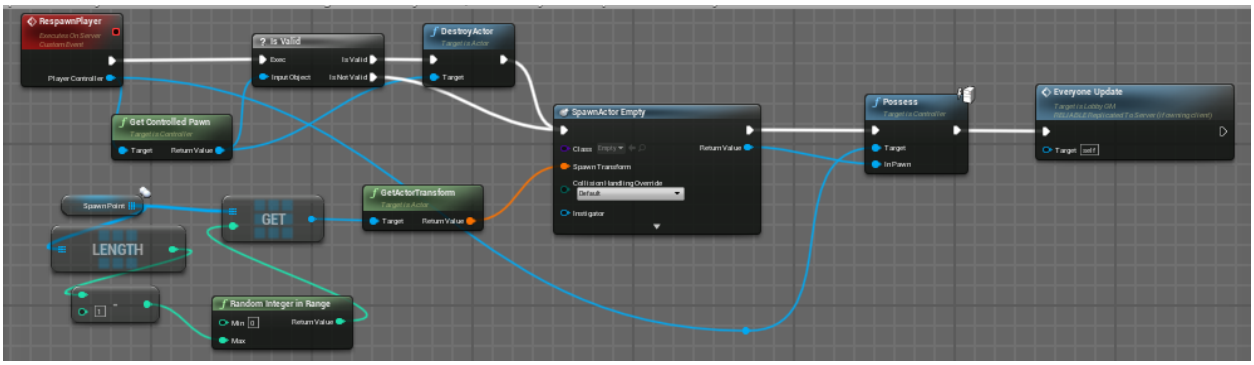


Slika 11. poslužitelj je pronađen - prikazuje odbrojavanje za ulazak

Slika 12. Prikaz obavijesti da je poslužitelj pronađen

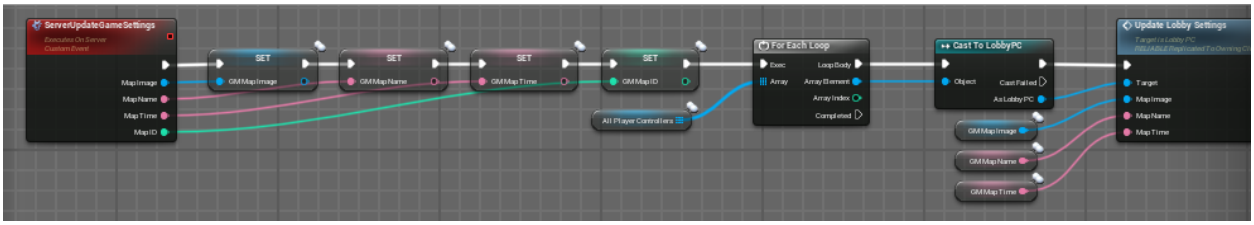
6.2 Lobi način igre(engl. Game mode) i lobi kontroler igrača(engl. Player controller)

Na slici 13 prikazano je stvaranje igrača u lobiju na jednoj od dostupnih lokacija za stvaranje i govori svim klijentima da ažuriraju informacija igrača pri ulasku u lobi. Ova funkcija se poziva na poslužitelju.

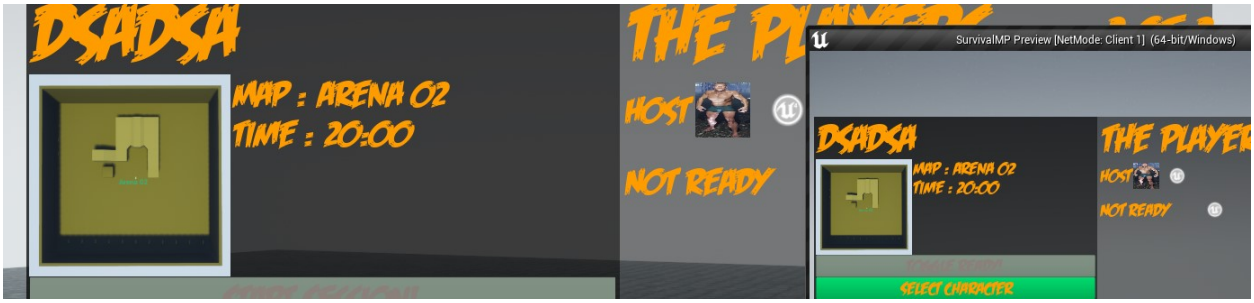


Slika 13. Stvaranje i ažuriranje igrača

Nakon što poslužitelj promijeni postavke kao mapu za igranje ili vrijeme igranja poslužitelj govori svim klijentima da ažuriraju informaciju o lobiju kao što je prikazano na slici 14 i 15. Ova funkcija se također poziva na poslužitelju.

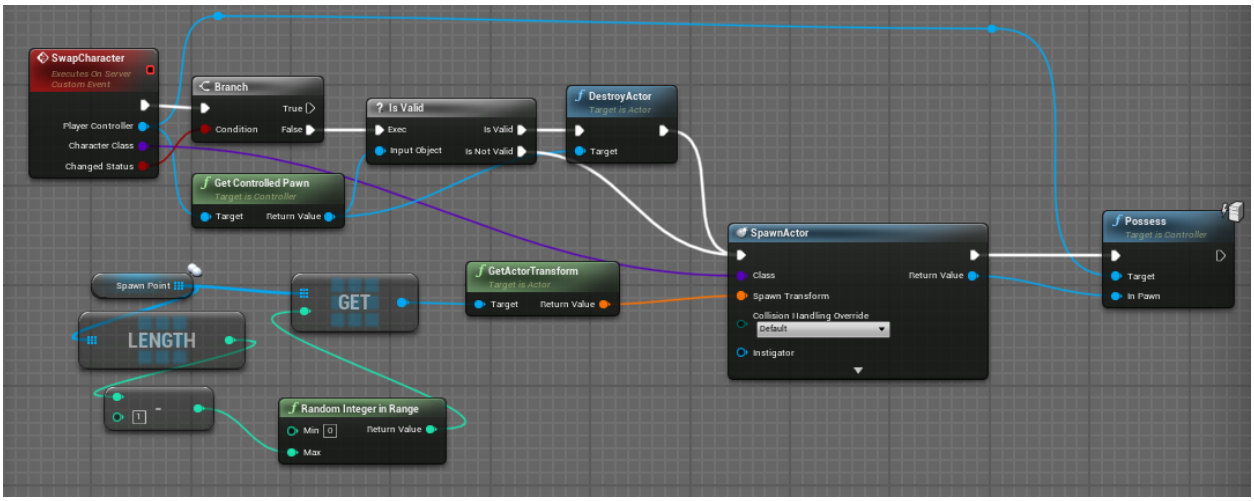


Slika 14. Poslužitelj govori svim klijentima da ažuriraju informacije o lobiju

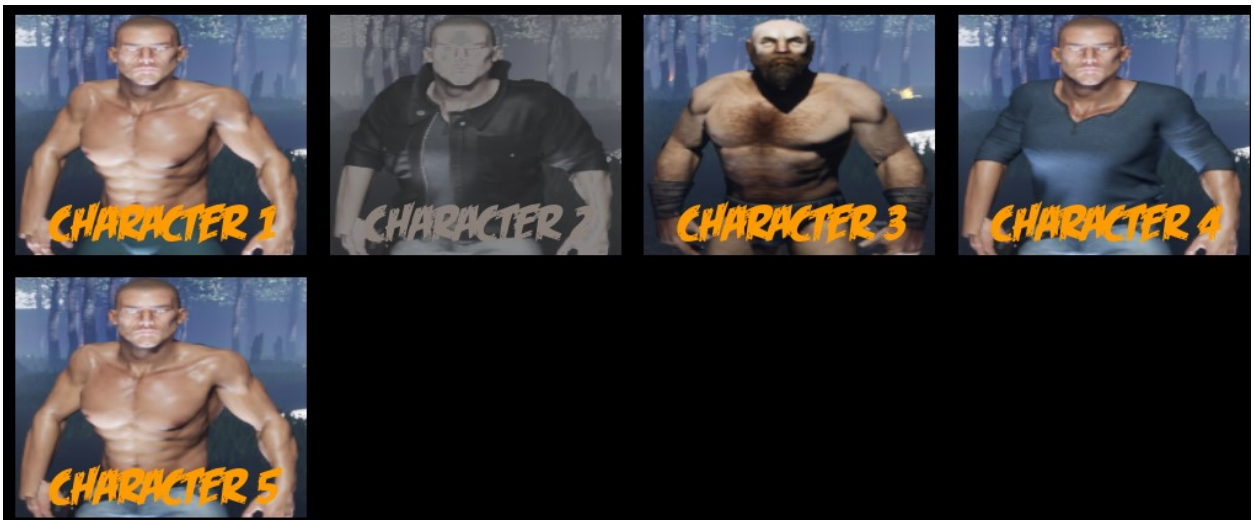


Slika 15. Prikaz promijene i ažuriranja postavki

Kod odabira lika za igranje prvo se provjerava da li već postoji kontorlirani akter, ako postoji onda se on uništava i stvara se lik koji je klijent odabrao i klijent ga zatim posjeduje kao što je prikazano na slici 16 i 17.

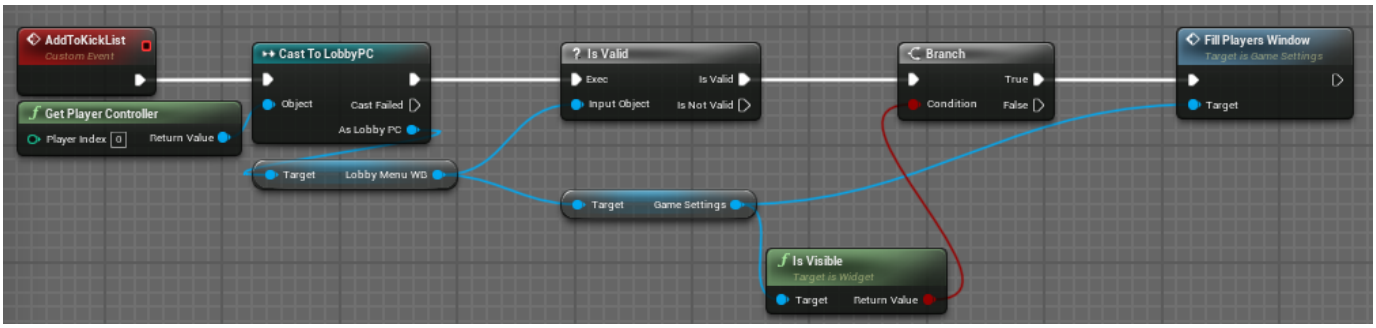


Slika 16. Zamjena trenutnog lik sa odabranim likom sa ekrana za odabir likova

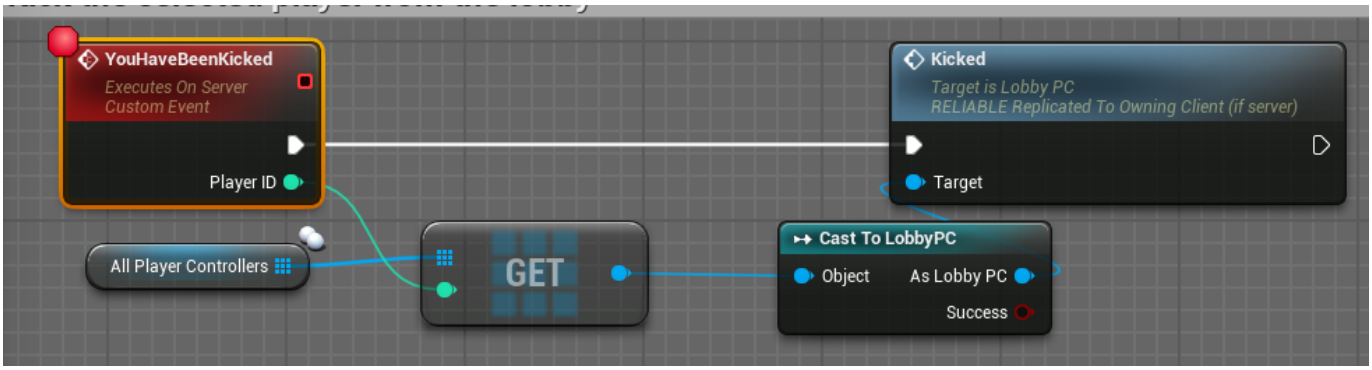


Slika 17. Prikaz zamijene likova

Slika 18 prikazuje funkciju koja dodaje spojenog igrača na listu za izbacivanje ako su otvorene opcije za igru. Poslužitelj može otvoriti listu za izbacivanje i izabciti igrača iz lobija kao što je prikazano na slici 19.

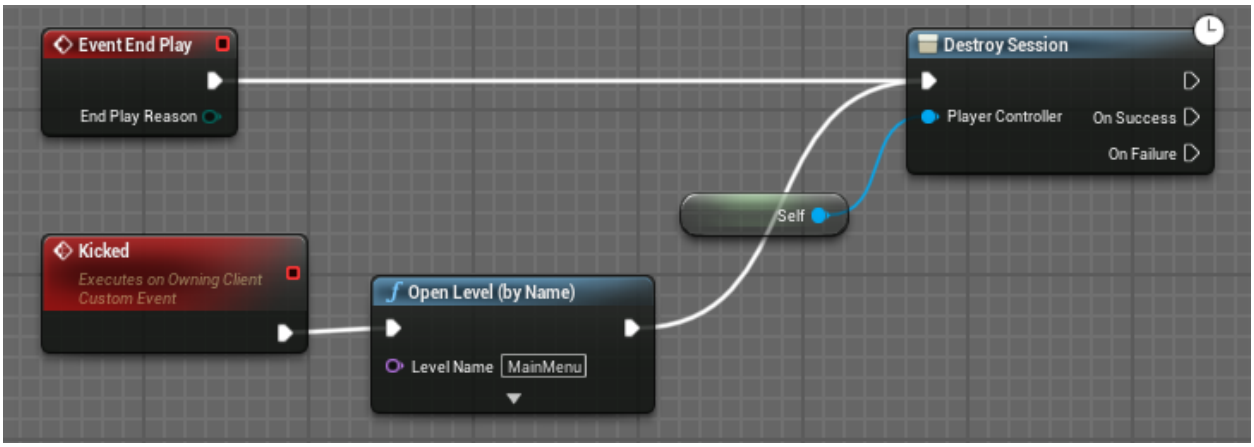


Slika 18. Dodavanje igrača na listu za izbacivanje



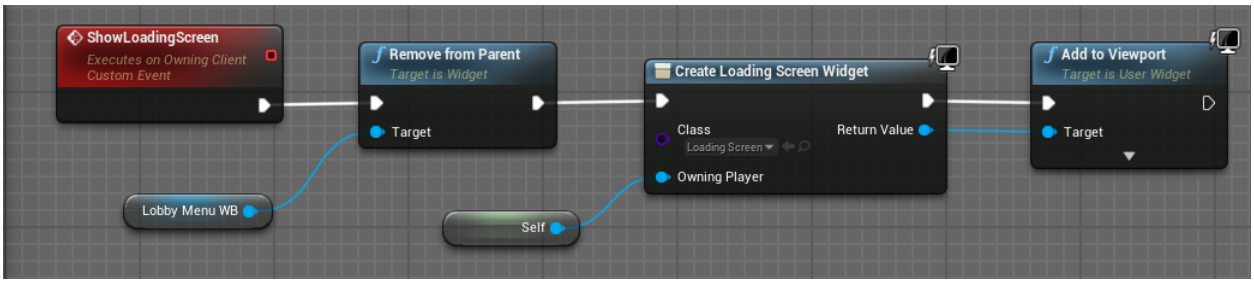
Slika 19. Izbacuje odabranog igrača iz lobija

Izbacivanje igrača se poziva preko lobi „Player Controller“ – a i poziva se preko posjedovnog klijenta koji uništava sesiju i otvara glavni izbornik kao što je prikazano na slici 20.



Slika 20. Izbacivanje igrača preko posjedovnog klijenta

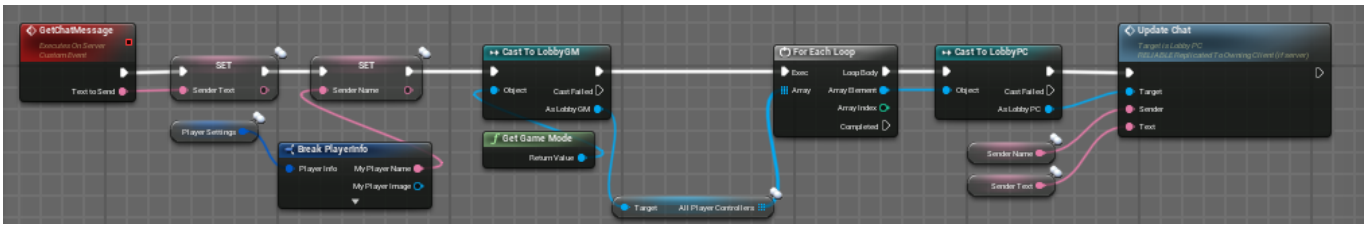
Nakon što igrač napusti lobi poziva se funkcija koja uklanja sučelje lobi izbornika i stvara prikaz za učitavanje mape kao što je prikazano na slici 21 i 22.



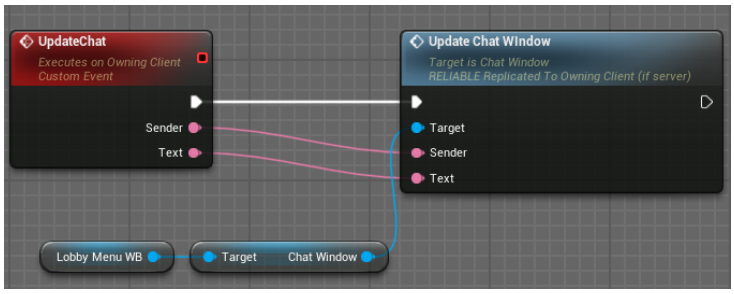
Slika 21. Izbacivanje lobi izbornika i stvaranje prikaza učitavanja

Slika 22. Učitavanje

Funkcija koja je prikazana na slici 23 procesira nadolazeće poruke u chatu i govori svim igračima da izvedu funkciju koja ažurira chat. Funkcija koja ažurira chat izvodi se na posjedovnom klijentu kao što je prikazano na slici 24, dok se funkcija koja procesira nadolazeće poruke u chatu izvodi na poslužitelju.

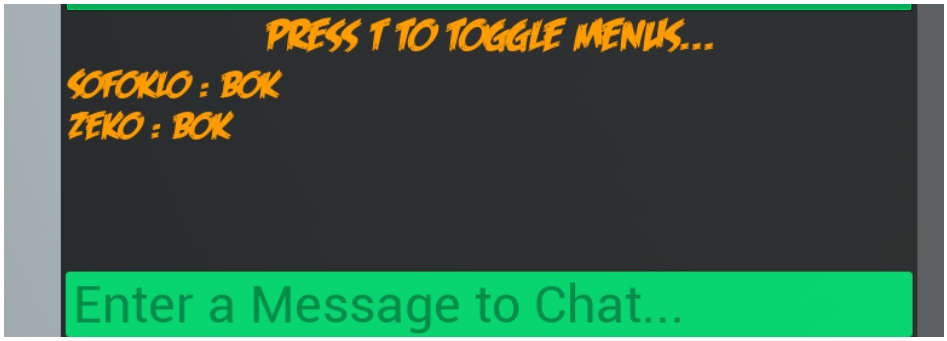


Slika 23. Procesiranje poruka u chatu



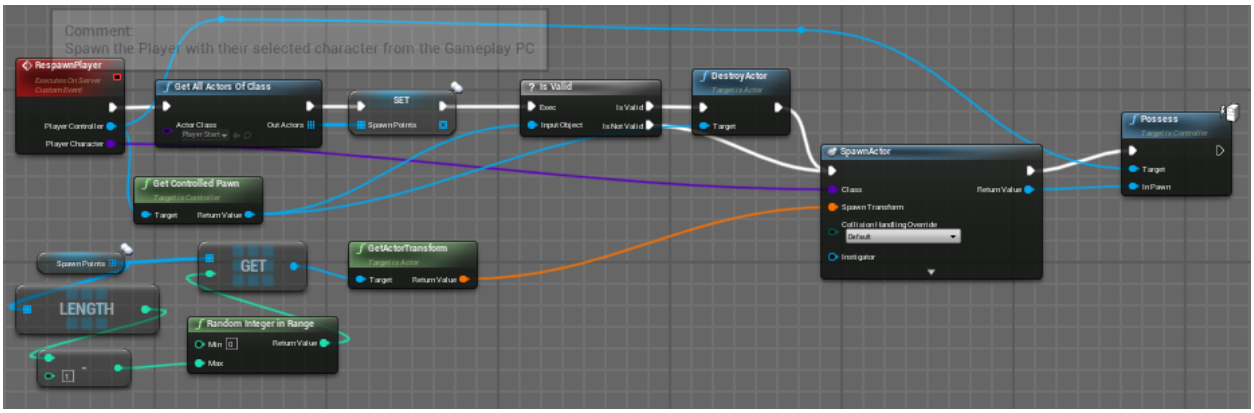
Slika 24. Ažuriranje prikaza chata

Na slici 25 prikazana je komunikacija između dva igrača.



Slika 25 Komunikacije između dva igrača

Slika 26 prikazuje funkciju gdje poslužitelj poziva stvaranje igrača sa odabranim likom iz lobija u glavnu mapu. Igrač se može stvoriti na jednoj od određenih lokacija za stvaranje, poslužitelj odlučuje nasumično koja će to lokacija biti, a slika 27 prikazuje stvorenog igrača na jednoj od lokacija.



Slika 26. Funkcija stvaranja igrača

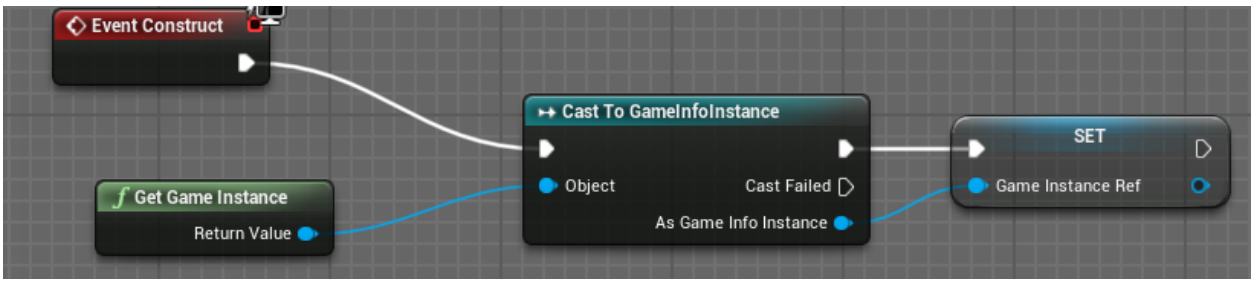
Slika 27. Prikaz stvorenog igrača

6.3 Korisničko sučelje

Glavni izbornik sastoji se od opcija za ugošćivanje igre, pronalazak igre, postavki i izlaza iz igre kao što je prikazano na slici 28.

Slika 28. Glavni izbornik - izgled

Slika 29 prikazuje uzimanje reference na instancu informacija o igri nakon što je sučelje kreirano.



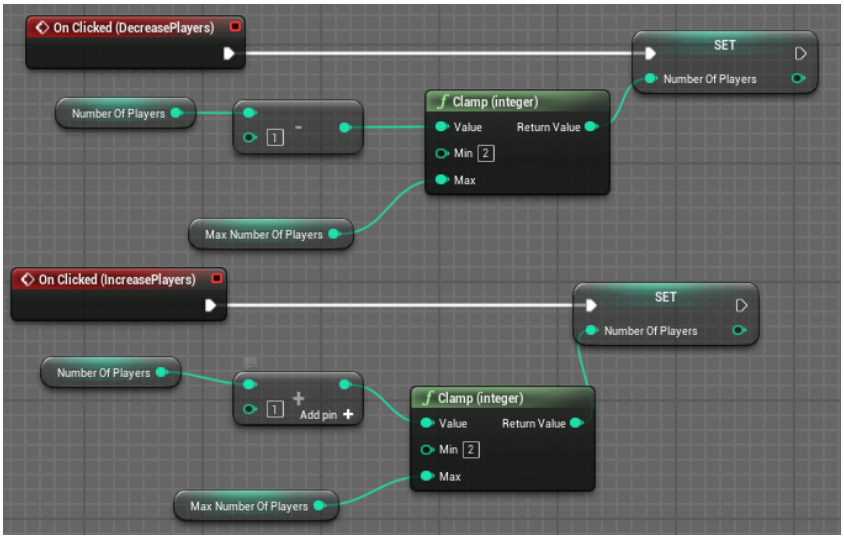
Slika 29. Referenca na instancu informacija o igri

Pritiskom na opciju za ugošćivanje izvodi se funkcija koja uklanja prikaz glavnog izbornika i prikazuje izbornik za ugošćivanje kao što je prikazano na slici 30.

Slika 30. Uzima i prikazuje izbornik za ugošćivanje igre, na ovaj način funkcioniraju i ostale opcije u glavnom izborniku

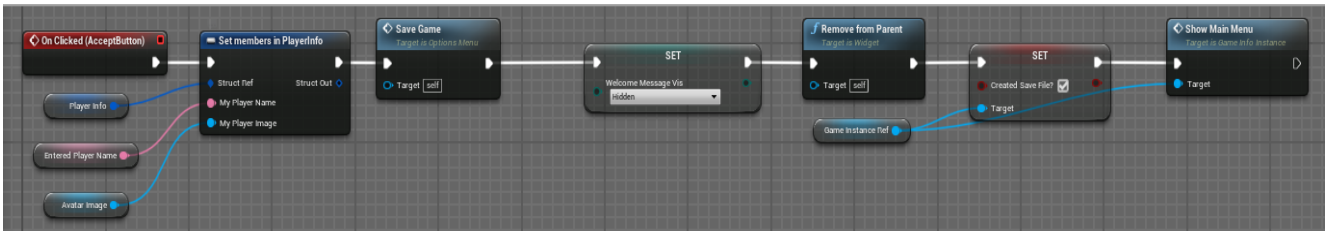
Funkcija za određivanje broja igrača gdje je minimalna vrijednost 2 igrača, a maksimalna vrijednost 8 igrača prikazana je na slici 31 i 32.

Slika 31. Izgled izbornika za ugošćivanje igre



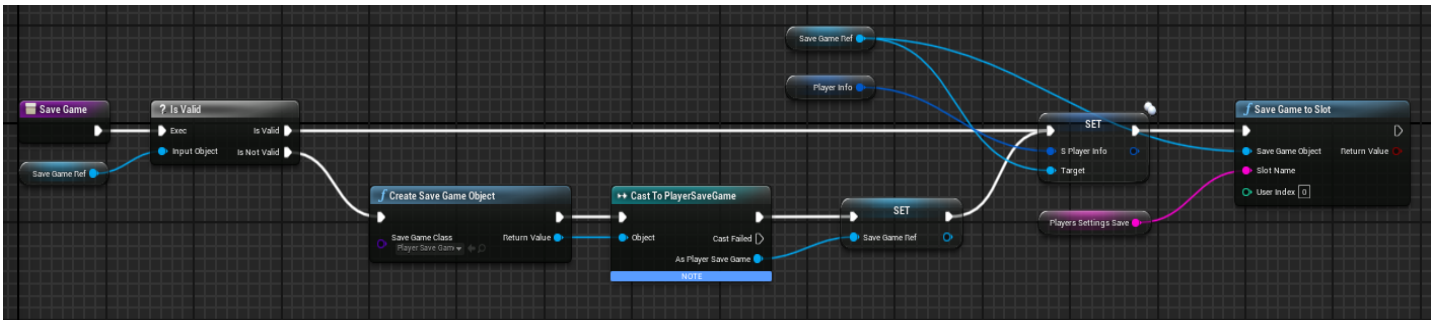
Slika 32. Povećavanje i smanjivanje broja igrača

Slika 33 prikazuje funkciju koja prihvaća i primjenjuje promjene koje je igrač napravio i sprema ih u spremljene podatke o igraču.



Slika 33. Prihvaća i primjenjuje korisničke promjene na S Player Info

Funkcija za spremanje igre provjerava dali postoje spremljeni podaci o igraču, a ako ne postoje stvara se novi objekt za spremanje igre u kojem će biti spremljeni podaci o igraču, ako postoje spremljeni podaci onda se stvaranje novog objekta preskače kao što je prikazano na slici 34.



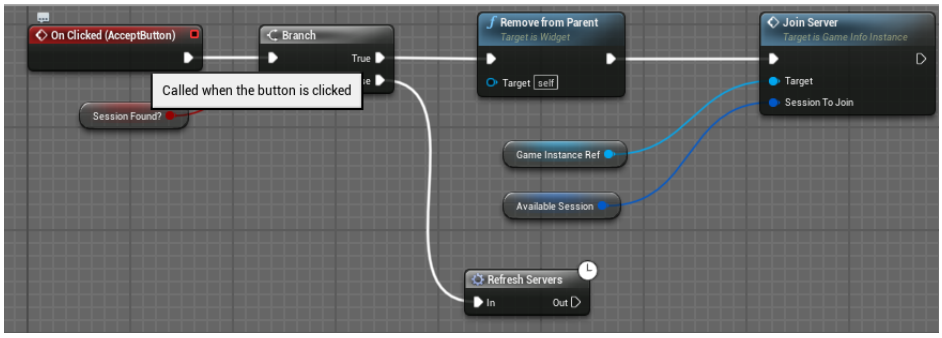
Slika 34. Spremanje igre

Slika 35 prikazuje opcije za unos imena i odabir avatara u postavkama za igrača.

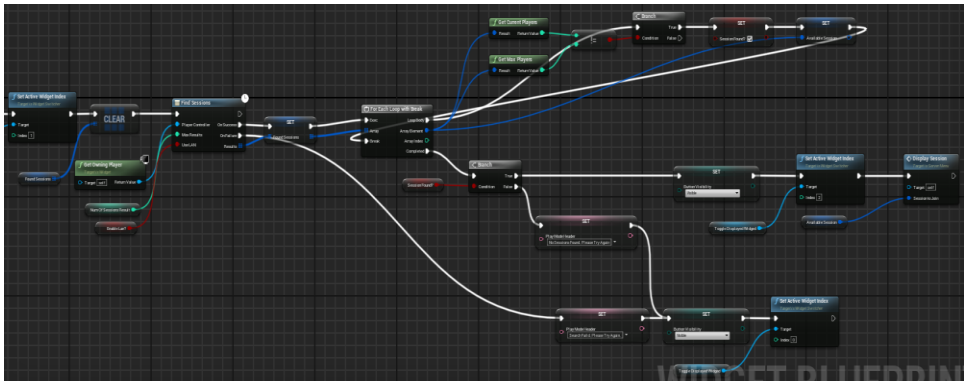
Slika 35. Postavke igrač

Pritiskom gumba za prihvaćanje pronalaska poslužitelja koji je prikazan na slici 36 poziva se funkcija koja pretražuje dali ima dostupnih poslužitelja kao što je prikazano na slici 37. Ako ima dostupnih poslužitelja igrač se pridružuje poslužitelju, a ako nema dostupnih poslužitelja poziva se funkcija koja konstantno provjerava dostupnost poslužitelja te je ta funkcija prikazana na slici 38.

Slika 36. Sučelje za pronalazak poslužitelja

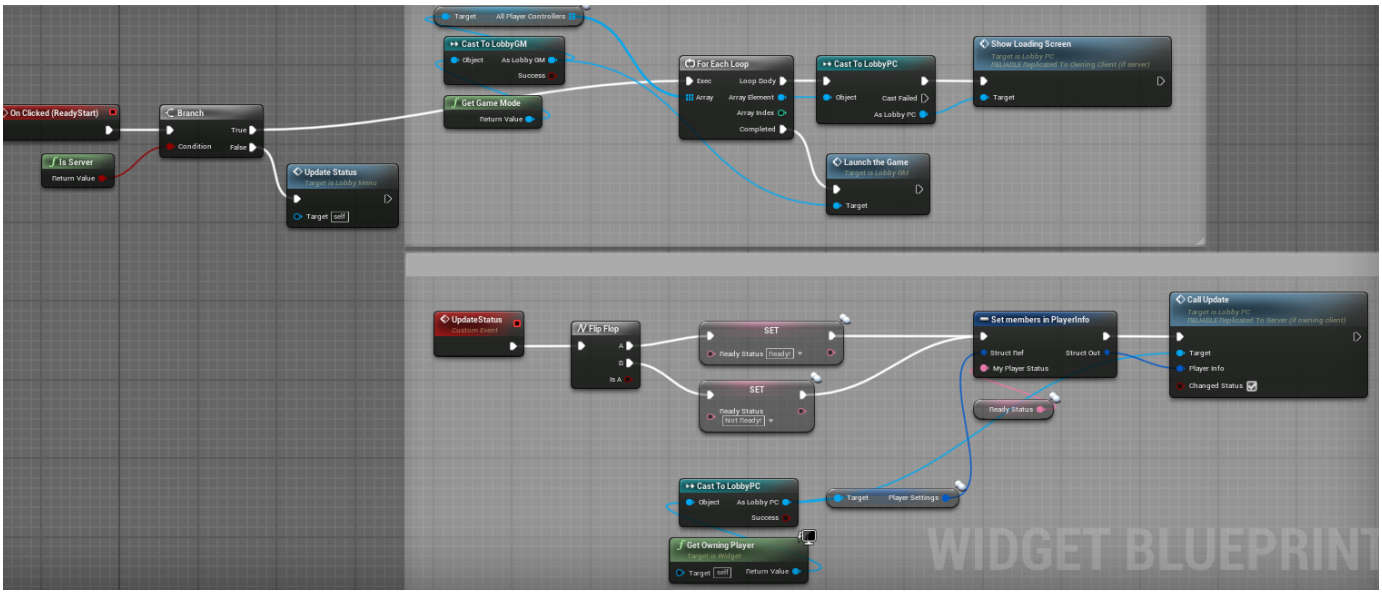


Slika 37. Gumb za prihvaćanje/pretraživanje i pridruživanje poslužitelju ako je dostupan



Slika 38. Provjera dostupnih poslužitelja

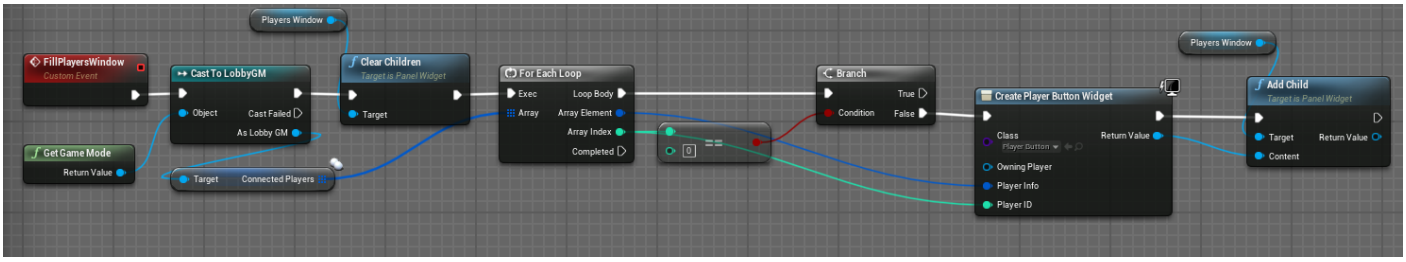
U lobi izborniku gumb za pokretanje igre dostupan je samo poslužitelju odnosno igraču koji ugošćuje lobi. Za klijenta taj gumb služi kako bi javio ostalim igračima da je spreman kao što je prikazano na slici 39 i 40.



Slika 39. Funkcija za pokretanje igre odnosno klijent javlja da je spreman

Slika 40. Izgled lobi izbornika za poslužitelja i za klijenta

Slika 41 prikazuje funkciju koja popunjava listu igrača kada je pozvana u postavkama igre.



Slika 41. Izgled izbornika za postavke igre

Funkcija prikazana na slici 42 izbacuje igrača iz lobija pritiskom na gumb s imenom igrača kojeg poslužitelj želi izbaciti. Slika 43 prikazuje izgled izbornika za izbacivanje igrača.

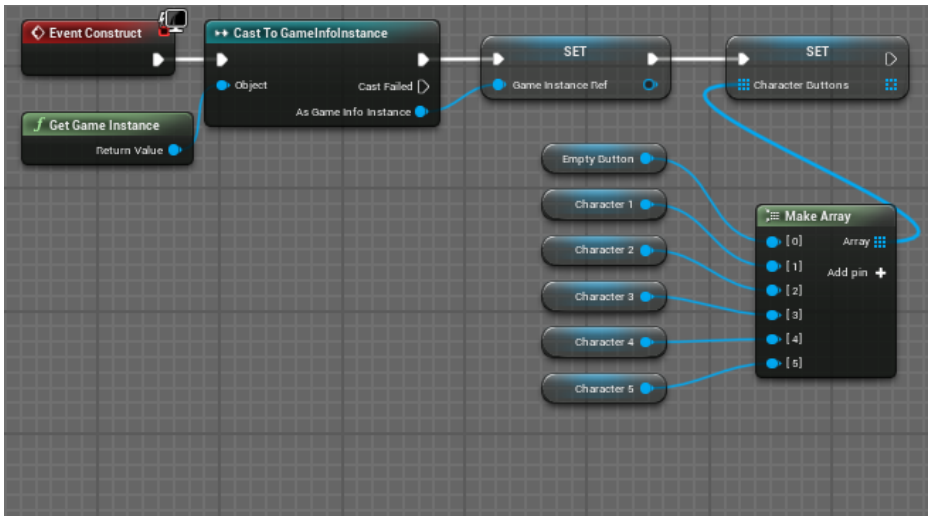
Slika 42. Izbacivanje igrača na pritisak gumba

Slika 43. Izbornik za izbacivanje igrača

Igrač pritiskom na sliku jednog od dostupnih likova odabire lika sa kojim želi igrati kao što je prikazano na slici 44.

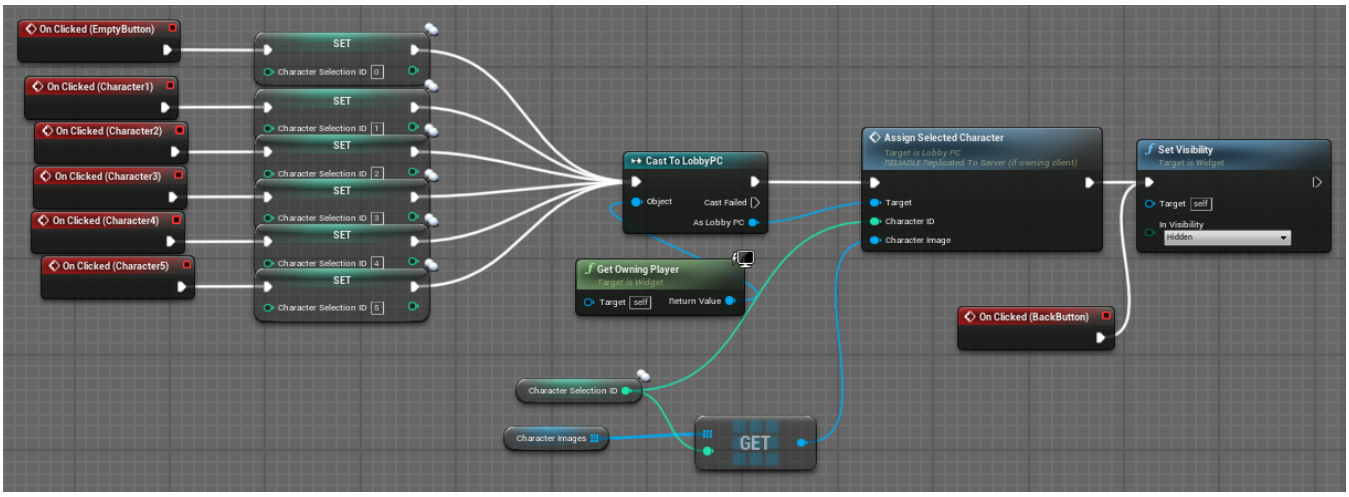
Slika 44. Odabir likova

Slika 45 prikazuje funkciju unutar sučelja za odabir likova koja uzima instancu igre i dodjeljuje gumbove za likove.



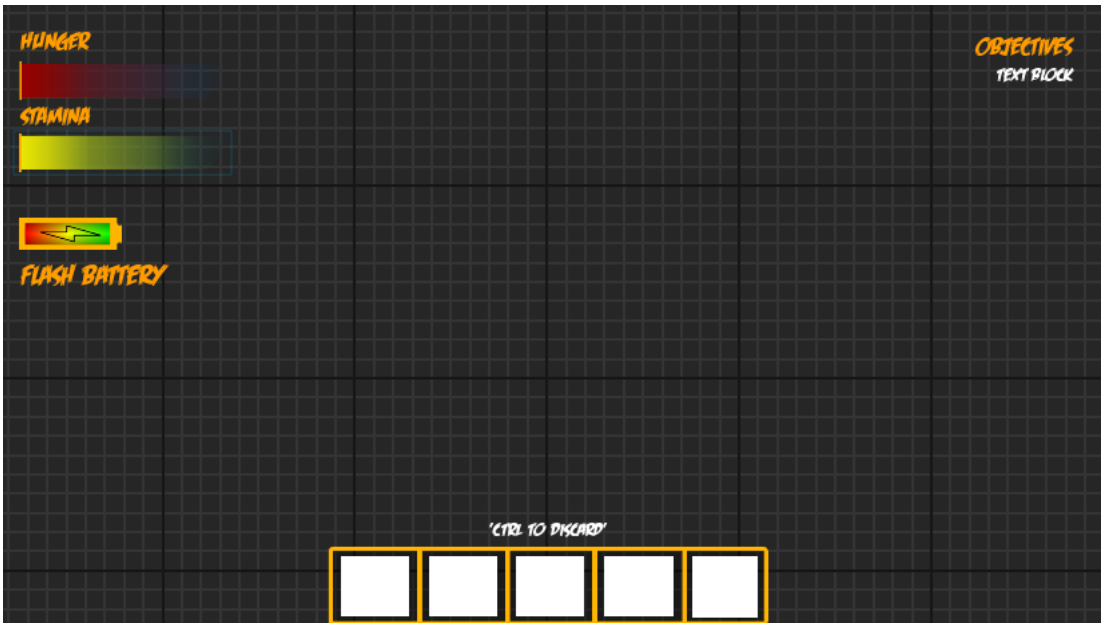
Slika 45. Uzima instancu igre/ dodjeljuje gumbove za likove

Funkcija prikazana na slici 46 dodjeljuje funkciju za odabir likova koja se poziva iz lobi kontrolera igrača i gumb odabranog lika postavlja nevidljivim.



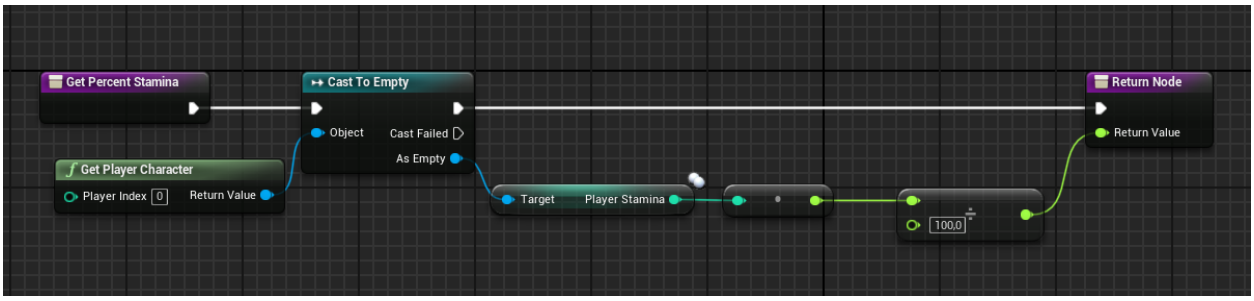
Slika 46. Pokušava dodijeliti odabranog lika iz izbornika za odabir likova

Slika 47 prikazuje sučelje igrača koje prati razne informacije o igraču.



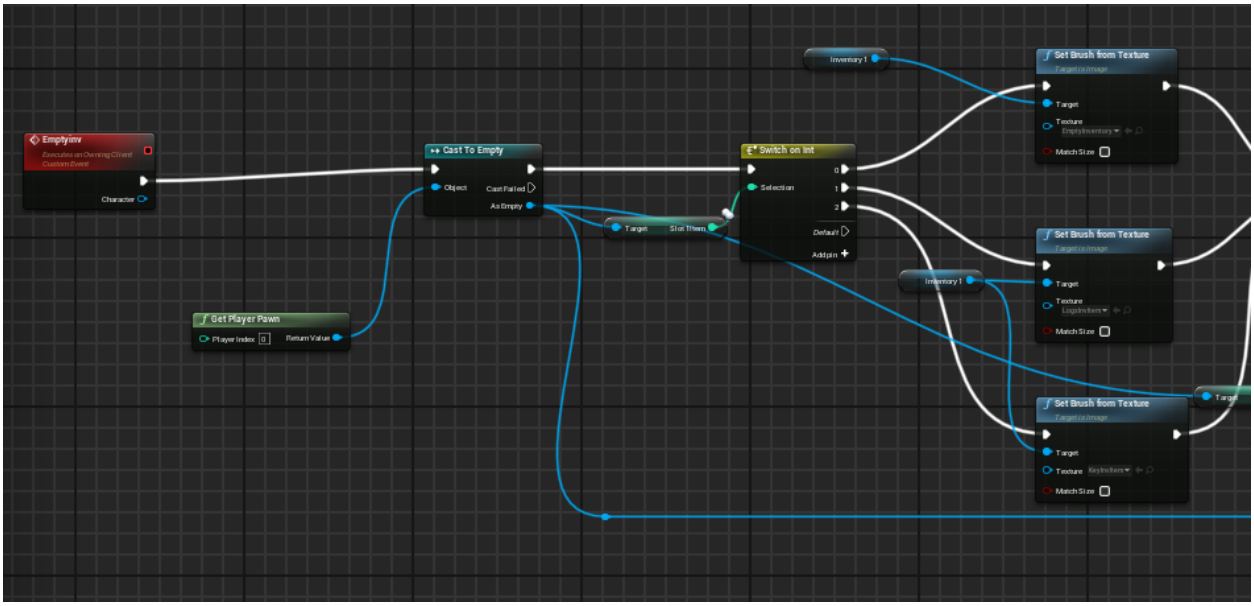
Slika 47. Sučelje igrača

Funkcije koje prikazuju glad, izdržljivost i bateriju za svjetiljku rade na isti način tako da uzimaju vrijednost jednog od tih podataka i pretvaraju ga u postotak kao što je prikazano na slici 48.



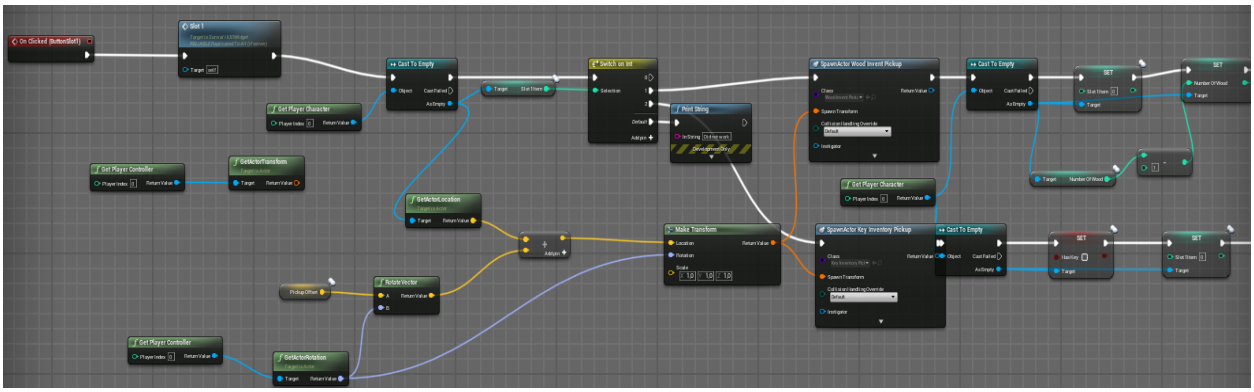
Slika 48. Funkcija za prikaz izdržljivosti

Svaki predmet koji igrač skupi ima svoju identifikaciju(ID) , kada igrač skupi neki predmet provjerava se koji je „ID“ tog predmeta te se zatim prikazuje ikona predmeta kojeg je igrač pokupio u inventaru, kao što je prikazano funkcijom na slici 49.



Slika 49. Prikaz ikona u inventaru

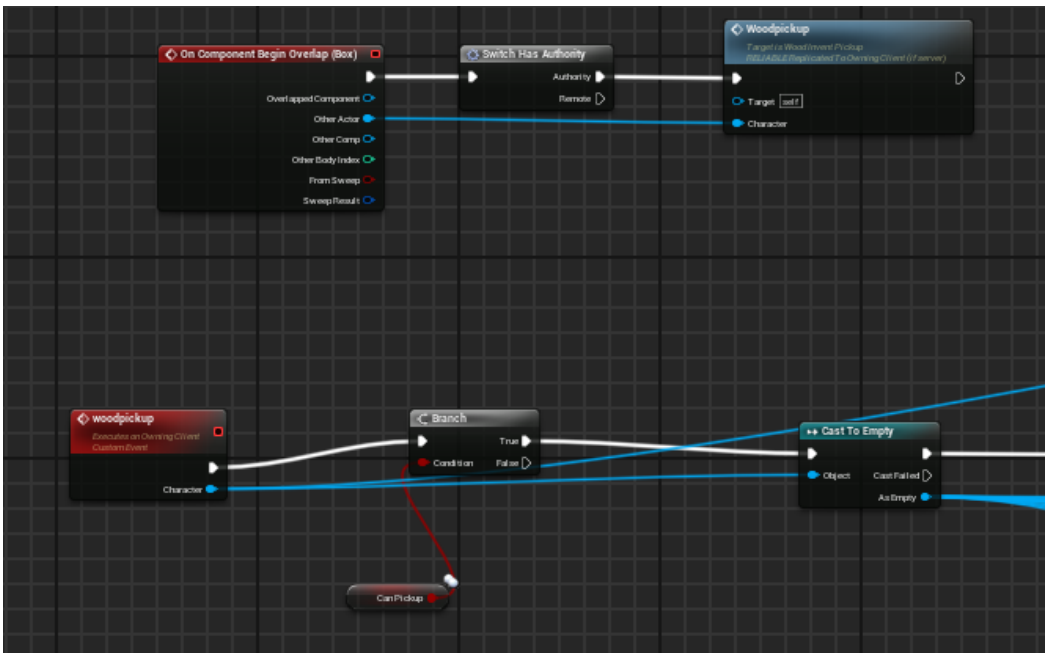
Izbacivanje predmeta funkcionira tako da igrač pritiskom na ikonu jednog od predmeta izbacuje taj predmet iz svog inventara i izbacuje ga ispred sebe kao što je prikazano na slici 50.



Slika 50. Funkcija za izbacivanje predmeta

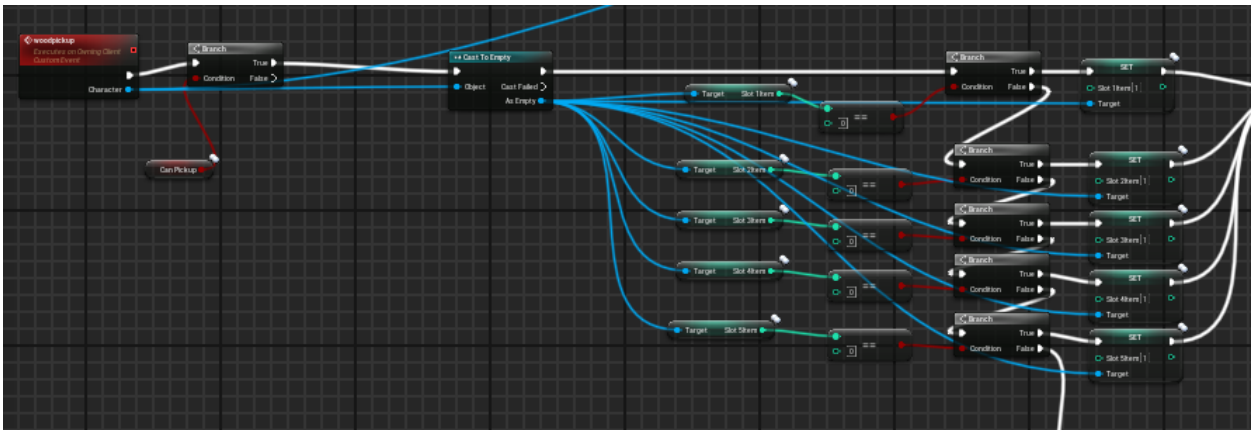
6.4 Akteri za interakciju sa igračima

Funkcija unutar aktera poziva se kada igrač prijeđe preko aktera unutar mape za igranje kao što je prikazano na slici 51.



Slika 51. Pozivanje funkcije nakon prijelaza aktera

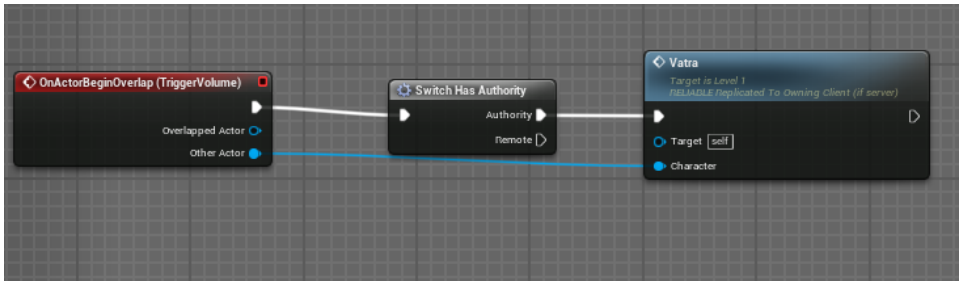
Slika 52 prikazuje dodavanje drva u inventar specifičnog igrača i praćenje broja drva koji on posjeduje, kada igrač skupi tri drveta dobiva obavijest da je obavio zadatak i dobiva sljedeći zadatak.



Slika 52. Funkcija za dodavanje drva u inventar

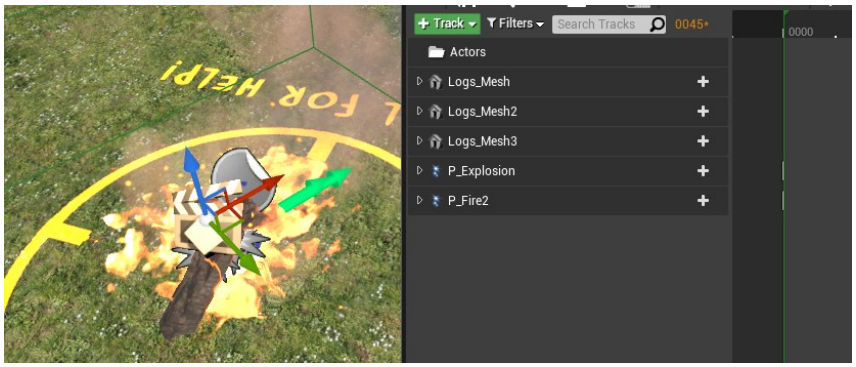
6.5 Događaji koji se aktiviraju direktno na mapi

Događaji koji se aktiviraju direktno na mapi su događaji koji imaju specifične uvijete kako bih se aktivirali. Jedan od tih događaja je paljenje vatre koji ima uvjet da igrač mora imati tri drveta kako bih ga aktivirao kao što je prikazano na slici 53.

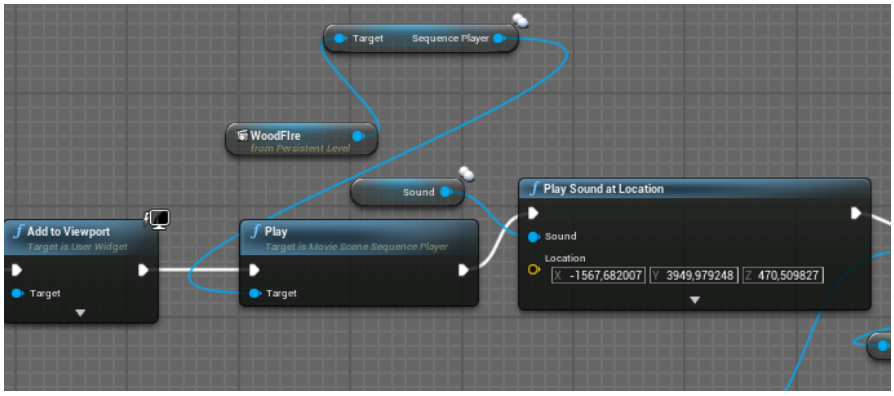


Slika 53. Kada akter aktivira ovaj događaj pokreče se sekvenca koja stvara vatru

Slika 54 prikazuje sekvencu koja izvodi stvaranje vatre i izvođenje zvuka na specifičnoj lokaciji kao što je prikazano na slici 55.



Slika 54. Prikaz sekvence

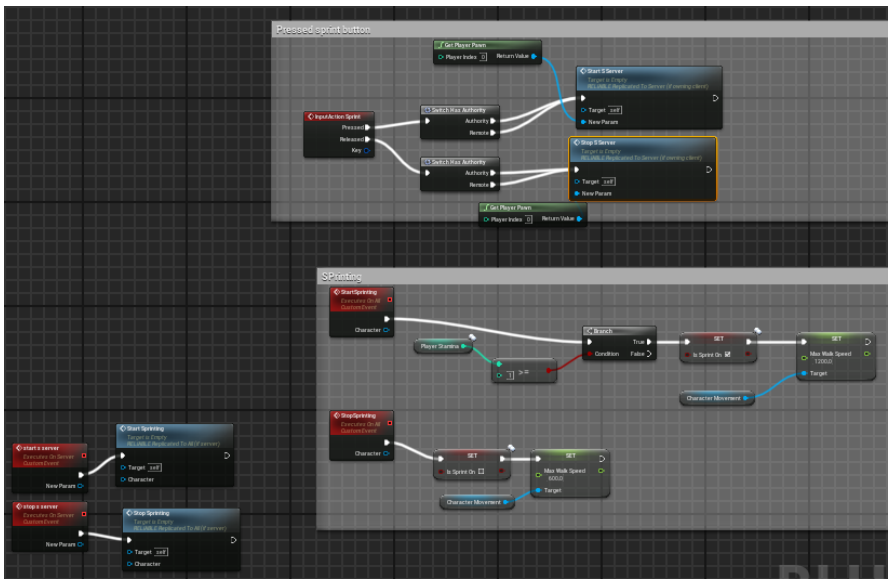


Slika 55. Pozivanje sekvence i zvuka na lokaciji

Sekvencer je kinematografski alat za stvaranje raznih animacija u sceni kao što je na primjer stvaranje vatre.

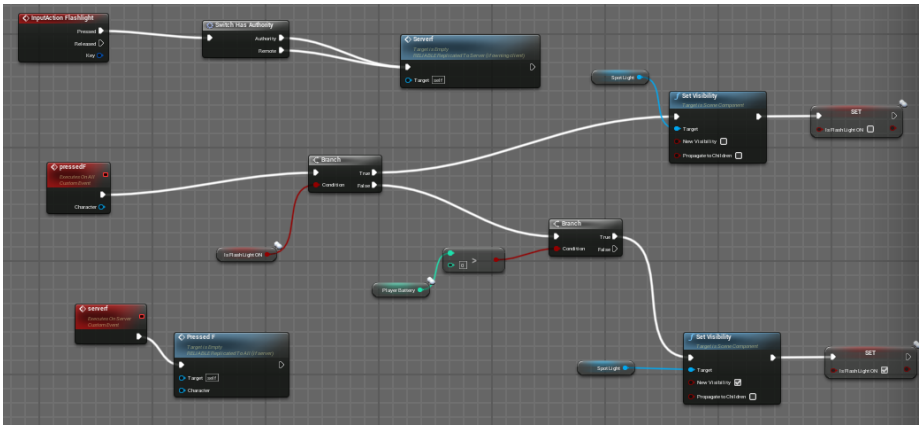
6.6 Glavni akter

U glavnom akteru nalaze se funkcije kao što su trčanje, uključivanje svjetiljke, smrt igrača i druge. Slika 56 prikazuje funkciju za trčanje koja se preko „multicast“-a poziva na poslužitelju kako klijent ne bih imao problema sa sinkronizacijom.



Slika 56. Funkcija za trčanje

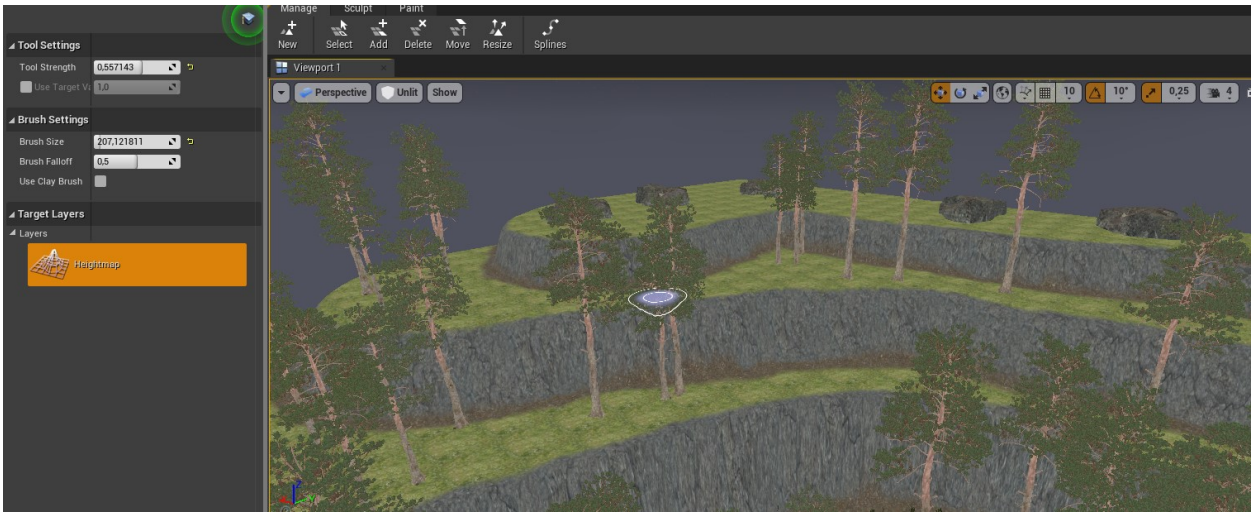
Funkcija za uključivanje svjetiljke se također preko „multicast“-a poziva na poslužitelju, svjetiljka radi sve dok je vrijednost baterije za svjetiljku veća od nule kao što je prikazano na slici 57.



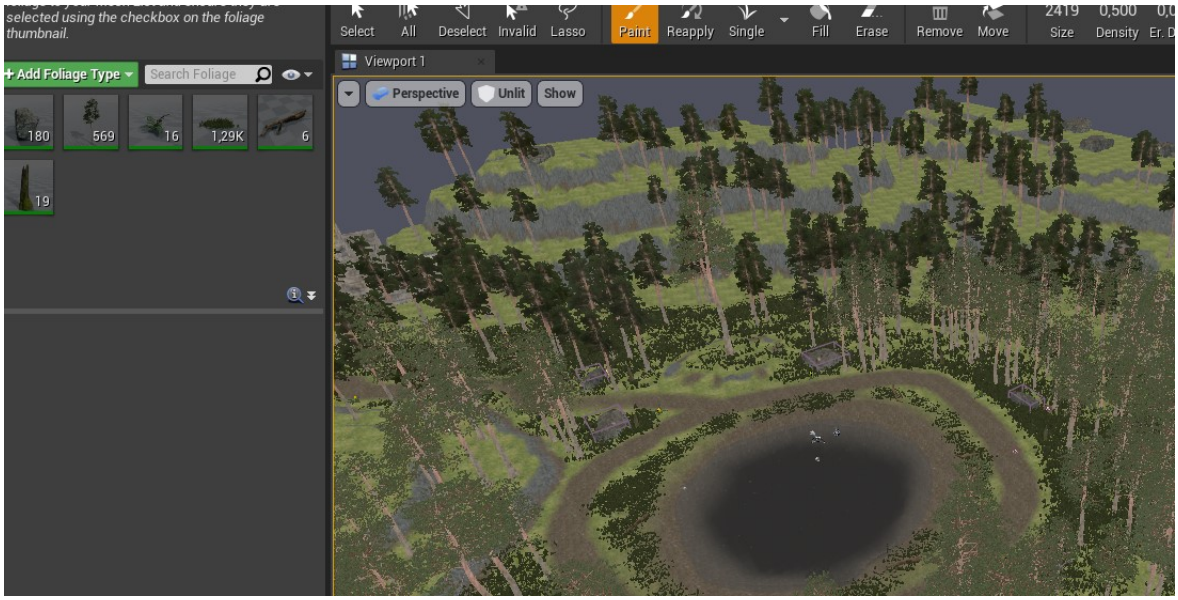
Slika 57. Pritisnut gumb za paljenje svijetla

6.7 Dizajniranje mape za igru

Dizajniranje i stvaranje mape napravljeno je pomoću moda za uređivanje pejzaža i „foliage“ moda gdje se lako dodaje razni okoliš kao što su drveće, trava, kamenje i ostale sitnice kao što je prikazano na slici 58 i 59.

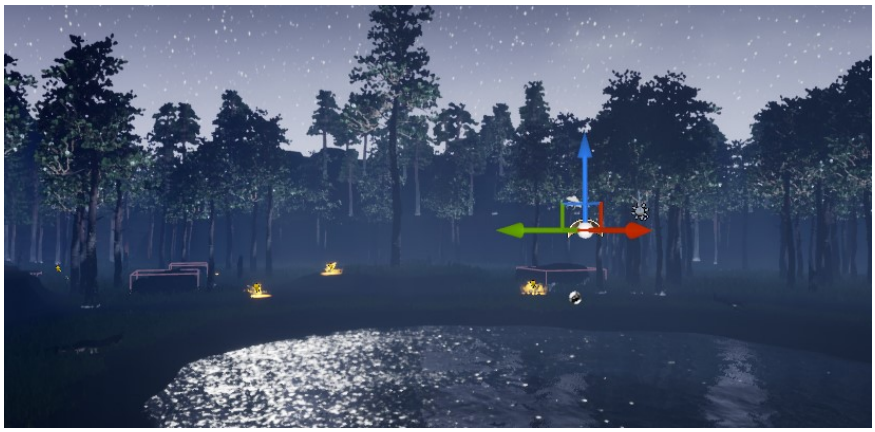


Slika 58. Dizajniranje mape pomoću moda za uređenje okoliša



Slika 59. Dodavanje okoliša pomoću „foliage“ moda

Kako bih scena izgledala mračno potrebno je postaviti nebesko svijetlo na željenu lokaciju kao što je prikazano na slici 60.



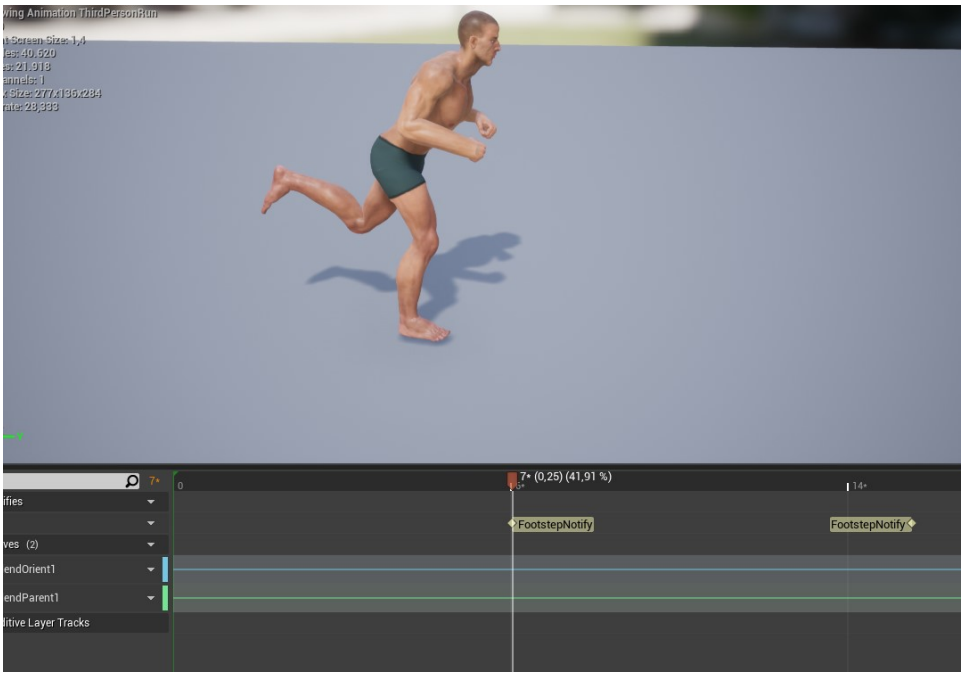
Slika 60. Postavljanje svijetla na željenu lokaciju

Dodavanje magle daje sceni jezivu atmosferu, a na slici 61 i 62 prikazana je usporedba scena s maglom i bez magle.

Slika 61. Scena bez magle

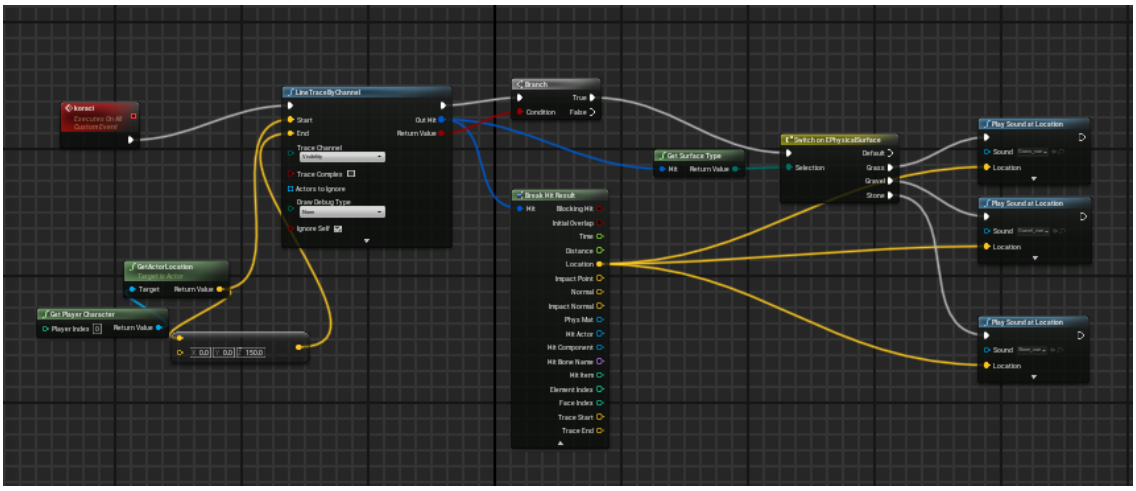
Slika 62. Scena s maglom

Zvuk koraka poziva se tako da svaki puta kada igrač dodirne pod postavlja se obavijest koja zatim poziva zvuk ovisno o površini na kojoj igrač hoda kao što je prikazano na slici 63.



Slika 63. Dodavanje obavijesti da je igrač dodirnuo pod

Funkcija prikazana na slici 64 uzima lokaciju igrača i tip površine na kojem hoda te reproducira zvuk ovisno o kojoj površini se radi.



Slika 64. Funkcija za reprodukciju zvuka koraka

6.8 Primjeri koda

Kod prikazan na slici 65 najprije provjerava dali je zvuk validan i dali je razvojno okruženje postavljeno da koristi zvuk ako i jedan od tih parametara nije zadovoljen funkcija se vraća ranije na radeći ništa. Zatim ako svijet nije dostupan ili ako reprodukcija zvuka nije dostupna u okruženju namjenskog poslužitelja funkcija se također vraća ranije ne radeći ništa. Ako je audio uređaj dostupan u svijetu onda on poziva funkciju koja pušta zvuk na lokaciji i tu se reproducira zvuk sa svim danim parametrima.

```
protected:

	void UGameplayStatics::PlaySoundAtLocation(const UObject* WorldContextObject, class USoundBase* Sound, FVector Location, FRotator Rotation, float VolumeMultiplier, float PitchMultiplier,
{
	if (!Sound || !GEngine || !GEngine->UseSound())
	{
		return;
	}

	UWorld* ThisWorld = GEngine->GetWorldFromContextObject(WorldContextObject, EGetWorldErrorMode::LogAndReturnNull);
	if (!ThisWorld || !ThisWorld->bAllowAudioPlayback || ThisWorld->IsNetMode(NM_DedicatedServer))
	{
		return;
	}

	if (FAudioDeviceHandle AudioDevice = ThisWorld->GetAudioDevice())
	{
		AudioDevice->PlaySoundAtLocation(Sound, ThisWorld, VolumeMultiplier, PitchMultiplier, StartTime, Location, Rotation, AttenuationSettings, ConcurrencySettings, nullptr, OwningActor);
	}
}
```

Slika 65 Reprodukcija zvuka na određenoj lokaciji

Slika 66 prikazuje funkcija koja provjerava dali postoji spremljena igra za specifičan korisnički indeks, koristi se za provjeru dali da se igra učita sa postojećom spremljenom igrom ili da stvori novu.

```
bool UGameplayStatics::DoesSaveGameExist(const FString& SlotName, const int32 UserIndex)
{
	if (ISaveGameSystem* SaveSystem = IPlatformFeaturesModule::Get().GetSaveGameSystem())
	{
		return SaveSystem->DoesSaveGameExist(*SlotName, UserIndex);
	}
	return false;
}
```

Slika 66 Provjera o spremljenoj igri

Prva funkcija prikazana na slici 67 omogućuje klijentu primanju chat poruka i šalje ih putem događaja ChatMessageReceivedEvent i tako omogućuje komunikaciju između igrača. Druga funkcija obrađuje ulazak novih igrača u lobi, formatira obavijest o ulasku s imenom igrača i šalje ih preko istok događaja kao u prvoj funkciji kako bih obavijestila ostale igrače. Treća funkcija radi sve isto kao u druga samo što ona obrađuje napuštanje igrača.

```
ALobbyClient::ALobbyClient(const FObjectInitializer& ObjectInitializer)
    : Super(ObjectInitializer)
{}

void ALobbyClient::ClientReceiveChatMessage_Implementation(const FString& Message)
{
    ChatMessageReceivedEvent.Broadcast(FText::AsCultureInvariant(Message), false);
}

void ALobbyClient::ClientReceiveJoinMessage_Implementation(const FString& NewPlayerName)
{
    FFormatNamedArguments Args;
    Args.Add(TEXT("PlayerName"), FText::AsCultureInvariant(NewPlayerName));

    ChatMessageReceivedEvent.Broadcast(
        FText::Format(NSLOCTEXT("Lobby", "PlayerJoinedFormat", "{PlayerName} joined!"), Args),
        true
    );
}

void ALobbyClient::ClientReceiveLeaveMessage_Implementation(const FString& LeavingPlayerName)
{
    FFormatNamedArguments Args;
    Args.Add(TEXT("PlayerName"), FText::AsCultureInvariant(LeavingPlayerName));

    ChatMessageReceivedEvent.Broadcast(
        FText::Format(NSLOCTEXT("Lobby", "PlayerLeftFormat", "{PlayerName} left."), Args),
        true
    );
}
```

Slika 67 Funkcije za chat i obavijesti u lobiju

Slika 68 prikazuje funkciju za izbacivanje igrača koja prvo provjerava da li je igrač koji pokušava izbaciti druge vlasnik lobija, ako je potvrđeno da je vlasnik onda se odabrani igrač izbacuje iz lobija.

```
void ALobbyHost::ProcessKickRequest(ALobbyPlayer* KickInstigator, ALobbyPlayer* PlayerToKick)
{

    if (KickInstigator != nullptr
        && KickInstigator->IsSessionOwner()
        && PlayerToKick != nullptr)
    {
        DisconnectClient(
            PlayerToKick->GetLobbyClient(),
            NSLOCTEXT("Lobby", "PlayerKicked", "The lobby owner has decided you needed to leave.")
        );
    }
}
```

Slika 68 Funkcija za izbacivanje igrača

Funkcija prikazana na slici 69 provjerava da li igrač postoji i da poruka nije prazna, formatira poruku koja sadrži ime igrača i sadržaj poruke. Nakon toga se poruku šalje svim klijentima u lobiju.

```
void ALobbyHost::ProcessChatMessage(ALobbyPlayer* Sender, const FString& Message)
{
    if (Sender != nullptr && Message.Len() > 0)
    {
        const FString FormattedMessage = FString::Printf(TEXT("%s: %s"), *Sender->GetPlayerName(), *Message);

        for (ALobbyistClient* LobbyClient : GetLobbyClients())
        {
            if (ALobbyClient* CastedLobbyClient = Cast<ALobbyClient>(LobbyClient))
            {
                CastedLobbyClient->ClientReceiveChatMessage(FormattedMessage);
            }
        }
    }
}
```

Slika 69 Provjera i slanje poruke

Kod prikazan na slici 70 pokreće igru iz lobija kada su svi uvjeti ispunjeni, na ovaj način samo vlasnik lobija može pokrenuti igru.

```
void ALobbyHost::ProcessStartGameRequest(ALobbyPlayer* Sender)
{
    if (!bStarting && Sender != nullptr && Sender->IsSessionOwner())
    {
        bStarting = true;
        StartGame();
    }
}
```

Slika 70 Pokretanje igre iz lobija

InitNewPlayer funkcija prikazana na slici 71 inicijalizira nove igrače u igri. Provjerava i skraćuje imena igrača ako su predugačka postavlja opcije igrača i dodjeljuje izgled lika igraču po njegovom odabiru ili nasumično iz dostupnih izgleda.

```
bool ALobbyHost::InitNewPlayer(ALobbyistPlayer* NewPlayer, const FString& InOptions)
{
    if (!Super::InitNewPlayer(NewPlayer, InOptions))
    {
        return false;
    }


    if (NewPlayer->GetPlayerName().Len() > 16)
    {
        ChangePlayerName(NewPlayer, NewPlayer->GetPlayerName().Left(16) + TEXT("\x2026"));
    }


    if (ALobbyPlayer* LobbyPlayer = Cast<ALobbyPlayer>(NewPlayer))
    {
        UExCharacterSkin* Skin = nullptr;

        const FString SkinPath = UGameplayStatics::ParseOption(InOptions, TEXT("CharacterSkin"));
        if (SkinPath.Len() > 0)
        {
            Skin = LoadObject<UExCharacterSkin>(nullptr, *SkinPath);
        }

        if (Skin == nullptr)
        {
            const UExGameInstance* GameInstance = GetGameInstance<UExGameInstance>();
            if (GameInstance != nullptr && GameInstance->AvailableSkins.Num() > 0)
            {
                const int32 RandomSkinIndex = FMath::RandHelper(GameInstance->AvailableSkins.Num());
                Skin = GameInstance->AvailableSkins[RandomSkinIndex].LoadSynchronous();
            }
        }

        if (Skin != nullptr)
        {
            LobbyPlayer->SetCharacterSkin(Skin);
        }
        else
        {
            UE_LOG(LogLobbyist, Warning, TEXT("Failed to load valid skin for %s."), *NewPlayer->GetPlayerName());
        }
    }

    return true;
}
```

Slika 71 Inicijalizacija novih igrača

Slika 72 prikazuje funkciju koja upravlja replikacijom podataka između različitih igrača. Izgled igrača treba biti repliciran između svih igrača i s tim omogućuje svima da vide isto informaciju o izgledu lika. Ping se replicira između poslužitelja i svih igrača ali preskače replikaciju prema vlasniku koji je poslao podatak.

```
void ALobbyPlayer::GetLifetimeReplicatedProps(TArray<FLifetimeProperty>& OutLifetimeProps) const
{
    Super::GetLifetimeReplicatedProps(OutLifetimeProps);

    DOREPLIFETIME(ThisClass, CharacterSkin);
    DOREPLIFETIME_CONDITION(ThisClass, Ping, COND_SkipOwner);
}
```

Slika 72 Replikacija izgleda igrača i pinga

Funkcija prikazana na slici 73 omogućuje stvaranje i ugošćivanje sesije za više igrača. Prvo obavlja postupak resetiranja prethodnih sesija i postavki kako bih se osiguralo čisto okruženje za novu sesiju. Ako stvaranje sesije ne uspije, prikazuje se obavijest o neuspjehu i ponovno se resetira sesija na početno stanje.

```
void AMenuPlayerController::HostSession()
{
    ResetSession();


    const FUniqueNetIdRepl PrimaryPlayerId = GetGameInstance()->GetPrimaryPlayerUniqueIdRepl();
    const IOnlineSessionPtr SessionInterface = Online::GetSessionInterface(GetWorld());
    if (!PrimaryPlayerId.IsValid() || !SessionInterface.IsValid())
    {
        return;
    }

    if (UExWaitingDialog* Dialog = ShowWaitingDialog(NSLOCTEXT("Lobby", "CreatingSession", "Creating session...")))
    {
        Dialog->OnCancel.AddDynamic(this, &ThisClass::ResetSession);
    }

    FOnlineSessionSettings SessionSettings;
    SessionSettings.NumPublicConnections = 8;
    SessionSettings.bShouldAdvertise = true;
    SessionSettings.bAllowJoinInProgress = true;
    SessionSettings.bIsLANMatch = false;
    SessionSettings.bUsesPresence = true;
    SessionSettings.bAllowJoinViaPresence = true;
    SessionSettings.bUseLobbiesIfAvailable = true;

    DelegateHandle_OnCreateSessionComplete = SessionInterface->AddOnCreateSessionCompleteDelegate_Handle(
        FOnCreateSessionCompleteDelegate::CreateUObject(this, &ThisClass::OnCreateSessionComplete));
    if (!SessionInterface->CreateSession(*PrimaryPlayerId, NAME_GameSession, SessionSettings))
    {
        ShowNotificationDialog(NSLOCTEXT("Lobby", "FailedToCreateSession", "Failed to create a new online session."));
        ResetSession();
    }
}
```

Slika 73 Stvaranje sesije

Kod prikazan na slici 74 služi za omogućavanje igraču da se pridruži postojećoj sesiji pomoću pretrage dostupnih sesija. Nakon što igrač pronađe sesiju ima opciju da otkaže ulazak u sesiju. Ako pridruživanje u sesiju ne uspije prikazuje se obavijest o neuspjehu i sesija se resetira.

```
void AMenuPlayerController::HostSession()
{
    ResetSession();


    const FUniqueNetIdRepl PrimaryPlayerId = GetGameInstance()->GetPrimaryPlayerUniqueIdRepl();
    const IOnlineSessionPtr SessionInterface = Online::GetSessionInterface(GetWorld());
    if (!PrimaryPlayerId.IsValid() || !SessionInterface.IsValid())
    {
        return;
    }

    if (UExWaitingDialog* Dialog = ShowWaitingDialog(NSLOCTEXT("Lobby", "CreatingSession", "Creating session...")))
    {
        Dialog->OnCancel.AddDynamic(this, &ThisClass::ResetSession);
    }

    FOnlineSessionSettings SessionSettings;
    SessionSettings.NumPublicConnections = 8;
    SessionSettings.bAllowJoinInProgress = true;
    SessionSettings.bIsLANMatch = false;
    SessionSettings.bUsesPresence = true;
    SessionSettings.bAllowJoinViaPresence = true;
    SessionSettings.bUseLobbiesIfAvailable = true;

    DelegateHandle_OnCreateSessionComplete = SessionInterface->AddOnCreateSessionCompleteDelegate_Handle(
        FOnCreateSessionCompleteDelegate::CreateUObject(this, &ThisClass::OnCreateSessionComplete));
    if (!SessionInterface->CreateSession(*PrimaryPlayerId, NAME_GameSession, SessionSettings))
    {
        ShowNotificationDialog(NSLOCTEXT("Lobby", "FailedToCreateSession", "Failed to create a new online session."));
        ResetSession();
    }
}
```

Slika 74 Pridruživanje sesiji

Slika 75 prikazuje funkciju koja obavještava igrača o različitim mrežnim problemima koji se mogu pojaviti dok se nalazi u lobiju. Igrač tako može reagirati na problem zbog kojeg je izgubio vezu.

```
void AMenuPlayerController::OnLobbyNetworkFailure(const ENetworkFailure::Type FailureType)
{
    FText ErrorMessage;
    switch (FailureType)
    {
        case ENetworkFailure::ConnectionLost:
        case ENetworkFailure::FailureReceived:
        case ENetworkFailure::OutdatedServer:
            ErrorMessage = NSLOCTEXT("Lobby", "ConnectionLostError", "Connection to the remote host has been lost.");
            break;

        case ENetworkFailure::ConnectionTimeout:
            ErrorMessage = NSLOCTEXT("Lobby", "ConnectionTimeoutError", "Connection to the remote host timed out.");
            break;
        case ENetworkFailure::OutdatedClient:
            ErrorMessage = NSLOCTEXT("Lobby", "OutdatedClientError", "Your game is out of date. Please update to the latest version and try again.");
            break;

        case ENetworkFailure::NetDriverAlreadyExists:
        case ENetworkFailure::NetDriverCreateFailure:
        case ENetworkFailure::NetDriverListenFailure:
        case ENetworkFailure::PendingConnectionFailure:
        case ENetworkFailure::NetGuidMismatch:
        case ENetworkFailure::NetChecksumMismatch:
        default:

            ErrorMessage = NSLOCTEXT("Lobby", "GenericNetworkError", "Failed to connect to the remote host.");
            break;
    }
    ShowNotificationDialog(ErrorMessage);
    ResetSession();
}
```

Slika 75 Mrežni problemi

Funkcija prikazana na slici 76. omogućuje igračima da lako prelaze između igre i izbornika putem pritiska određene tipke ili akcije i tako olakšava upravljanje igrom i izbornikom tijekom igranja.

```
void APlayerController::ToggleInGameMenu()
{
    if (InGameMenuWidget == nullptr)
    {
        return;
    }

    if (InGameMenuWidget->IsInViewport())
    {
        InGameMenuWidget->RemoveFromParent();
        SetInputMode(FInputModeGameOnly());
        bShowMouseCursor = false;
    }
    else
    {
        InGameMenuWidget->AddToViewport();
        SetInputMode(FInputModeGameAndUI());
        bShowMouseCursor = true;
    }
}
```

Slika 76 Otvaranje i zatvaranje izbornika

7. OPIS IGRE

Nakon što igrač napravi lobi ili uđe u postojeći, prvo što mora je odabrati lika s kojim želi igrati i kada je spreman javiti poslužitelju da je spreman kako bih on mogao pokrenuti igru. Ako je igrač ugostitelj on još mora odabrati mapu na kojoj želi igrati, vrijeme i ima opciju za izbacivanjem drugih igrača iz lobija. U lobiju se također igrači mogu dopisivati.

Pri ulasku u glavnu mapu igraču se prikazuju instrukcije koje govore igraču da mora pronaći hranu kako bih preživio, prikazuju koje gumbove koristiti za trčanje i paljenja svjetiljke i prikazuje gdje se nalaze podaci poput postotka gladi kao što je prikazano na slici 77.

Slika 77. Instrukcije pri ulasku u igru

Igračev prvi zadatak je pronaći drva kako bih zapalio vatru, drva se nalazi negdje na mapi koju igrač mora istražiti, ali isto tako igrač mora pratiti svoju glad kako ne bih umro, na slici 78 i 79 prikazan je izgled drva i hrane za skupljanje.

Slika 78. Izgled drva

Slika 79. Izgled hrane

Nakon što igrač skupi tri drveta dobiva obavijest da je obavio zadatak, ažurira mu se zadatak i sljedeći zadatak mu postaje zapaliti vatru kao što je prikazano na slici 80.

Slika 80. Skupljena tri drveta

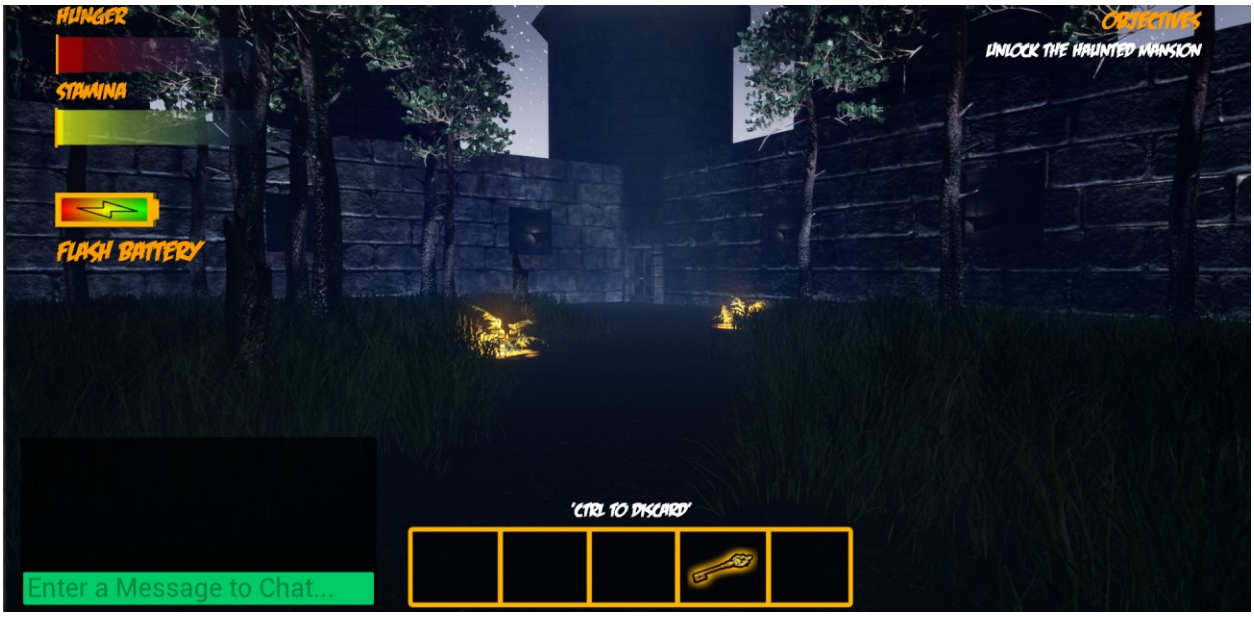
Igrač traži na mapi lokaciju gdje treba zapaliti vatru i nakon što ju pronađe vatra se pali i igraču se oduzimaju drva iz inventara kao što prikazano na slici 81.

Slika 81. Vatra zapaljena

Sljedeći zadatak igrača je pronalazak ključa za ukletu palaču kojeg igrač također mora pronaći negdje na mapi, na slici 82 prikazan je izgled i lokacija ključa.

Slika 82. Prikaz ključa

Nakon što igrač pokupi ključ dobiva novi zadatak da otvori ukletu palaču kao što je prikazano na slici 83.

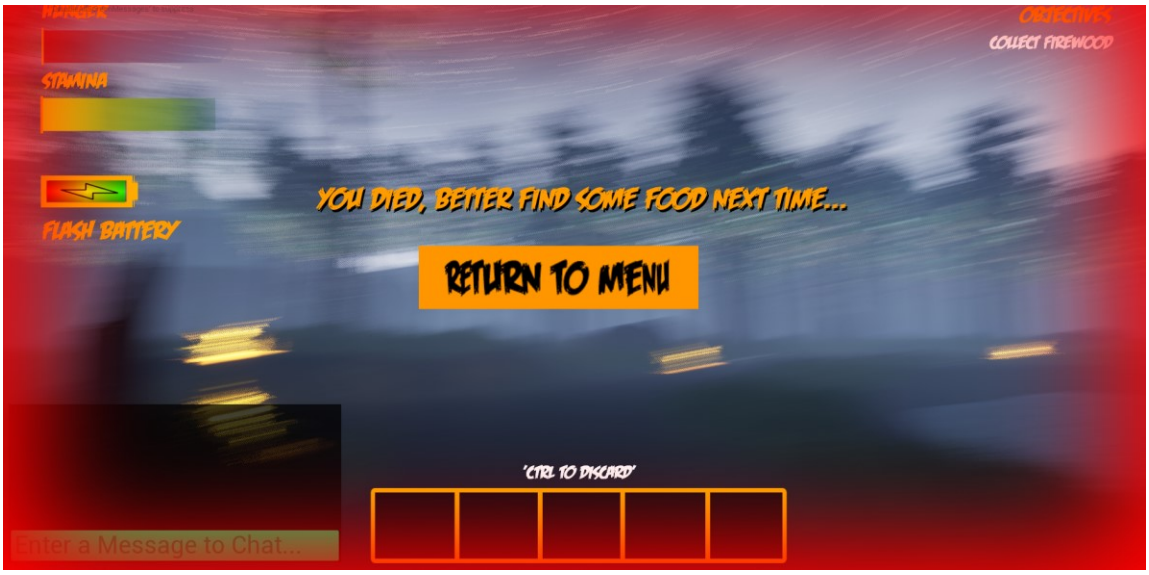


Slika 83. Otključaj ukleti palaču

Kada igrač otključa ukletu palaču otvaraju se vrata i nakon što igrač prođe kroz vrata pokreče se prepad gdje iskače zombi sa strašnim zvukom kao što je prikazano na slici 84.

Slika 84. Otvorena vrata i prepad

Kada igrač ostane bez hrane igrač umire i prikazuje se sučelje koje mu javlja da je umro sa gumbom za povratak na glavni izbornik kao što je prikazano na slici 85.



Slika 85 Smrt igrača

8. ZAKLJUČAK

Zadatak ovog rada bio je opisati izradu horor igre s preživljavanjem za više igrača. U početnom dijelu rada opisane su neke igra koje su poslužile kao inspiracija, prikazano je na što sve treba paziti kod izrade za više igrača i po tome odrediti kako želimo da igra funkcionira. Opisane su osnovne značajke replikacije kao što su replikacija varijabli i replikacija komponenti. Izrada same igre detaljno je opisano pomoću isječaka koda i objašnjenja funkcionalnosti prikazanih isječaka.

Neke od prednosti igre napravljene u ovom radu su te da je lako povezivanje više igrača preko klijent poslužitelja i razne mogućnosti koje igrač ima u lobiju kao što je odabir mape i izbacivanje igrača. Ograničenja igre su ta da u ovoj verziji igre nema audio komunikacije između igrača već samo tekstualne komunikacije. Audio komunikacija bila bih jedna od mogućih nadogradnji igre isto kao i dodavanje novih zadataka u igru.

Izrada video igre za više igrača je skroz drugi svijet u usporedbi sa izradom igre za jednog igrača. Izrada je kompleksnija i zahtjeva dobro bazu pomoću koje onda možemo lakše programirati funkcionalnosti za igru. Također testiranje pomoću drugih korisnika je važan dio kod izrade igre za više igrača jer samostalno testiranje nam neće prikazati mnoge greške u igri koje će biti izložene tek kada se drugi korisnici spoje na našu igru. Ključni dio kod izrade bilo je razumijevanje replikacije odnosno u kojim akterima se ona treba uključiti a u kojim ne i korištenje RPC-ova tako da se zna na kojoj će se mašini svaki od njih izvoditi.

Podžanr odabran za ovaj rad je već spomenuta horor igra s preživljavanjem za više igrača. Igra se igra u pravom prvom licu gdje se mogu vidjeti ruke i tijelo lika, takvo lice često se koristi u ovakvim igrama. Atmosfera i zvuk u takvim igrama su jako važni aspekti jer preko njih igrač se uživljava u igru i tako kada dođe do plašenja veće su šanse da će se igrač uplašiti. U igri ovog žanra igrač najčešće ima izazov pronaći predmete koji otključavaju put do novih područja i rješavati zagonetke za nastavak igre kao što je izazov i u igri opisanoj u radu.

LITERATURA

Carnall, B. (2016). *Unreal Engine 4.X By Example.* Packt Publishing. Preuzeto 23. 8 2022

Cordone, R. (2019). *Unreal Engine 4 Game Development Quick Start Guide.* Birmingham: Packt Publishing. Preuzeto 30. 7 2022

DE, S. (7. Prosinac 2021). *5 Different Types of Horror Video Games, Explained*. Preuzeto 11. 8 2022 iz MUO - Technology, Simplified: https://www.makeuseof.com/horror-video-game-subgenres-explained/

Grove, R. (23. 7 2019). *Focus: Unreal Engine - A Brief History of Unreal*. Preuzeto 1. 9 2022 iz Renderosity Magazine: https://magazine.renderosity.com/article/5330/focus-unreal-engine-a-brief-history-of-unreal

Networking Overview. (n.d. n.d n.d). *Networking Overview.* Preuzeto 3. 7 2022 iz Unreal Engine 4 Documentation: https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/Networking/Overview/

Tarantola, A. (13. 5 2020). *Epic Games teases its new, nearly-photorealistic Unreal Engine 5*. Preuzeto 17. 9 2022 iz Engadget | Technology News & Reviews: https://www.engadget.com/epic-games-unreal-engine-5-demo-150044561.html?

PRILOZI

Popis Slika

Slika 1 Snimka ekrana iz igre The Forest 2
Slika 2 Snimka ekrana iz igre Outlast 3
Slika 3. Prikaz igranja na lokalnoj mreži 4
Slika 4. Poslužitelj obrađuje igru, a klijenti prokazuju igru korisnicima 5
Slika 5. Primjer namjenskog poslužitelja s dva poslužitelja 7
Slika 6. Provjera dali imaju spremljeni podaci o igri 9
Slika 7. Unos korisničkog imena i ikone 9
Slika 8 Otvaranje lobija sa odabranim postavkama 10
Slika 9 Otvoren lobi sa odabranim postavkama 10
Slika 10. Ulaz u poslužitelj - pozvan i glavnog izbornika 11
Slika 11. poslužitelj je pronađen - prikazuje odbrojavanje za ulazak 11
Slika 12. Prikaz obavijesti da je poslužitelj pronađen 11
Slika 13. Stvaranje i ažuriranje igrača 12
Slika 14. Poslužitelj govori svim klijentima da ažuriraju informacije o lobiju 12
Slika 15. Prikaz promijene i ažuriranja postavki 12
Slika 16. Zamjena trenutnog lik sa odabranim likom sa ekrana za odabir likova 13
Slika 17. Prikaz zamijene likova 13
Slika 18. Dodavanje igrača na listu za izbacivanje 14
Slika 19. Izbacuje odabranog igrača iz lobija 14
Slika 20. Izbacivanje igrača preko posjedovnog klijenta 15
Slika 21. Izbacivanje lobi izbornika i stvaranje prikaza učitavanja 15
Slika 22. Učitavanje 15
Slika 23. Procesiranje poruka u chatu 16
Slika 24. Ažuriranje prikaza chata 16
Slika 25 Komunikacije između dva igrača 16
Slika 26. Funkcija stvaranja igrača 17
Slika 27. Prikaz stvorenog igrača 17
Slika 28. Glavni izbornik - izgled 18
Slika 29. Referenca na instancu informacija o igri 18
Slika 30. Uzima i prikazuje izbornik za ugošćivanje igre, na ovaj način funkcioniraju i ostale opcije u glavnom izborniku 19
Slika 31. Izgled izbornika za ugošćivanje igre 19
Slika 32. Povećavanje i smanjivanje broja igrača 20
Slika 33. Prihvaća i primjenjuje korisničke promjene na S Player Info 20
Slika 34. Spremanje igre 21
Slika 35. Postavke igrač 21
Slika 36. Sučelje za pronalazak poslužitelja 22
Slika 37. Gumb za prihvaćanje/pretraživanje i pridruživanje poslužitelju ako je dostupan 22
Slika 38. Provjera dostupnih poslužitelja 22
Slika 39. Funkcija za pokretanje igre odnosno klijent javlja da je spreman 23
Slika 40. Izgled lobi izbornika za poslužitelja i za klijenta 23
Slika 41. Izgled izbornika za postavke igre 24
Slika 42. Izbacivanje igrača na pritisak gumba 24
Slika 43. Izbornik za izbacivanje igrača 24
Slika 44. Odabir likova 25
Slika 45. Uzima instancu igre/ dodjeljuje gumbove za likove 25
Slika 46. Pokušava dodijeliti odabranog lika iz izbornika za odabir likova 26

Slika 47. Sučelje igrača....26
Slika 48. Funkcija za prikaz izdržljivosti....27
Slika 49. Prikaz ikona u inventaru....27
Slika 50. Funkcija za izbacivanje predmeta....28
Slika 51. Pozivanje funkcije nakon prijelaza aktera....28
Slika 52. Funkcija za dodavanje drva u inventar....29
Slika 53. Kada akter aktivira ovaj događaj pokreče se sekvenca koja stvara vatru....29
Slika 54. Prikaz sekvence....30
Slika 55. Pozivanje sekvence i zvuka na lokaciji....30
Slika 56. Funkcija za trčanje....31
Slika 57. Pritisnut gumb za paljenje svijetla....31
Slika 58. Dizajniranje mape pomoću moda za uređenje okoliša....32
Slika 59. Dodavanje okoliša pomoću „foliage“ moda....32
Slika 60. Postavljanje svijetla na željenu lokaciju....33
Slika 61. Scena bez magle....33
Slika 62. Scena s maglom....33
Slika 63. Dodavanje obavijesti da je igrač dodirnuo pod....34
Slika 64. Funkcija za reprodukciju zvuka koraka....34
Slika 65 Reprodukcija zvuka na određenoj lokaciji....35
Slika 66 Provjera o spremljenoj igri....35
Slika 67 Funkcije za chat i obavijesti u lobiju....36
Slika 68 Funkcija za izbacivanje igrača....37
Slika 69 Provjera i slanje poruke....37
Slika 70 Pokretanje igre iz lobija....38
Slika 71 Inicijalizacija novih igrača....38
Slika 72 Replikacija izgleda igrača i pinga....39
Slika 73 Stvaranje sesije....39
Slika 74 Pridruživanje sesiji....40
Slika 75 Mrežni problemi....41
Slika 76 Otvaranje i zatvaranje izbornika....41
Slika 77. Instrukcije pri ulasku u igru....42
Slika 78. Izgled drva....43
Slika 79. Izgled hrane....43
Slika 80. Skupljena tri drveta....44
Slika 81. Vatra zapaljena....44
Slika 82. Prikaz ključa....45
Slika 83. Otključaj ukleti palaču....45
Slika 84. Otvorena vrata i prepad....46
Slika 85 Smrt igrača....46