

# Stvarno vrijeme u Unreal Engineu

---

**Blažeković, Lea**

**Undergraduate thesis / Završni rad**

**2024**

*Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj:* **University North / Sveučilište Sjever**

*Permanent link / Trajna poveznica:* <https://um.nsk.hr/um:nbn:hr:122:300311>

*Rights / Prava:* [In copyright](#)/[Zaštićeno autorskim pravom.](#)

*Download date / Datum preuzimanja:* **2024-12-25**



*Repository / Repozitorij:*

[University North Digital Repository](#)





# Sveučilište Sjever

**Završni rad br. 891/MM/2024**

## **Stvarno vrijeme u Unreal Engineu**

**Lea Blažeković, 0016139040**

Varaždin, rujan 2024. godine



### IZJAVA O AUTORSTVU

Završni/diplomski/specijalistički rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Lea Blažeković (ime i prezime) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog/specijalističkog (opisati nepotrebno) rada pod naslovom Stručna vježba u Integral Enginju (upisati naslov) te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:

(upisati ime i prezime)

Lea Blažeković  
Blažeković  
(vlastoručni potpis)

Sukladno članku 58., 59. i 61. Zakona o visokom obrazovanju i znanstvenoj djelatnosti završne/diplomske/specijalističke radove sveučilišta su dužna objaviti u roku od 30 dana od dana obrane na nacionalnom repozitoriju odnosno repozitoriju visokog učilišta.

Sukladno članku 111. Zakona o autorskom pravu i srodnim pravima student se ne može protiviti da se njegov završni rad stvoren na bilo kojem studiju na visokom učilištu učini dostupnim javnosti na odgovarajućoj javnoj mrežnoj bazi sveučilišne knjižnice, knjižnice sastavnice sveučilišta, knjižnice veleučilišta ili visoke škole i/li na javnoj mrežnoj bazi završnih radova Nacionalne i sveučilišne knjižnice, sukladno zakonu kojim se uređuje umjetnička djelatnost i visoko obrazovanje.

# Prijava završnog rada

## Definiranje teme završnog rada i povjerenstva

ODJEL

STUDIJI

PRISTUPNIK  JMBAG

DATUM  KOLEGIJI

NASLOV RADA

NASLOV RADA NA ENGL. JEZIKU

MENTOR  ZVANJE

- ČLANOVI POVJERENSTVA
- - 
  - 
  - 
  -

## Zadatak završnog rada

BROJ

OPIS

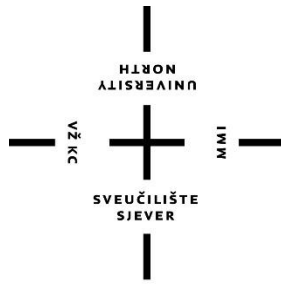
U ovom završnom radu bit će korišten Unreal Engine za kreiranje simulacije sobe. Izmodelirati će se 3D model stvarne sobe i kroz prozor će se moći vidjeti vremenski uvjeti koji će također biti kreirani u istom programu. Vrijeme (kiša, snijeg, sunce, oblačno vrijeme i slično) će biti sinkronizirano s pravom prognozom putem API ključa i stranice Open Weather Map koja na temelju spomenutog ključa i određene lokacije (geografska visina i širina) vraća trenutnu vremensku prognozu traženog mjesta. Povezivanje podataka s Open Weather Map stranice i simulacije bit će omogućeno pomoću Blueprint Visual Scripting sistema unutar Unreal Enginea.

ZADATAK URUČEN

30.8.2024.



*Bernik*



# Sveučilište Sjever

Odjel za multimediju

Završni rad br. 891/MM/2024

## Stvarno vrijeme u Unreal Engineu

### Student

Lea Blažeković, 0016139040

### Mentor

Andrija Bernik, Doc.dr.sc.

Varaždin, rujan 2024. godine

## **Predgovor**

Korištenje podataka iz stvarnog svijeta putem raznih API-a te njihova integracija unutar Unreal Engine projekta i transformacija tih podataka u multimedijalni prikaz je jedno od područja koje može imati široku primjenu. Takva prezentacija podataka nije samo zanimljiva unutar video igara nego i za prezentaciju različitih podataka u vizualno zanimljivom obliku. Zbog toga je u ovom radu odabrana tema prikaz vremenskih prilika u stvarnom vremenu.

## Sažetak

Ovaj rad će pokazati mogućnost Unreal Engine-a za simuliranje vremenskih prilika u stvarnom vremenu koristeći podatke dohvaćene sa Open Weather servera putem njihovog API-a. Objasniti će se kakav je Unreal Engine program i koje su njegove primjene. Opisati će se rad VaRest dodatka koji isključivo odrađuje komunikaciju između klijenta i servisa. Pojasniti će se što online servis Open Weather Map pruža i kako on pomaže pri izradi ove simulacije. Proces izrade simulacije sastajat će se od kreiranja projekta u Unreal Engine-u, kako se modelirala okolina u kojoj će se izvoditi simulacija te kako kreirati tablicu i podatke u njoj potrebne za prikazivanje vremenskih prilika. Podaci unutar tablice prikazivati će različite tipove varijabli kao što su brojevi, zvukovi ili sustavi čestica. Za svaki tip varijable dodatno će se pojasniti kako on utječe na prikaz vremenskih pojava. Pokazati će se postupak izrade sustava čestica i novih parametara koji će utjecati na prostor u simulaciji. Nakon toga izrađuje se logika simulacije unutar Blueprint sustava za programiranje. Blueprint je vizualni skriptni alat koji omogućava korisnicima da implementiraju kompleksnu logiku i interakciju unutar Unreal Engine-a. Uz Blueprint i integraciju VaRest dodatka uspostaviti će se komunikacija s Open Weather Map servisom kako bi se dohvatili podaci sa Open Weather servera. Povratna informacija će dohvatiti odgovarajuće podatke iz tablice da se prikaže trenutno vrijeme. Osim prikaza trenutnog vremena projekt će također moći prikazati neke opcije koje mogu biti simulirane. Izraditi će se sučelje putem kojeg će se moći birati prikaz vremenskih prilika koje aplikacija može simulirati i vremenskih prilika dohvaćenih sa Open Weather Maps servisa. Prilikom pokretanja aplikacije, aplikacija automatski dohvati podatke putem API-a te iz tablice u kojoj su definirani svi parametri za prikaz simulacije putem ID-a dohvati određenu skupinu parametara i prikazuje ih na sceni. Korisnik putem sučelja može odabrati kako se prikazuju određene vremenske prilike ili odabrati da aplikacija učita i prikaže podatke sa servisa.

**Ključne riječi:** Unreal Engine, simulacija, API, VaRest dodatak, tablica podataka, vremenska prognoza, Blueprint sustav

## Summary

This project will demonstrate the capabilities of Unreal Engine for simulating real-time weather conditions using data retrieved from the Open Weather server via their API. It will explain what Unreal Engine is and its use. The functionality of the VaRest plugin, which exclusively handles communication between the client and the service, will be described. The online service Open Weather Map and how it assists in the creation of this simulation will also be explained. The process of creating the simulation will consist of project creation in Unreal Engine, how the environment in which the simulation will take place is modeled, and how to create a data table and the necessary data within it to display weather conditions. The data in the table will display different types of variables, such as numbers, sounds, or particle systems. Each type of variable will be further explained, focusing on how it affects the display of weather phenomena. The process of creating a particle system and new parameters that will influence the space in the simulation will also be shown. Afterward, the simulation logic will be developed within the Blueprint visual scripting system. Blueprint is a visual scripting tool that allows users to implement complex logic and interactions within Unreal Engine. Along with Blueprint and the integration of the VaRest plugin, communication with the Open Weather Map service will be established to retrieve data from the Open Weather server. The returned information will fetch the appropriate data from the table to display the current weather conditions. In addition to displaying the current weather, the project will also be able to present some options that can be simulated. An interface will be created through which different weather conditions, both simulated by the application and retrieved from the Open Weather Map service, can be selected. When the application is launched, it will automatically retrieve data via the API, and from the table where all parameters for the simulation display are defined, it will fetch a specific set of parameters by ID and display them in the scene. Through the interface, the user will be able to choose how certain weather conditions are displayed or select to load and display data from the service.

**Keywords:** Unreal Engine, simulation, API, VaRest plugin, data table, weather forecast, Blueprint system



## Popis korištenih kratica

<b>API</b>	Application Programming Interface Skup pravila i protokola za komunikaciju između aplikacija/programa.
<b>ID</b>	Identification Jedinstveni identifikacijski broj često korišten unutar baza podataka.
<b>UE</b>	Unreal Engine
<b>3D</b>	Trodimenzionalno
<b>REST</b>	Representational State Transfer Omogućuje jednostavnu razmjenu podataka između klijenta i poslužitelja.
<b>HTTP</b>	Hypertext Transfer Protocol Osnovni protokol za komunikaciju između preglednika i web poslužitelja.
<b>URL</b>	Uniform Resource Locator Adresa jedinstvena određenom resursu koji se nalazi na internetu.
<b>JSON</b>	JavaScript Object Notation Dio JavaScripta koji razmjenjuje podatke između servisa.

# Sadržaj

Predgovor.....	5
Sažetak.....	6
Summary.....	7
Popis korištenih kratica.....	8
Sadržaj.....	9
1. Uvod.....	10
1.1. Cilj projekta.....	10
1.2. Unreal Engine.....	10
1.3. VaRest dodatak.....	11
1.4. Open Weather servis.....	12
2. Početak rada u Unreal Engine-u.....	14
2.1. Kreiranje mape.....	14
2.1.1. Modeliranje 3D objekata.....	14
2.1.2. Landscape mode.....	17
2.1.3. Dodavanje modela i uređivanje okoline.....	17
2.1.4. Instanciranje.....	20
3. Migracija projekta iz verzije Unreal Engine 4 u verziju Unreal Engine 5.....	23
4. Kreiranje Open Weather profila i generiranje API ključa.....	24
5. Tablica podataka.....	27
5.1. Struktura tablice.....	27
6. Sastavni dijelovi vremenskih prilika.....	29
6.1. Sunce, nebo i oblaci.....	29
6.1.1. Sunce.....	29
6.1.2. Nebo.....	29
6.1.3. Gustoća oblaka.....	32
6.1.4. Ugođaj unutar interijera.....	34
6.2. Niagara sustav čestica.....	34
6.2.1. Snijeg.....	35
6.2.2. Kiša.....	39
6.2.3. Grmljavina.....	42
6.2.4. Magla.....	44
6.2.5. Parametri materijala.....	46
6.3. Glazba i ambijentalni zvukovi.....	51
6.3.1. Istovremenost zvuka.....	51
6.3.2. Prikaz vremenskih prilika i njihovih postavki.....	53
7. Programiranje simulacije.....	55
8. Otvaranje i zatvaranje prozora.....	65
9. Interaktivno sučelje.....	69
10. Zaključak.....	73
11. Literatura.....	74
Popis slika.....	75

# 1. Uvod

## 1.1. Cilj projekta

Cilj ovog projekta je prikazati korištenje podataka iz stvarnog svijeta putem API-a kako bi se izradila simulacija pravog vremena za određenu lokaciju. Simulacija je postignuta tako da izrađena logika unutar Blueprint sustava Unreal Engine-a kontrolira svojstva i prisustvo elemenata na sceni. Projekt teži stvaranju simuliranog okruženja koje se prilagođava vremenskim uvjetima.

## 1.2. Unreal Engine

Unreal Engine je svestran alat koji je primarno korišten za izradu video igara. Trenutno aktualna verzija programa je Unreal Engine 5.4. UE1 je razvio Tim Sweeney, osnivač Epic Games tvrtke. Inicijalno, prva verzija ovog programa bila je korištena za izradu igre „Unreal“. Epic je kasnije odlučio licencirati program ostalim tvrtkama. Verzijom 4 program je postao besplatno dostupan svima. Iako je Unreal Engine 4 sam po sebi objektivno odličan program za izradu igara, nedavno je izašao Unreal Engine 5. Ova verzija je dobila potpuno novi sustav osvjetljenja koji realističnije osvjetljuje prostor u 3D grafici. Nadograđen je i sustav za geometriju koji omogućava prikaz vrlo detaljnih objekata pod nazivom Nanite. Također je moguće koristiti dodatne resurse računala kako bi generiranje aplikacije prošlo brže.



*Slika 1.1 Logotip Unreal Engine-a*

Kako je efikasnost i funkcionalnost programa porasla, porasla je i njegoa primjena. Tvrtke koriste Unreal Engine za kreiranje 3D animacija, kreiranje realističnih okolina za projekciju na filmskim scenama, vizualizaciju prostora i/ili proizvoda. Danas ovaj program nije ograničen samo na gaming industriju, već se proširio na marketing, televiziju, film i ostale sektore gdje potreban detaljan i realističan vizualan prikaz.

Jedna od ključnih značajki Unreal Engine-a koja ga čini toliko pristupačnim i moćnim alatom kojeg može koristiti bilo tko je njegov Blueprint sustav. Pomoću tog grafičkog sučelja moguće je

izgraditi potpunu logiku igre, simulacije ili aplikacije tako da je u potpunosti moguće izbjeći korištenje C++ jezika. Ovakav vizualan sustav programiranja otvara vrata široj demografiji korisnika što za sobom povlači dodatni potencijal za nove primjene ovog programa.

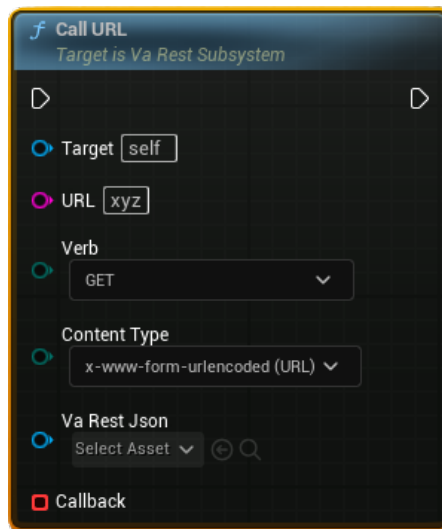
### 1.3. VaRest dodatak



*Slika 1.2 VaRest dodatak*

Unreal Engine uz osnovnu funkcionalnost moguće je proširiti dodacima (*eng. plugins*). VaRest, originalno napravljen za Unreal Engine 4 od strane Vladimira Alyamkina, pruža različite funkcije za REST komunikaciju između aplikacije i servera. REST je jednostavan i fleksibilan način komunikacije servera na internetu. Funkcije unutar VaRest dodatka mogu se koristiti u Blueprint sustavu. Prednost toga jest lakša primjena funkcija programa i preglednija vizualna povezanosti svih elemenata. VaRest se ažurira kako bi se mogao koristiti u novijim verzijama UE.

Ono što VaRest omogućava za ovaj projekt jest komunikacija između klijenta i poslužitelja. Ova simulacija koristi VaRest kako bi poslala HTTP zahtjev na API krajnju točku sa određenim parametrima kako bi dobila odgovor u JSON formatu sa Open Weather Map servera.



Slika 1.3 VaRest čvor za obradu HTTP zahtjeva

HTTP zahtjev sastoji se od URL adrese, metode koja se koristi za slanje („Verb“ odjeljak na čvoru sa slike 3), API krajnje točke severa. Koristi se GET metoda koja omogućava da se u zahtjevu pošalju potrebni parametri unutar URL-a. Odgovor poslužitelja su podaci o vremenskoj prognozi. Ti podaci šalju se na Callback funkciju koja obrađuje primljene JSON podatke. Sve postavke unutar VaRest čvora za HTTP zahtjev se određuju prilikom izrade aplikacije.

#### 1.4. Open Weather servis

Ova simulacija koristi uslugu Open Weather Map servisa. On pruža različite pakete podataka. Korisnicima pruža pristup velikom spektru informacija putem API-a čije podatke je moguće integrirati u vlastite aplikacije, web stranice i programe. U ovoj simulaciji će se koristiti besplatan paket.

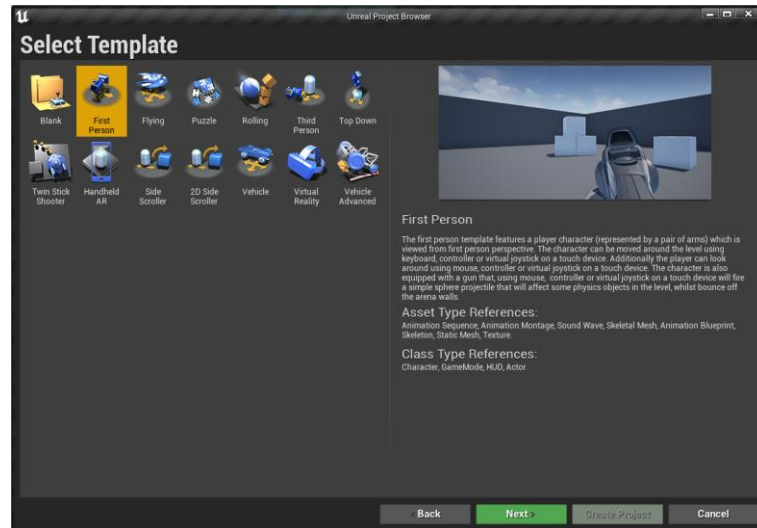


*Slika 1.4 Logotip Open Weather Map servisa*

Open Weather daje podatke o vremenskim prilikama, a za potrebe ovog projekta dovoljna je pristup trenutnim podacima o prognozi koji se nalazi u besplatnom paketu. Kako bi simulacija mogla pristupiti podacima potrebno je na servisu kreirati profil u kojem se mora generirati API ključ koji je potreban za svu komunikaciju sa servisom. Potrebno je napomenuti da postoji ograničenje na broj poziva API-u u određenom vremenskom periodu (60 poziva u minuti), no nije potrebno vršiti zahtjev često jer se podaci na servisu ne mijenjaju u tom ograničenom vremenskom periodu. Većinom slučajeva će poziv unutar kraćeg vremenskog perioda rezultirati u istim dohvaćenim podacima.

## 2. Početak rada u Unreal Engine-u

Kod pokretanja Unreal Engine-a dobije se ekran s predlošcima za projekt, a za potrebe ovog projekta odabran je First Person predložak jer sadrži temelj za izradu ove simulacije.



Slika 2.1 Kreiranje projekta

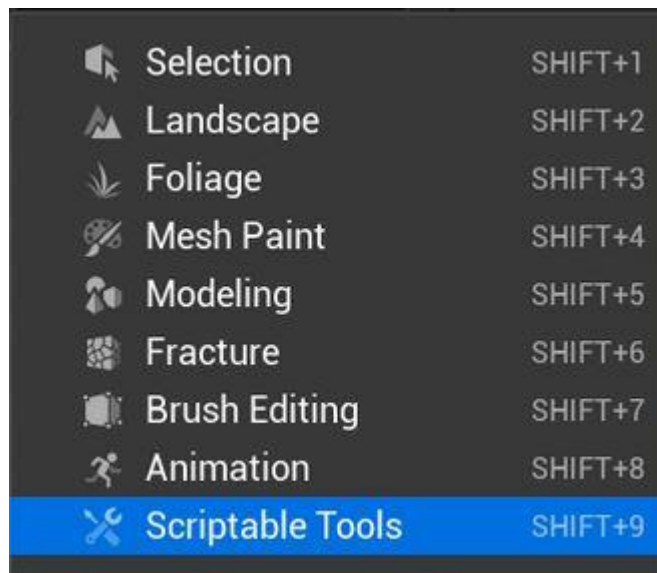
Nakon toga potrebno je nazvati projekt, odabrati metodu kreiranja logike projekta i lokaciju gdje će se projekt spremati. Potrebno je paziti da dužina naziva projekta ne prelazi 20 znakova i ne smije sadržavati specijalne znakove kao zarez, doljnja crta i slično.

### 2.1. Kreiranje mape

Najprije je potrebno izraditi prostor unutar kojeg će se simulacija odvijati. Ova simulacija će imati interijer i eksterijer. Prvo će se izraditi interijer, a potom eksterijer.

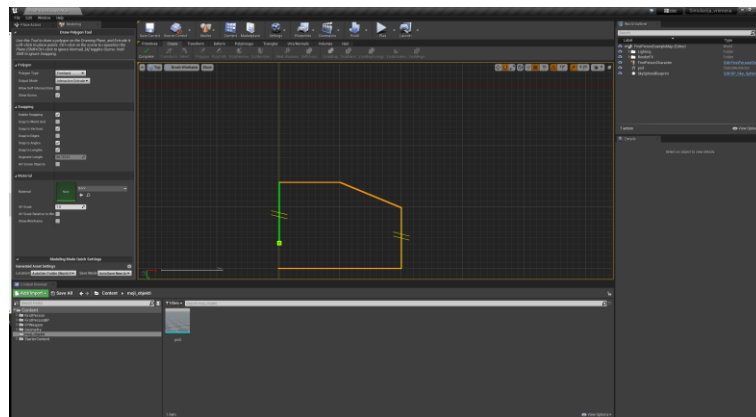
#### 2.1.1. Modeliranje 3D objekata

Unreal Engine ima mogućnost modeliranja 3D objekata. U lijevom dijelu sučelja programa nalazi se padajući izbornik u kojima se odabire način rada.



Slika 2.2 Načini rada u Unreal Engine-u

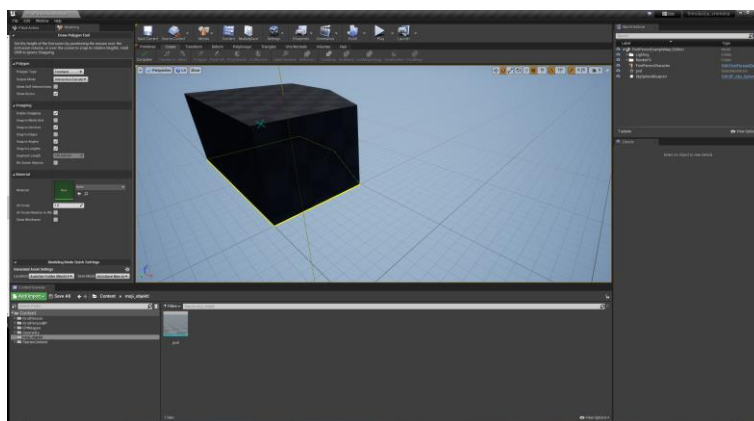
Modeliranje 3D objekata vrši se u djelu gdje je modeliranje (*eng. Modeling mode*)  
 Način modeliranja sastoji se od brojnih alata. Oblik prostorije izrađen je Polygon alatom.



Slika 2.3 Polygon alat

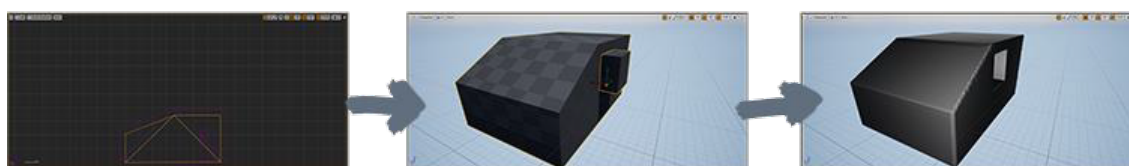
Ovaj alat omogućava izradu modela koji nije standardnog oblika kao kocka, valjak ili kugla. Na slici 7 vidljivo je da je uključen ortografski prikaz koji omogućuje rad s 2D projekcijama objekata bez perspektivnog izobličenja. Na taj način će lakše biti izraditi bazu za oblik sobe. Klikom miša postavljaju se početne točke koje će biti kutevi novog modela. Kada se inicijalni oblik nacrtat, može se vratiti na 3D način pogleda (*eng. view*) i nacrtani oblik se ekstrudira (povlačenjem miša dodaje se treća dimenzija objektu).





*Slika 2.4 Novi oblik nastao podloču Polygon alata*

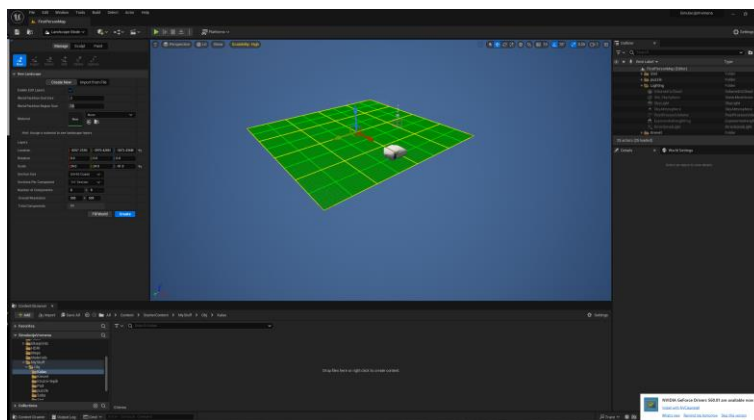
Dobiveni oblik je dalje moguće dodatno uređivati ostalim alatima unutar načina rada za modeliranje. Novi oblik je solidan objekt. Kako bi soba dobila prostor i poziciju gdje dolazi prozor za to je potreban alat VoxBoolean koji korištenjem drugog 3D oblika oduzima dijelove originalnog oblika. Prethodno izrađeni model kopira se (tipke ctrl i D) kako bi se pomoću njega kreirao prostor unutar sobe. Duplikat se proporcionalno samnjuje alatom za skaliranje (*eng. scale*). Pomoću gizma (*eng. gizmo*) biraju se osi po kojima se želi mijenjati veličina objekta, ukoliko se objekt želi skalirati proporcionalno klikne se na dio gdje se na gizmu sve osi sijeku, u sredinu gizmo alata. Za precizno pozicioniranje skaliranog objekta za primjenu VoxBoolean alata ponovno se uključuje ortografski prikaz. Tako je moguće centrirati oba oblika precizno. Kada su objekti u odgovarajućoj poziciji primjenom VoxBoolean alata nastaje novi oblik koji u sebi ima prazan prostor. Isti postupak se ponavlja za otvor prozora. Stvori se kocka proporcija odgovarajućim veličini prozora pomoću koje se boolean alatom oduzima od modela sobe.



*Slika 2.5 Izrađen model sobe*

### 2.1.2. Landscape mode

Unreal Engine u ovom načinu rada nudi praktičan i intuitivan način izrade terena.



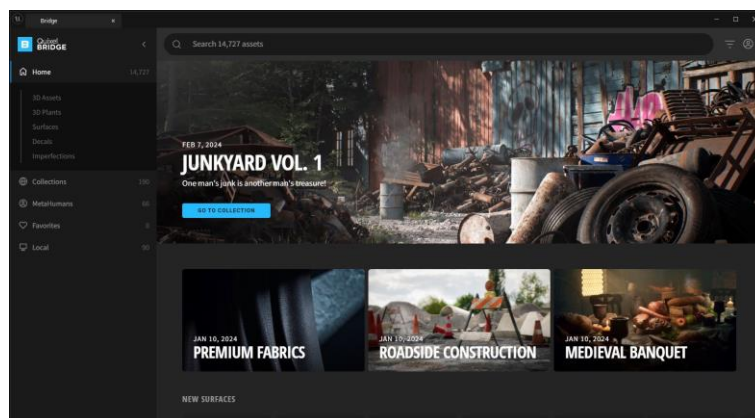
*Slika 2.6 Kreiranje terena Landscape mode-om*

Započinje se postavljanjem proporcija i položaja terena. U ovom djelu veličina terena se određuje skalarnim veličinama „World Partition Grid“ i „World Partition Region“ (particija/segmenti mreže i regija). U ovom projektu je potreban dio ulice s dinamičnim reljefom koji je uvijek učitao. Mreža dijeli teren na manje, upravljive dijelove koji kod velikih mapa pomaže u optimizaciji performansa igre gdje je moguće selektivno odabrati koji dio mape treba biti učitao. Veličina regije određuje specifične dijelove terena koje treba obraditi detaljno i učitati u memoriju. Na temelju tih činjenica i informacije da je potrebno imati cijelu „mapu“ učitano s jednim tipom reljefa, ove veličine će se postaviti na najmanju moguću skalu; veličina mreže je 2, i regija je 1.

Postoje različite metode modeliranja terena unutar načina rada Sculpt. Za potrebe ovog terena koristili su se alati Sculpt i Smooth. Sculpt podiže izbočine na reljefu, a Smooth postepeno smanjuje razliku u visini reljefa.

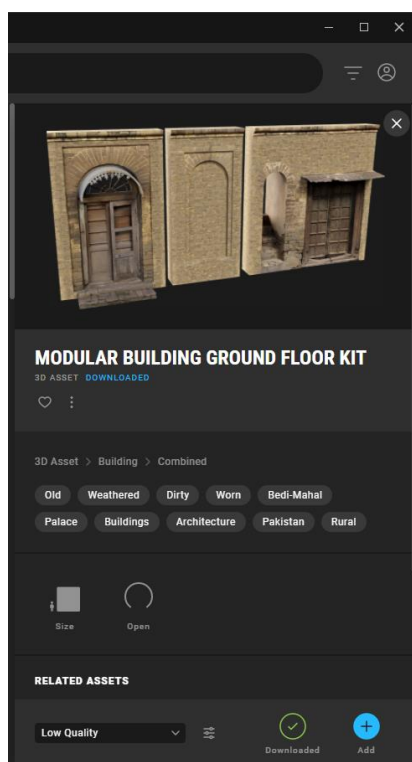
### 2.1.3. Dodavanje modela i uređivanje okoline

Unreal Engine je dovoljno dobar za izradu jednostavnih modela, ali neke stvari nije sposoban napraviti. Unreal Engine 5 u sebi automatski ima dodatak Quixel Bridge. Ugrađena kolekcija kvalitetnih modela velika je prednost ovakvog programa. Potrebno je napomenuti da su svi resursi unutar Quixel-a besplatni, jedini uvjet korištenja je da korisnik resurse koristi samo u svrhe Unreal Engine-a.



*Slika 2.7 Quixel Bridge kolekcija resursi*

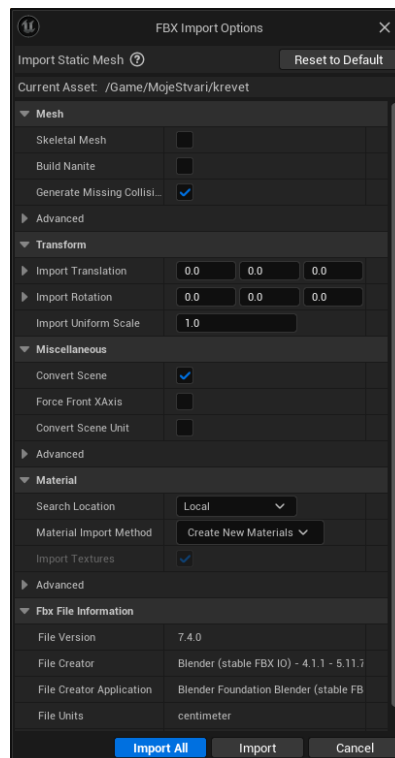
Quixel u sebi sadrži 3D objekte i materijale koje je moguće skinuti i dodati u vlastiti projekt. Ovdje se odabire modularni model kuće koji će biti dio scene na ulici. Modularni objekti služe kao lego kockice pomoću kojih se gradi nešto novo. Tako ovaj modularni paket u sebi sadrži dijelove od kojih se gradi kuća. Odabirom (lijevim klikom) otvara se popis detalja s lijeve strane prozora.



*Slika 2.8 Preuzimanje modela s Quixel Bridge-a u projekt*

U popisu detalja prije preuzimanja moguće je odabrati kvalitetu resursa. Nakon preuzimanja pritiskom na plavi gumb plus ovaj model sada se nalazi u pregledniku sadržaja unutar projekta.

Također je moguće uvesti vlastite modele, teksture i ostale resurse. Model kreveta izrađen je u Blenderu. Model je spremljen kao fbx datoteka te potezom unutar preglednika sadržaja pojavljuje se interaktivni prozor. Klikom na „Import All“ uvode se svi sastavni modeli, materijali i teksture fbx datoteke.



Slika 2.9 Uvođenje modela u projekt

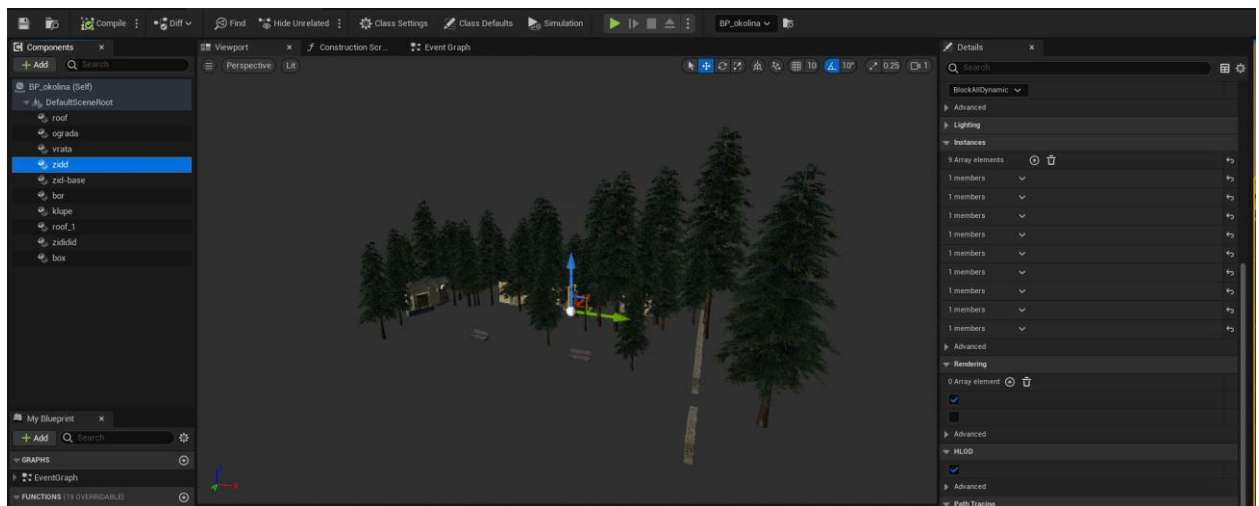


*Slika 2.10 Model kreveta unutar scene i preglednika sadržaja*

Model je sada unutar preglednika. Krevet se sastoji od više dijelova i ako se na scenu uvodi svaki dio posebno oni neće biti na pravim pozicijama. Kako bi se dijelovi kreveta rasporedili onako kao su bili unutar fbx datoteke potrebno ih je zajedno selektirati te tako odvući mišem u scenu.

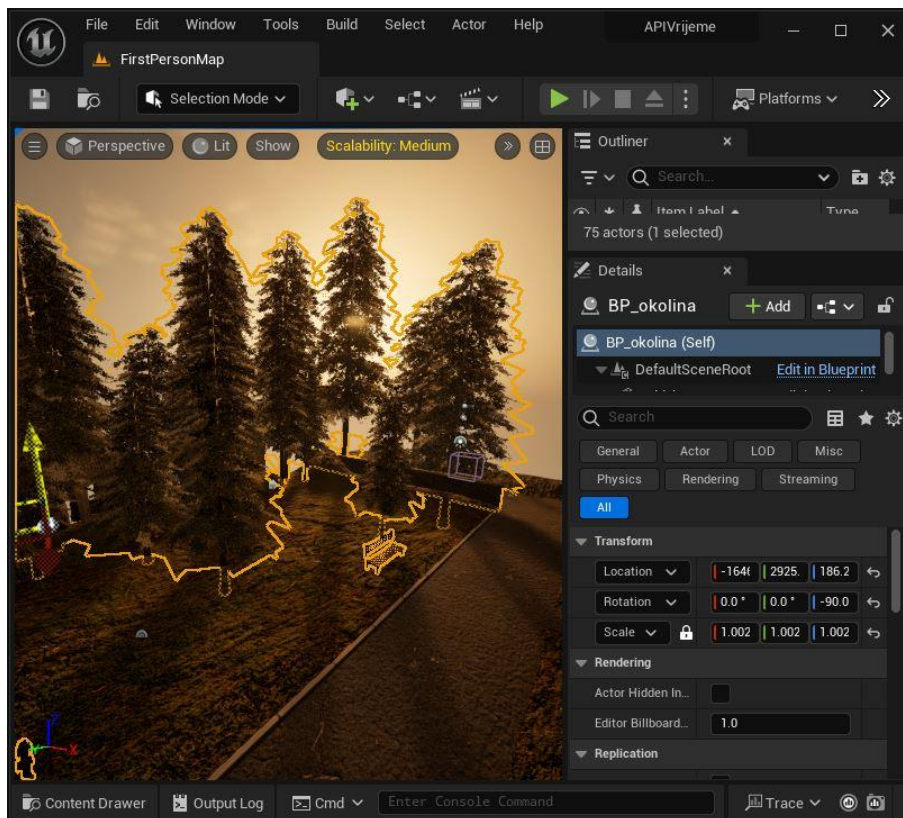
#### **2.1.4. Instanciranje**

Kada na sceni postoje više kopija modela, za optimizaciju izvršavanja aplikacije koristi se instanciranje tih modela. Instanciranje je proces stvaranja više kopija modela bez da se za svaki model kreira zasebni objekt. Time se koristi manje memorije i procesorske snage.



Slika 2.11 Instance modela na ulici

Na slici iznad prikazano je kako izgledaju dodane instance koje zajedno čine dio scene na ulici. Instance 3D modela dodaju se unutar pregleda (*eng. viewport*) Blueprint-a. Nova instanca modela dodaje se pritiskom na zeleni plus simbol s lijeve strane. Pojaviti će se padajući izbornik iz kojeg je onda moguće odabrati dodati Instanced Static Mesh, to će biti instanca modela. S desne strane prozora nalazi se popis detalja. U njemu se nalazi slot gdje se odabire 3D model (*eng. static mesh*). U odjeljku Instance array klikom na plus ikonu dodaje se kopija odabranog modela. Model je nakon toga vidljiv u pregledu Blueprinta i njemu je moguće mijenjati veličinu, poziciju i rotaciju. Postupak se ponavlja za svaku kopiju modela. Za svaki model koji se želi kopirati dodaje se Instanced Static Mesh i cijeli postupak se ponavlja. Da bi instance bile vidljive njihov Blueprint se povlači na scenu projekta.



*Slika 2.12 Instance postavljene na sceni*

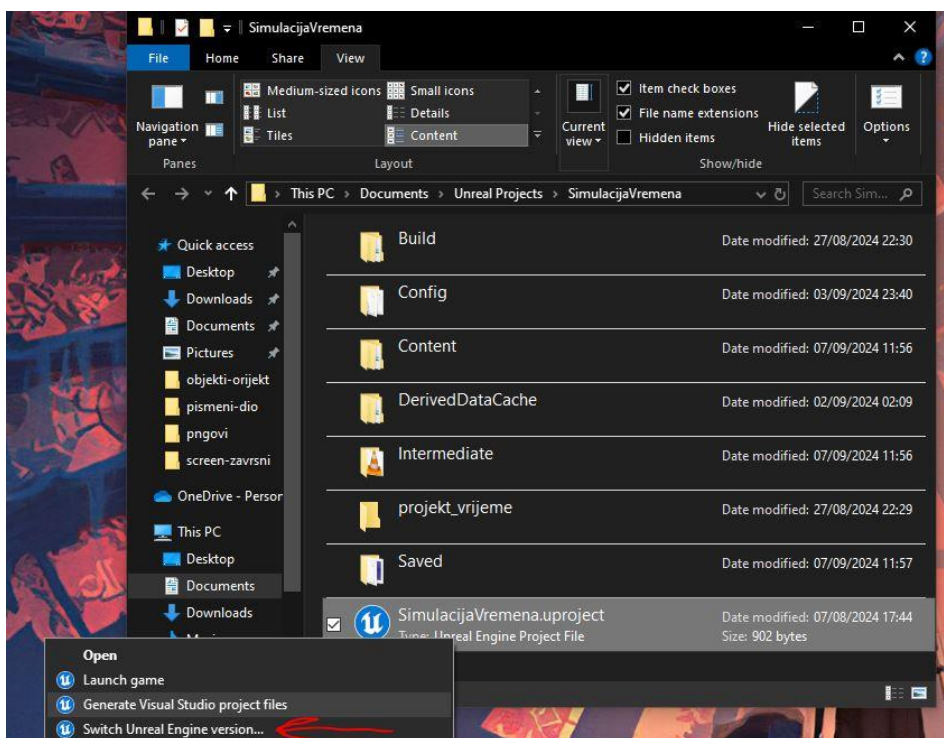
Blueprint sa instancama također je moguće transformirati kao i svaki objekt na sceni. Isto tako je naknadno moguće micanje modela sada kada su u sceni kako bi se bolje prilagodili izrađenoj mapi. Bitno je spomenuti da instanciranje modela nema nikakvog utjecaja na materijal originalnog modela, materijal će se i dalje jednako ponašati i davati će iste rezultate kao da je i model postavljen u mapu sam po sebi.



### 3. Migracija projekta iz verzije Unreal Engine 4 u verziju Unreal Engine 5

Ovaj projekt je započet u Unreal Engine-u 4 jer VaRest još nije bio dostupan za verziju 5. U međuvremenu je dodatak ažuriran te ga je moguće koristiti u verziji 5.3 Unreal Engine-a.

Prvi korak je skinuti novu verziju UE, a zatim pronaći gdje se projekt nalazi na računalu.



Slika 3.1 Mijenjanje verzije programa

Desnim klikom na sam projekt odabere se opcija „Izmjeni verziju Unreal Engine-a“. Pojaviti će se prozor s padajućim izbornikom koji sadrži instaliranje verzije UE dostupne na računalu. Odabere se željena verzija.



Slika 3.2 Odabir verzije programa




Klikom na „OK“ projekt je konvertiran za rad u odabranoj verziji Unreal Engine-a.



## 4. Kreiranje Open Weather profila i generiranje API ključa

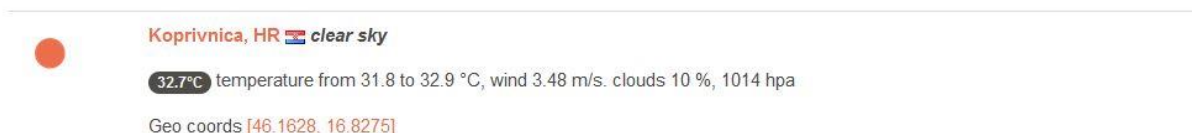
Kako bi se mogako koristiti API Open Weather servisa potrebno je generirati jedinstveni API ključ. Sva komunikacija sa servisom traži prisustvo API ključa kako bi se potvrdilo da je zahtjev legitiman.

U traci za navigaciju stranice nalazi se meni čiji je naslov korisničko ime profila, unutar padajućeg izbornika postoji link „My API keys“. Gumb otvara stranicu gdje su izlistani već postojeći ključevi unutar tablice i gdje je moguće generirati i imenovati nove. Kod besplatne moguće je imati samo jedan aktivan ključ.

Key	Name	Status	Actions	Create key
	KC_vrijeme	Active	 	<input type="text" value="API key name"/> <input type="button" value="Generate"/>

Slika 4.1 Sučelje za kreiranje ključa

Nakon što se generirao ključ potrebno je saznati geografsku duljinu i širinu mjesta za koju nam trebaju meteorološki podaci. Odmah pokraj izbornika nalazi se pretraživač u kojeg je dovoljno upisati ime grada i stranica će izbaciti trenutnu prognozu uz koordinate grada.



Slika 4.2: Rezultat pretraživanja Koprivnice

Na Open Weather stranici nalazi se URL do API krajnje točke s vitičastim zagradama (*eng. placeholder*) u koje je potrebno umetnuti vlastite podatke. „{Lat}“ i „{lon}“ dijelovi su koordinata grada dok je „{API key}“ potrebno nadomjestiti s aktivnim ključem koji se prethodno generirao.

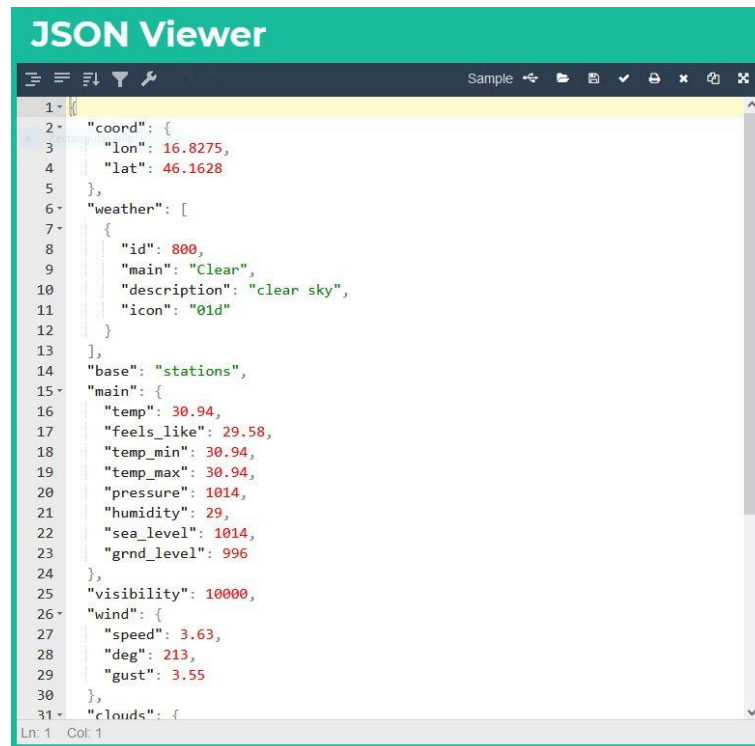
```
API call
```

---

```
https://api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}&appid={API key}
```

Slika 4.3 URL za poziv API-u

Za detaljniji pregled dohvaćenih informacija koristi se besplatni alat JSON Viewer. Kada se prvi puta dođe na stranicu otvara se dijaloški prozor. Iznad „Load Data“ u tekstni okvir unosi se URL s popunjenim podacima. JSON Viewer prikazuje podatke dohvaćene sa servisa.



```
1 {
2   "coord": {
3     "lon": 16.8275,
4     "lat": 46.1628
5   },
6   "weather": [
7     {
8       "id": 800,
9       "main": "Clear",
10      "description": "clear sky",
11      "icon": "01d"
12    }
13  ],
14  "base": "stations",
15  "main": {
16    "temp": 30.94,
17    "feels_like": 29.58,
18    "temp_min": 30.94,
19    "temp_max": 30.94,
20    "pressure": 1014,
21    "humidity": 29,
22    "sea_level": 1014,
23    "grnd_level": 996
24  },
25  "visibility": 10000,
26  "wind": {
27    "speed": 3.63,
28    "deg": 213,
29    "gust": 3.55
30  },
31  "clouds": {
```

Slika 4.4 JSON odgovor na stranici JSON viewer

Iz slike 4.4 vidi se da je ključna informacija o trenutnoj prognozi identifikacijski broj („ID“). Putem njega je moguće saznati koje specifično vrijeme je trenutno ponuđeno, jer „ID“ implicira da je broj jedinstven za specifični entitet. Open Weather stranica nudi listu vremenskih prilika s odgovarajućim identifikacijskim brojevima.

#### Group 800: Clear

800	Clear	clear sky	● 01d ● 01n
-----	-------	-----------	----------------

#### Group 80x: Clouds

801	Clouds	few clouds: 11-25%	☀ 02d ☀ 02n
802	Clouds	scattered clouds: 25-50%	☁ 03d ☁ 03n
803	Clouds	broken clouds: 51-84%	☁ 04d ☁ 04n
804	Clouds	overcast clouds: 85-100%	☁ 04d ☁ 04n

*Slika 4.5 Dio liste vremenskih prilika Open Weather stranice*

Pomoću ovih podataka kreirat će se tablica unutar Unreal Engine-a, a nazivi redova će biti identifikacijski brojevi vremenskih prilika. To će osigurati da svaka komponenta vremena koja će odgovarati tipu podatka unutar tablice reagirati na točno specifično vrijeme.

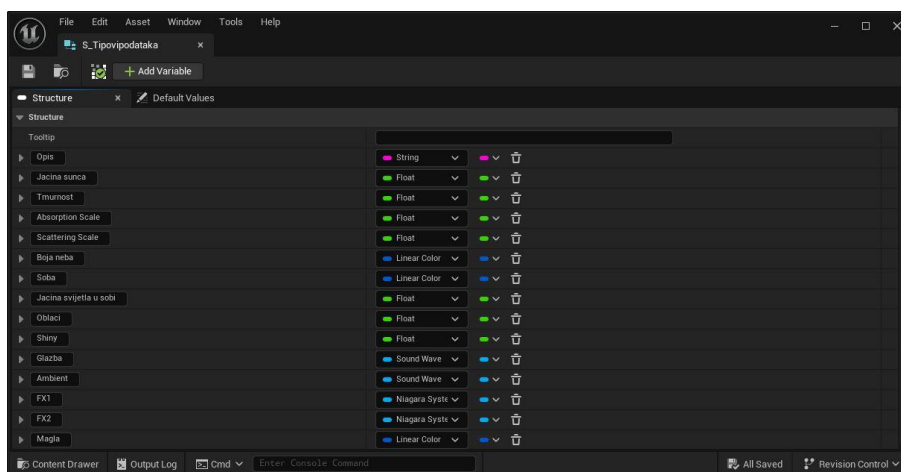
## 5. Tablica podataka

Tablica podataka, tj. Data Table kako je nazvana u UE, služi kao strukturiran način pohrane podataka. Bitno je napomenuti da je iz nje moguće samo čitati, podaci unutar tablice se mijenjaju samo kada se tablica kreira u svrhu programa. Korisnik igre, simulacije ili aplikacije nije u mogućnosti mijenjati podatke, niti tipove podataka unutar tablice.

### 5.1. Struktura tablice

Da bi se mogla izraditi tablica podataka, potrebno je najprije izraditi strukturu koja u sebi sadrži nazive varijabli i njihove tipove, u kakvom obliku će podaci biti zapisani u pojedinoj varijabli.

Izbornik sadržaja (*eng. Content browser*) unutar Unreal Engine-a sadrži sve elemente projekta. Elemente je moguće strukturirano raspodijeliti i organizirati po mapama (*eng. folder*). Najlakši način kreiranja novih elemenata projekta je desnim klikom unutar izbornika sadržaja. Dovoljno je navigirati do dijela gdje želimo novi element i desnim klikom pojavljuje se skočni izbornik gdje je potrebno samo odabrati element koji se želi izraditi.



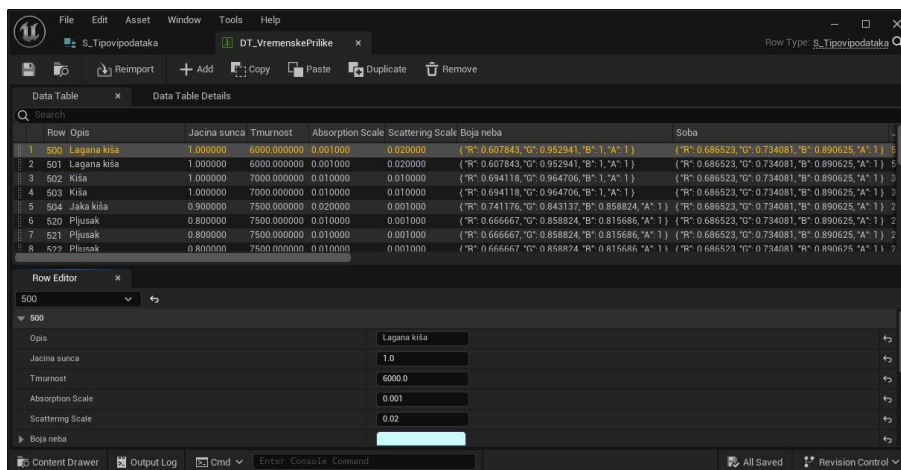
Slika 5.1 Struktura tablice s varijablama

Ova struktura će se zvati „S\_Tipovipodataka“. U UE nazivanje elemenata je proizvoljno, ali je ustaljena praksa korištenja prefiksa kod kreiranja naziva elemenata. Iz prefiksa je odmah vidljivo koji je to tip elementa.

Podaci u strukturi su varijable i svaka varijabla predstavlja jedan stupac unutar tablice. Vrijednost varijable može biti bilo kojeg oblika, može biti tekst, cijeli broj, decimalni broj, boja, specifičan oblik svjetla, zvuk i slično. Varijable se dodaju klikom na zeleni plus simbol na vrhu prozora i zatim se varijable nazivaju i bira se njihov tip. Isto tako je opcija postaviti početne vrijednosti koje će ispunjavati ćelije tablice osim ako se vrijednosti ne promijene. To je praktično

kada se radi s velikim brojem podataka. Na taj način se izbjegava da neka varijabla ne ostane nedefinirana. Novo izrađena struktura predstavlja varijable čiji tipovi odgovaraju sastavnim komponentama vremenskih prilika koje će simulacija prikazivati.

Zatim slijedi izrada same tablice. Postupak kreiranja je isti, samo što je potrebno kliknuti na opciju „Data Table“.



Slika 5.2 Tablica vremenskih pojava

Tablica će se zvati „DT\_VremenskePrilike“. Novi red dodaje se klikom na simbol plus iznad tablice i time se stvara novi red u tablici. Svaki red tablice predstavlja jednu vremensku pojavu te se tipovi podataka popunjuju unutar redova sukladno tome. Razlog zašto u strukturi ranije nije uključen identifikacijski broj jest što se pozivi na redove tablica vrše putem imena reda (eng. row name). Program generira ime reda automatski, no njega je moguće promijeniti dvoklikom i upisivanjem željene vrijednosti.

## 6. Sastavni dijelovi vremenskih prilika

U prethodnom poglavlju je spomenuto da struktura koja definira podatke u tablici je sastavljena od varijabli različitih tipova koje odgovaraju vremenskim prilikama. Tipovi varijabli najbolje će se objasniti putem vremenskih pojava koje za su namijenjene. Najjednostavniji oblik vremena koje nudi Open Weather servis je vedrina neba.

### 6.1. Sunce, nebo i oblaci

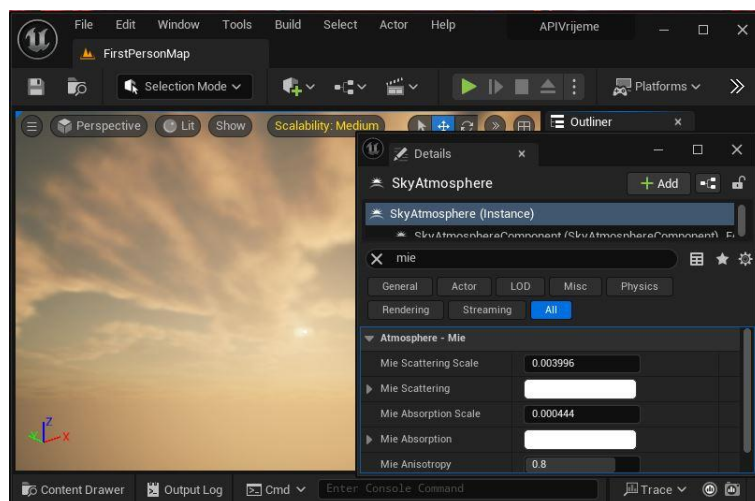
Unreal Engine sastavljen je od brojnih elemenata koji su tamo nazvani „akterima“ (*eng. actors*). Akteri unutar Unreal Engine-a predstavljaju svaki objekt koji je moguće postaviti u scenu i manipulirati u 3D prostoru gotovo kao što bi se predmetima upravljalo u stvarnom 3D prostoru. Jedni od očitijih aktera koji se mogu manipulirati podacima iz tablice su sunce, nebo i oblaci.

#### 6.1.1. Sunce

Primarni način manipuliranja sunca je toplina i intenzitet svjetla, koji su oboje skalarne vrijednosti, stoga će tip podataka biti float (decimalni broj). Osnovne vremenske prilike Open Weather opisuje u pet intenziteta, od vedrog neba, do potpune naoblake. Prema tome se određuje intenzitet i toplina svjetla. Najefikasniji način odabira idealne jačine i topline svjetla je uzeti najvedriju i najoblačniju opciju te intenzitete svjetla između uskladiti s njima.

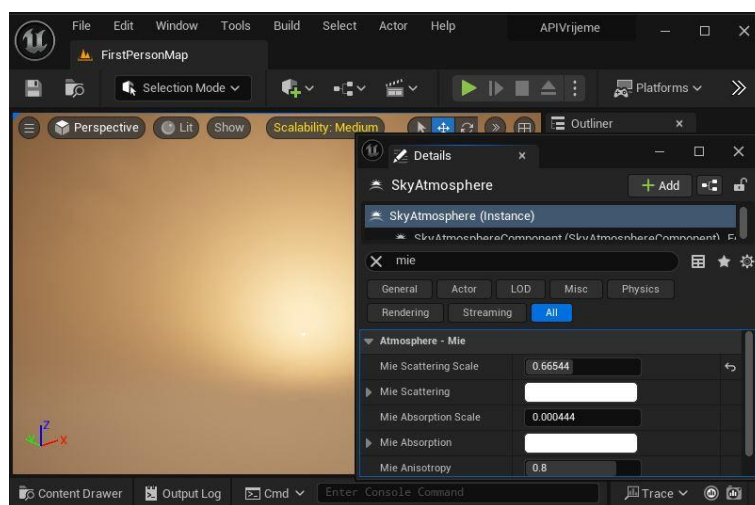
#### 6.1.2. Nebo

Nebo je unutar Unreal Engine-a prikazano putem aktera „atmosfera neba“ (*eng. Sky atmosphere*). Ugođaj vedrine ili tmurnosti vremena pomoću ovog elementa moguće je definirati njegovom bojom i načinom na koje se svjetlo ponaša kada prolazi kroz atmosferu; vrijednost raspršenja (*eng. Scattering scale*) i vrijednost apsorpcije (*eng. Absorption scale*).



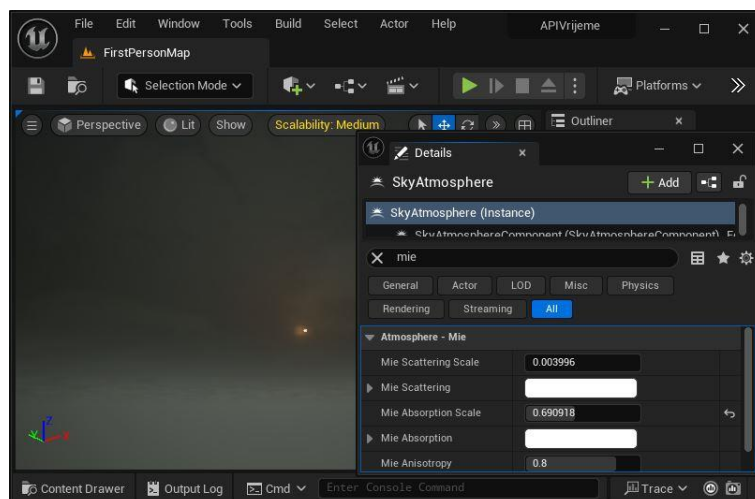
*Slika 6.1 Izgled neba sa standardnim raspršenjem i apsorpcijom*

Vrijednost raspršenja više utječe na boju i njezin intezitet. U prirodi, nebo izgleda plavo zbog duljine valova svjetlosti plave boje koja se najviše raspršuje. U Unreal Engine-u veći brojevi povećavaju jačinu raspršenja što čini svijetlo mekšim, ali i utječe na boju neba, s toga je potrebno pronaći balans.



*Slika 6.2 Izgled neba kada se pojača raspršenje*

Vrijednost apsorpcije simulira atmosferino upijanje svjetlosti. Veći brojevi čine nebo tamnijim, bez da se intezitet svijetla sunca mijenja što daje veću kontrolu nad ugođajem i vidljivošću okoline. Kontrola boje neba reagira na veoma očit način, no vedrina će biti definirana saturacijom boje. Što je nebo tmurnije, to će boja neba biti manje saturirana.



*Slika 6.3 Izgled neba kada se pojača apsorpcija*



*Slika 6.4 Skala apsorpcije i raspršenja*

Iz slike iznad vidljivo je da su vrijednosti apsorpcije i raspršenja brojevi veoma malih vrijednosti pa svaki pomak u vrijednosti drastično utječe na izgled neba.

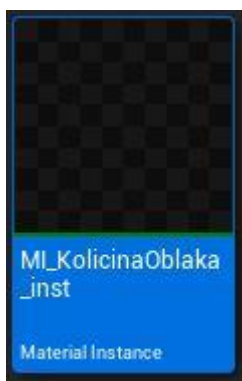


### 6.1.3. Gustoća oblaka

Gustoća oblaka nije prilagodljiva na lako dostupan način mijenjanjem različitih vrijednosti koje bi tablica mogla koristiti. Potrebno je napraviti materijal koji će se primijeniti na komponentu oblaka koji se mijenja u skladu s broječanom varijablom iz tablice koja mijenja gustoću oblaka.

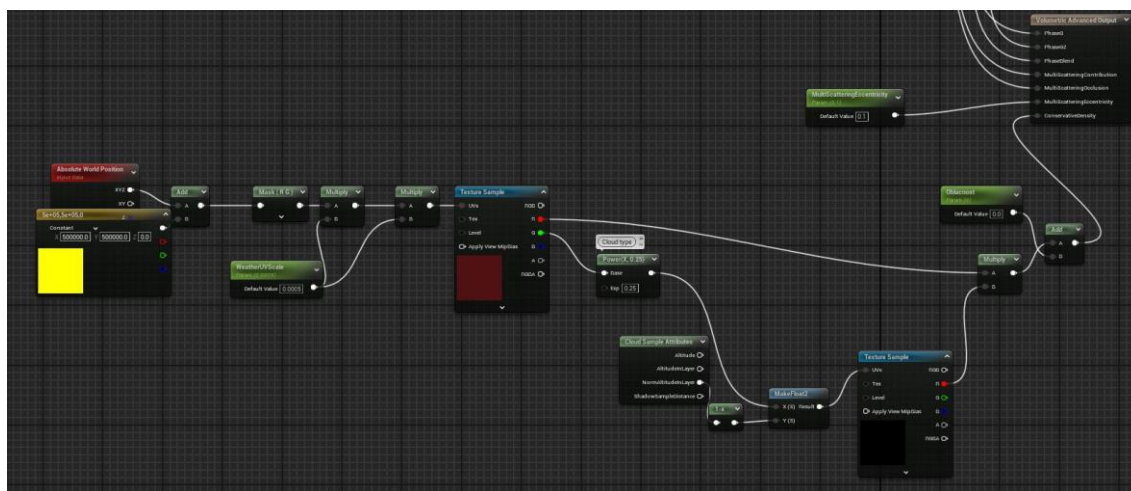
Kreiranje instanci postojećih materijala nudi nove mogućnosti prilagodbe vremenu. Prvi korak je duplicirati već postojeći materijal za oblak te na toj kopiji raditi sve potrebne izmjene. Materijal za oblak unutar Unreal Engine-a je vrlo kompleksan te nudi puno načina za modificiranje.

Materijal je dupliciran, preimenovan u „M\_Oblacnost“, desnim klikom na novi materijal i odabirom „kreiraj novu instancu materijala“ nastaje nova instanca bazirana na tom materijalu.



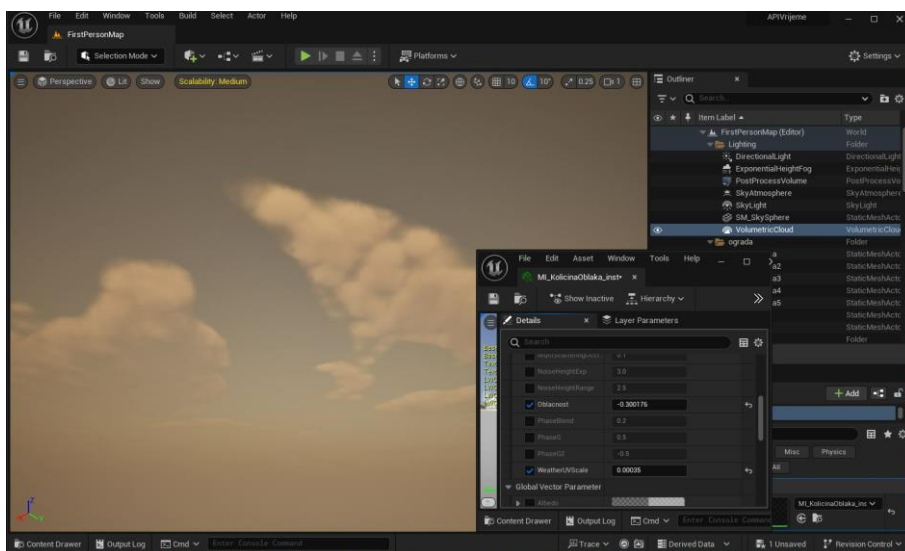
Slika 6.5 Instanca materijala za oblake

Novi materijali se primjenjuju na objekte tako da se klikom na objekt on selektira i u prozoru s detaljima gdje se nalaze podaci o materijalu klikne na padajući izbornik. On sadrži sve moguće primjenjive materijale unutar projekta, te se selektira željeni materijal, u ovom slučaju „MI\_KolicinaOblaka\_inst“.

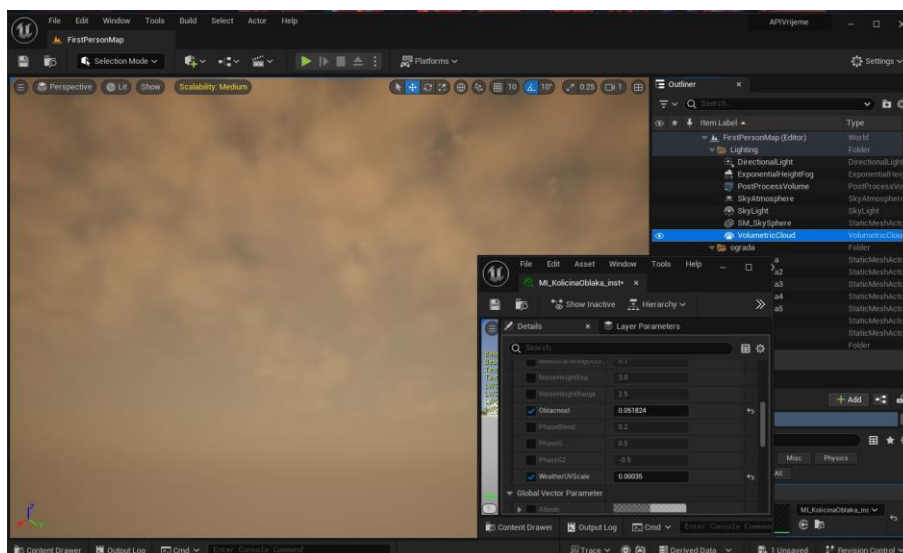


Slika 6.6 Dio logike materijala oblaka

Iz priložene slike vidljiva je kompleksnost originalnog materijala. Isto tako vidljivo da koristi šum (*eng. noise map*) da kontrolira prikaz oblaka. Čvorovi su spojeni na „Conservative Density“ što određuje gustoću oblaka. Mijenjanjem te vrijednosti moguće je mijenjati gustoću oblaka. Noise mape koriste se za generiranje slučajnih ili nasumičnih uzoraka često kako bi efekti izgledali prirodnije. To je tekstura koja generira nasumične vrijednosti (od crne do bijele, ili u ovom slučaju od žute do crvene) te se u ovom kontekstu odnosi na gustoću oblaka i koliko svjetla prolazi kroz njih. Da se dobije kontrola nad tim parametrom dovoljno je dodati jednostavan čvor koji množi vrijednosti, Multiply node. On koristi A i B vrijednost. B vrijednost će se spojiti na rezultat noise mape, a A rezultat se spaja na čvor koji je konstanta. Taj čvor je moguće kreirati pritiskom tipke 1 i klikom na prazno mjesto u Blueprint-u. Ta konstanta će se nazvati „Oblačnost“ i ona je sada dostupna kao parametar kojeg je moguće mijenjati.



Slika 6.7 Parametar oblačnosti je podešen na -0.3



Slika 6.8 Parametar oblačnosti je podešen na 0.05

Određeni parametri oblačnosti slijede redom od najmanje do najviše oblačnog: -1, -0.6, 0, 0.01, 0.03. To je pet varijanti za pet različitih intenziteta oblačnosti.

#### 6.1.4. Ugođaj unutar interijera

Sastavne komponente ugođaja unutar sobe su dva svijetla tipa točkasto svijetlo (*eng. Point light*). Ono se kontrolira brojevnim vrijednostima i bojom. Intenzitet svijetla odgovara vedrini vremena dok boja svijetla za sada ostaje ista. Boja će kasnije biti drugačija u drugim vremenskim pojavama.

Time je izrađena baza za prikazivanje i ostalih vremenskih prilika. Glavni dijelovi će biti modelirani prema ovim postavkama, a po potrebi će se postavke mijenjati kako bi ugođaj vremenske prilike bio prikladniji.

## 6.2. Niagara sustav čestica

Niagara je sustav vizualnih efekata koje je moguće izvesti s dodatnim funkcionalnostima bez dodatnog programiranja. On je fleksibilan i može se prilagoditi potrebama programa. Ponaša se kao spremnik za sve što je potrebno za određeni efekt. Moguće je dodavati različite elemente koji zajedno tvore jedan vizualni efekt.

Sustavi su sastavljeni od emitera. To su pozicije gdje se čestice generiraju i moguće je postaviti kako, kada i koliko ih se generira. Također je moguće kontrolirati se sa česticama događa kada se stvore ili sudare i koliko dugo su vidljive.

### 6.2.1. Snijeg

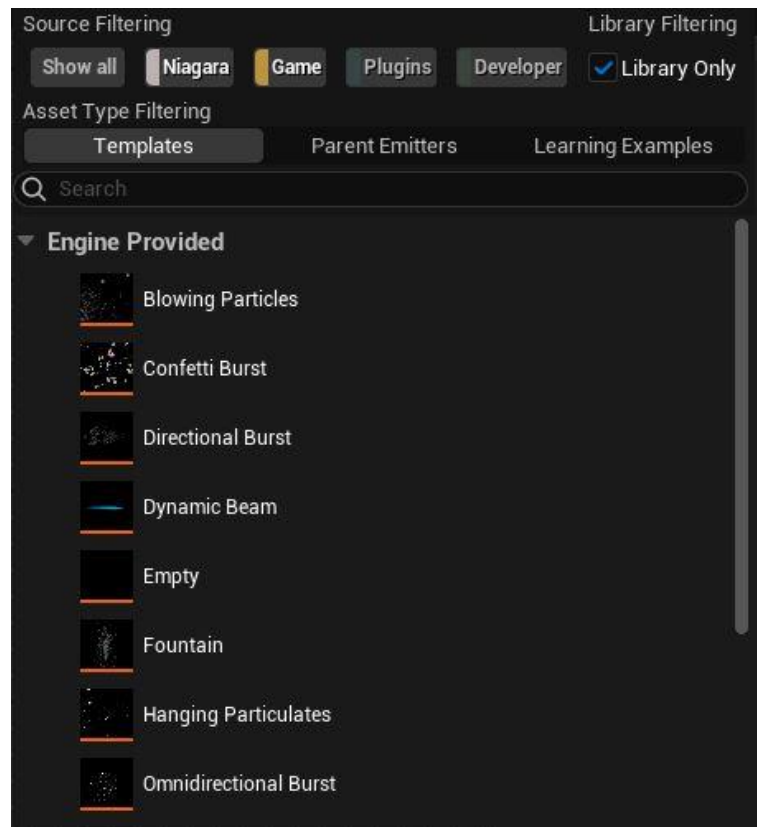
Snijeg je jedan od jednostavnijih Niagara sustava izrađenih u ovom projektu. Služiti će kao uvod u modeliranje emitera.

Novi Niagara sustav izrađuje se kao i svaki akter u programu; desni klik i odabir opcije željenog aktera. Ovaj emiter biti će baza ostalim intenzitetima snijega. Open Weather servis nudi informacije za lagan, umjeren i jak snijeg. Emiter se naziva „NS\_LSnijeg“.



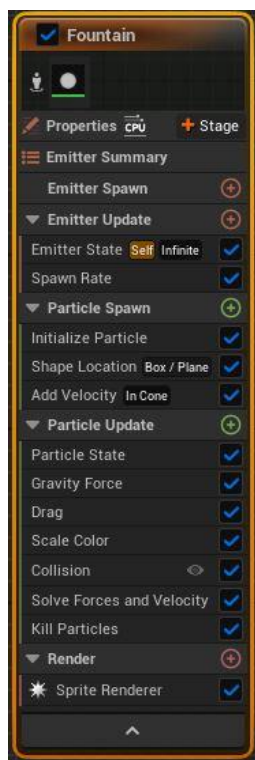
*Slika 6.9 Sustav čestica za snijeg*

Kada se novi sustav otvori potrebno je dodati u njega novi emiter. Desnim klikom na prazno mjesto i klikom na opciju dodaj novi emiter (*eng. Add new emitter*) otvara se izbornik s njihovim opcijama.



*Slika 6.10 Opcije emitera*

Za potrebe snijega prikladna je opcija fontane (*eng. fountain*). Taj emiter se sada nalazi u pregledu sustava i način pregleda njegovih čestica je moguće kontrolirati.



Slika 6.11 Kreirani emiter s postavkama

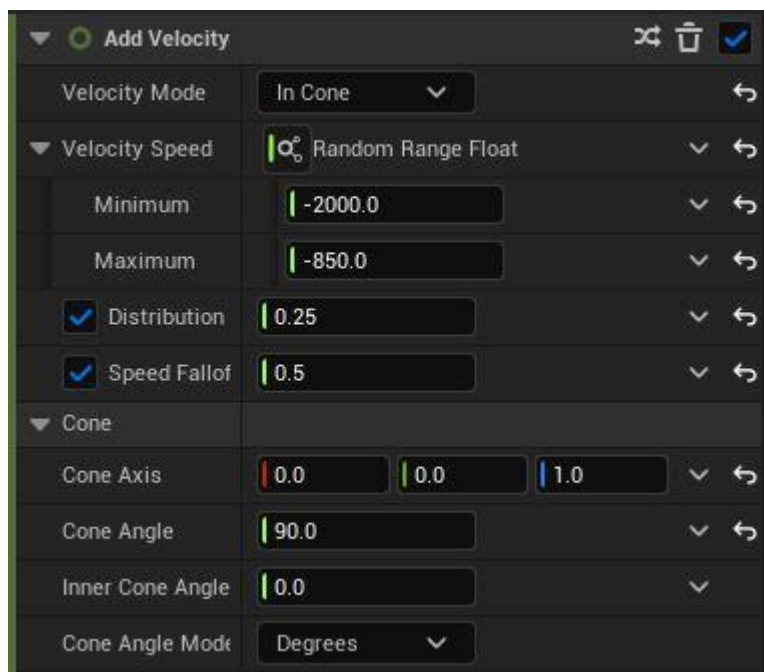
Prva bitna postavka je postavljanje područja gdje će se čestice pojavljivati, to je dostupno pod dijelom gdje se kontrolira stvaranje čestica (*eng. Particle spawn*). Područje i njegov oblik i veličina nalaze se pod oblikom lokacije (*eng. Shape location*). Klikom na taj izbornik otvaju se detalji s desne strane pregleda.



Slika 6.12 Oblik lokacije emitera

Zbog oblika mape koja je kreirana najviše ima smisla odabrati kocku (*eng. box*), te njezine dimenzije se prilagođavaju veličini mape tako da čestice pokrivaju cijelu mapu.

Brzina čestica pod „dodaj brzinu“ (*eng. Add velocity*) kontrolira brzinu padanja, ali i smjer padanja čestica. Kako bi čestice padale, brzina se postavlja na negativnu vrijednost, a način brzine je unutar stožca (*eng. Velocity in cone*). Moguće je postaviti kut stožca te će to utjecati na kut putanje kretanja čestica. Time se čestice ne kreću pod istim kutem nego slijede rubove stožca i slijede liniju padanja pahuljica sličnijim realnim uvjetima.



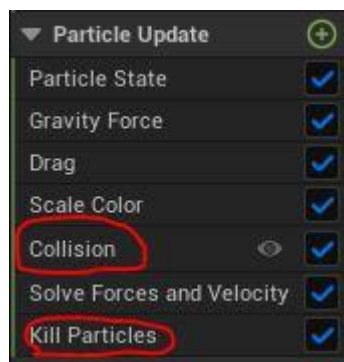
Slika 6.13 Brzina i smjer čestica

Količina čestica ovisi o brzini, ali i o broju stvaranja čestica (eng. *Spawn rate*). To je skalarna veličina koja će služiti za tri različita intenziteta padanja snijega. Veličina 400 znači da se u jednoj sekundi stvori 400 čestica. Još jedan od faktora je i trajanje čestica, tj. „život čestica“ (eng. *Particle lifetime*) kojeg je uvijek poželjno postaviti nasumično, kako bih pojava izgledala što prirodnije. Svaki puta kada je odabrana nasumičnost, određuje se minimalne i maksimalne vrijednosti, a generirane vrijednosti će se unutar tih granica.



Slika 6.14 Trajanje čestica

Kako bi se spriječilo da čestice prolaze kroz čvrste objekte potrebno je postaviti koliziju (eng. *collision*). Ponašanje čestica rješava se u dijelu ažuriranja čestica (eng. *Particle update*).



Slika 6.15 Ažuriranje čestica

Kako bi kolizija funkcionirala, potrebno je dodati događaj kolizije (*eng. Collision event*); Particle update > Collision event. Sada će se čestice sudarati sa svime što ima definiranu koliziju. Nije poželjno da čestice odskakuju nego da nestanu u trenutku kada se sudare s površinom. Dodaje se opcija za uklanjanje čestica (*eng. Kill particles*) i njezina funkcija će biti postavljena na „dogodio se sudar“ (*eng. Has colided*). To je boolean vrijednost. Boolean vrijednosti mogu imati samo dvije opcije: istina i laž. Tako će program donijeti odluku ukoliko se čestca sudarila i treba li je ukloniti.

### 6.2.2. Kiša

Kiša će biti nešto kompleksniji sustav koji će se sastojati od dva emitera.

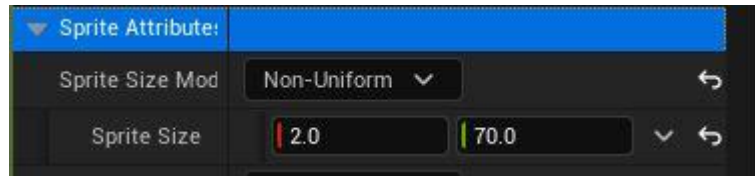


Slika 6.16 Sustav čestica za kišu

Prvi emiter je istog tipa kao i snijeg uz nekoliko manjih promjena. Ovdje oblik čestica neće bit okrugao. Čestice trebaju biti izdužene što se postavlja pod atributima spritea (*eng. Sprite attributes*). Sprite je 2D slika koja se koristi u video igrama i aplikacijama uglavnom za prikaz

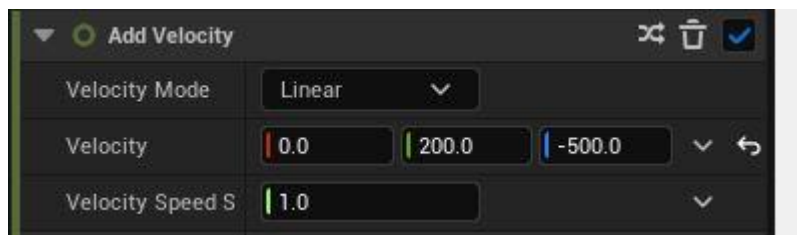


jednostavnijih grafičkih elemenata. Opcija non-uniform znači da proporcije neće biti jednake, dakle x i z os čestice će biti različitih vrijednosti.



Slika 6.17 Veličina čestice

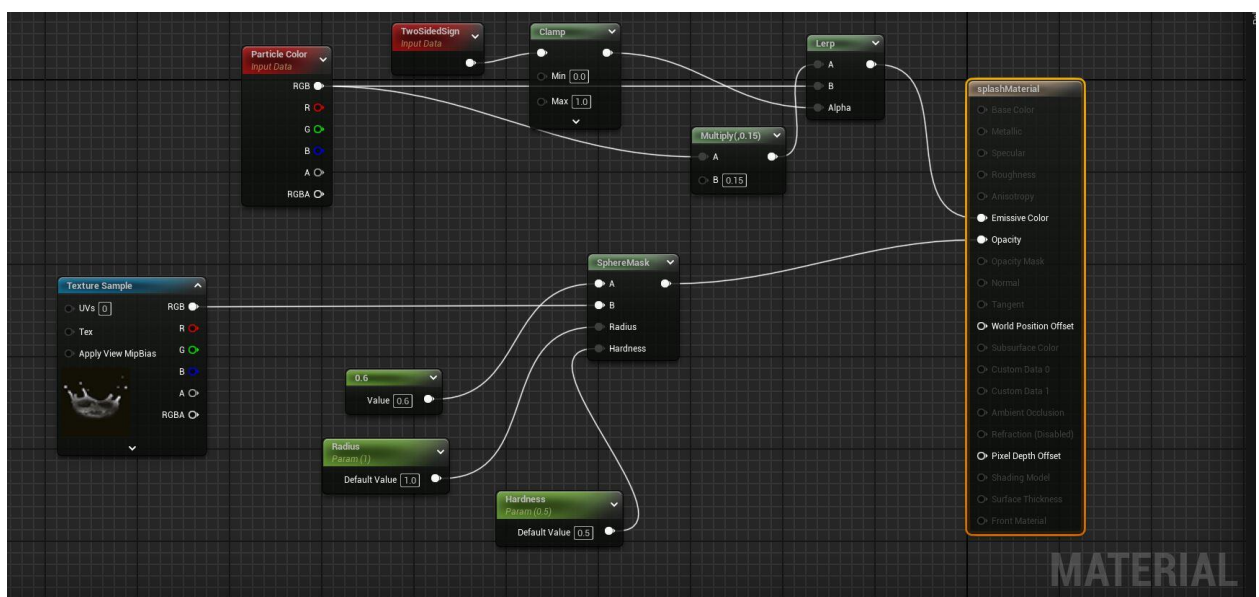
Ovdje je također način simuliranja brzine linearan, jer se čestice trebaju kretati u istom smjeru.



Slika 6.18 Smjer i brzina kretanja čestica kiše

Veličine 200 i -500 postižu to da se čestice kreću u koso, a skalar brzina ubrzanja (eng. Velocity speed) je broj koji će se povećavati u svrhu intenziteta kiše; slaba, umjerena i jaka kiša.

Dodatni emiter koji je potreban su lebdeće čestice (eng. Hanging particulates). Za njih je potrebno napraviti novi materijal.

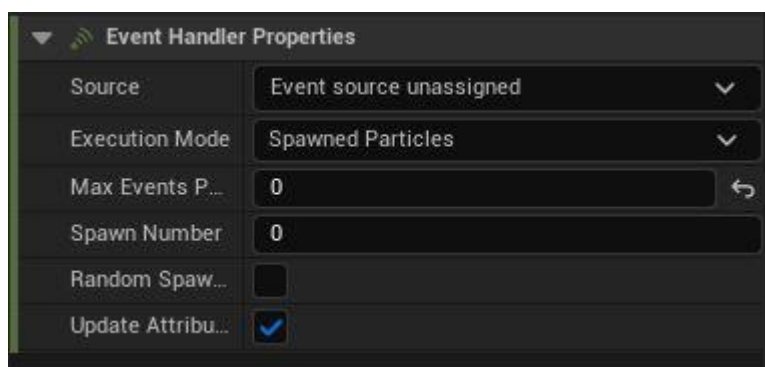


Slika 6.19 Materijal za odbijanje kapi kiše

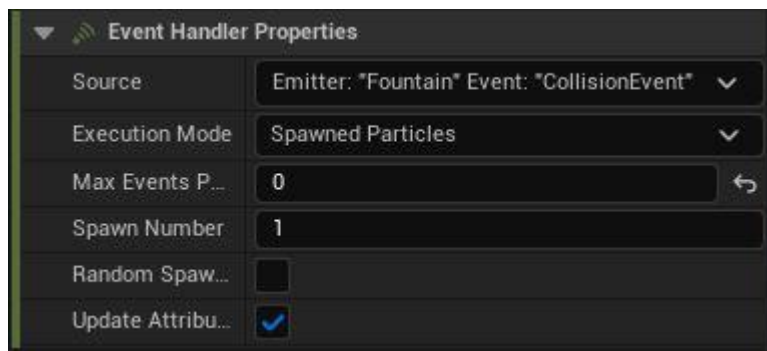
Unutar materijala dodani su razni čvorovi. Čvor za sam izgled čestice koji mora biti crno-bijeli.

Čvor sferna maska (*eng. Sphere mask*) stvara kružni gradijent na temelju udaljenosti od određene točke, tj. centra. Maske funkcioniraju isto tako da ono što je crno na teksturi je nevidljivo, a vidljivo je bijelo. Čvor množenja (*eng. multiply*) množi dvije ulazne vrijednosti te se ponaša kao obična matematička operacija. Ograničenje (*eng. clamp*) ograničava vrijednost na određen raspon zadane minimalne i maksimalne vrijednosti. Čvor „lerp“ interpolira između dvije vrijednosti na temelju treće koja se nalazi u rasponu od 0 do 1. Kombinacijom tih čvorova dobio se materijal izgleda kapi kiše koji je prsnuo.

Kako bi ovi materijali mogli vršiti interakciju jedan s drugim, potrebno im je dodati kolizijske događaje. Svaki emiter ima upravitelj događaja (*eng. Event handler*). Emiter koji oponaša kišu mora dobiti događaj i emiter koji oponaša raspršenje.



Slika 6.20 Događaj za kišu



Slika 6.21 Događaj za raspršenje

Izvor događaja (*eng. source*) za kišu je vlastit događaj te se ne mora pridodati, no događaj za raspršenje je događaj sudara fontane, tj. kiše. Broj pojave (*eng. Spawn number*) određuje koliko čestica nastaje na mjestu kolizije. Sada svaki puta kada se kap kiše sudari sa nečim što ima postavljenu koliziju pojaviti će se prsnuta kap kiše.

Postoji posljedica kiše koja će se isto prikazati u projektu, a to je količina vlažnosti površine koju ona dotakne.

### 6.2.3. Grmljavina

Grmljavina je zvučni i vizualni fenomen koji se može stvoriti unutar sustava čestica. Njihova konfiguracija se razlikuje u odnosu na kišu ili snijeg.

Za početak, kreira se materijal za munju koja će se pojavljivati. Proces kreiranja je jednak raspršenju kapi kiše. Razlika je tekstura i vrijednost drugih parametara.

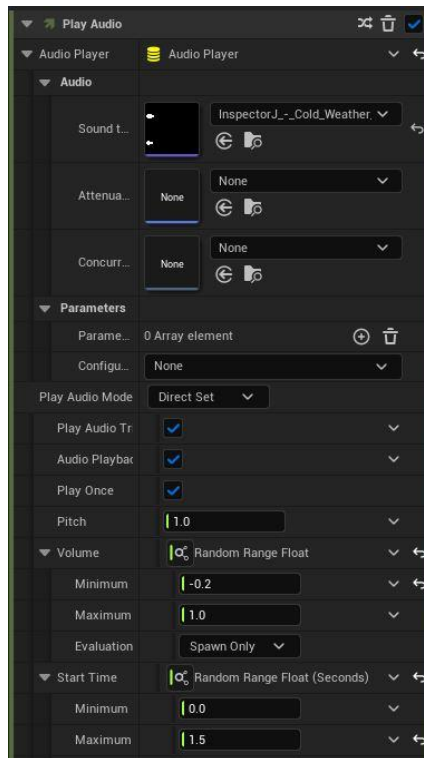
Kod munje se pojavljuje samo jedna čestica unutar jednog emitera. U sustavu se nalaze dva emitera koji simuliraju grmljavinu.



Slika 6.22 Pojavljivanje groma

Munja će se pojavljivati manje nego jednom u sekundi, a vjerojatnost pojavljivanja (*eng. Spawn probability*) je 30%. Ove postavke pridonose nasumičnosti pojavljivanja groma.

Česticama je moguće pridodati zvuk unutar događaja. Potrebno je dodati događaj „pokreni zvuk“ (*eng. Play audio*). U detaljima događaja dalje je moguće odabrati zvuk, jačinu zvuka i kada zvuk započinje. Ovdje je bitno spomenuti da Unreal Engine barata s wav formatom datoteka.



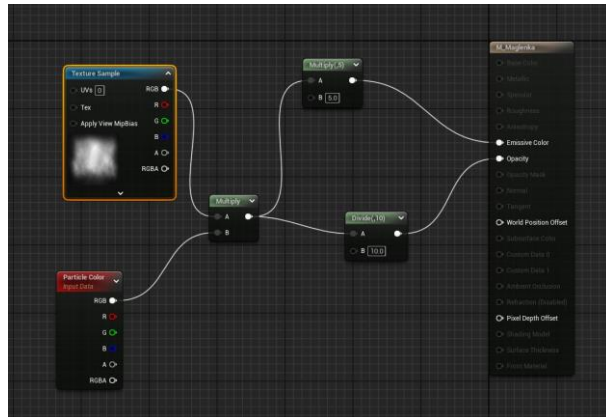
Slika 6.23 Zvuk grmljavine

S obzirom da je optimalno koristiti jedan zvuk, monotoniju je moguće razbiti tako da se napravi pomak u odnosu na početak zvučnog zapisa (*eng. Start Time*) i njegova glasnoća (*eng. Volume*). Ti parametri se mijenjaju nasumično između dodijeljenih minimalnih i maksimalnih vrijednosti.

## 6.2.4. Magla

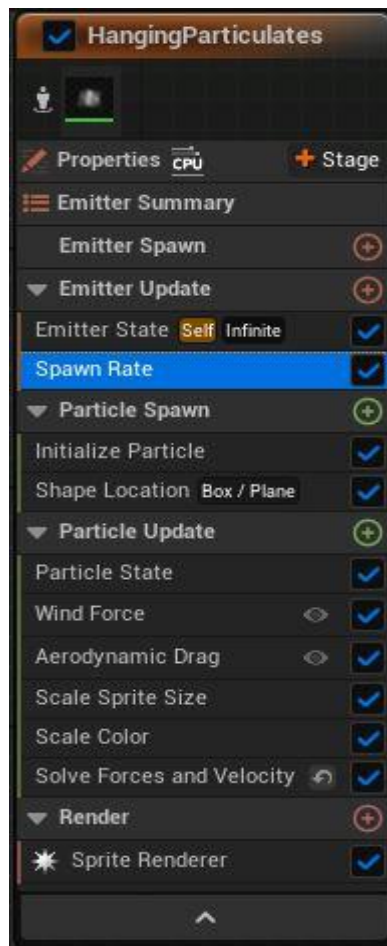
Magla u Niagara sustavu izrađena je pomoću Hanging Particulates emitera. Na čestice ne utječe gravitacija i one lebde u zraku, što odgovara vizualnom ponašanju magle.

Izrađen je materijal koji će biti primijenjen na čestice magle.



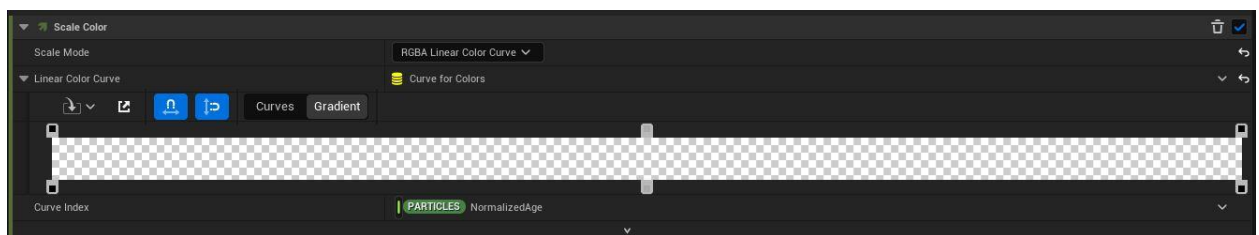
Slika 6.24 Materijal za maglu

Dodana je crno-bijela tekstura, tj. slika oblaka magle te je materijalu dodan čvor za boju čestica kao bi se čestici mogla pripisati prirodniija boja magle i također je moguće mijenjati prozirnost čestica kod njihovog pojavljivanja i nestajanja.



Slika 6.25 Sustav čestica za maglu

Brzina pojavljivanja je znatno manja za razliku od kiše ili snijega te je također „životni ciklus“ čestica dulji.

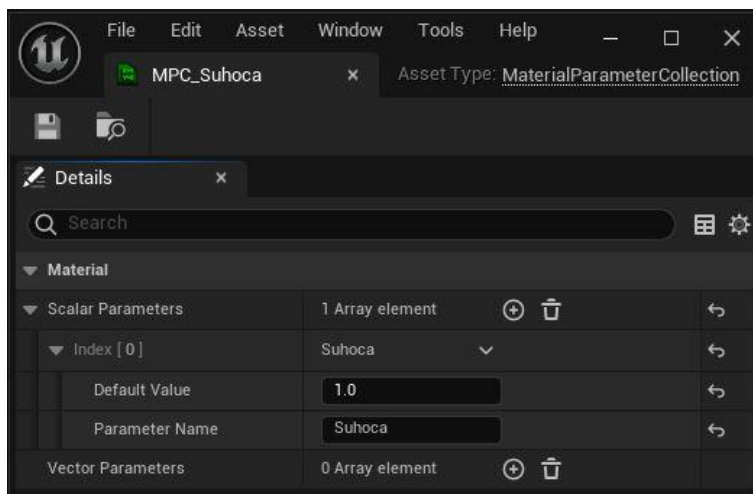


Slika 6.26 Postepeno pojavljivanje i nestajanje čestica

Dio koji određuje ponašanje čestica „Particle Update“ ima opciju „Scale Color“ gdje se podešava boja kliznim izbornikom. Na slici 6.26 prikazano je kako je postignuto da se čestice postepeno pojavljuju i da postepeno nestaju. Klizni gumbi bijele i crne boje kontroliraju alfa kanal (prozirnost) teksture, crna predstavlja prozirnost, bijela vidljivost.

## 6.2.5. Parametri materijala

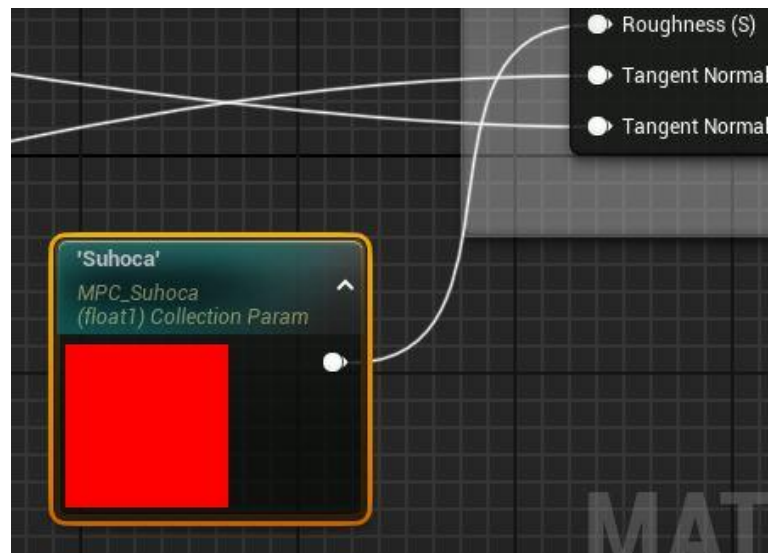
Svakom materijalu koji se nalazi izvan sobe dodjeljuje se parametar suhoće. Tome služi kolekcija parametara materijala (eng. *Material parameter collection*). Isto kao i kod gustoće oblaka, samo što je sada ovo primjenjivo na više materijala. Omogućeno je centralizirano upravljanje i mijenjanje parametara više materijala koji će imati isti indeks suhoće. Najprije se „suhoća“ kreira kao zasebno sredstvo.



Slika 6.27 Kolekcija parametara materijala suhoće

Dodaje se novi parametar i daje mu se naziv „Suhoca“. Davanje početne vrijednosti također je opcija. Sada je moguće taj parametar primijeniti različitim materijalima unutar Blueprint dijela materijala. Dodaje se kao zasebni čvor i rezultat se spaja na grubost materijala (eng. *roughness*). Čvoru je potrebno pridodati određeni parametar „Suhoca“.





Slika 6.28 Primjena parametra suhoće na materijal



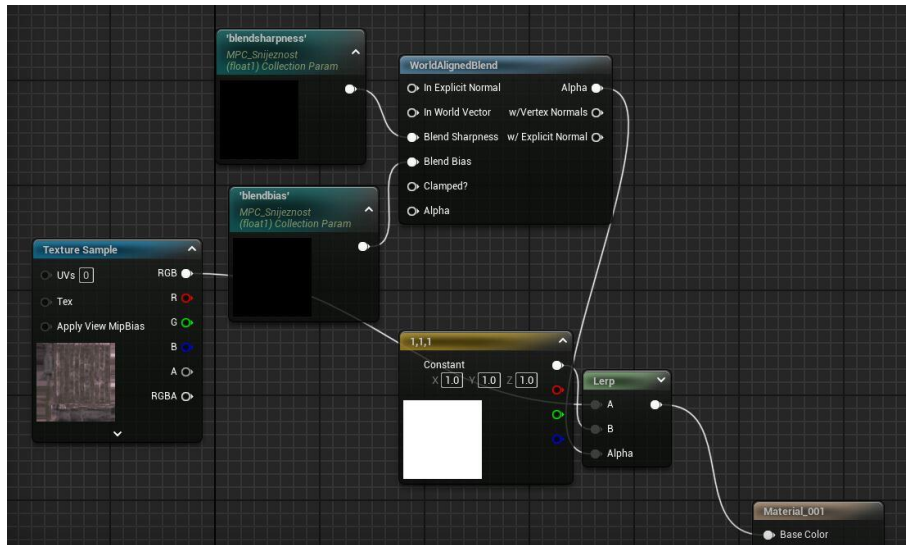
Slika 6.29 Suhoća vrijednosti 1.0



Slika 6.30 Suhoća vrijednosti 0.5

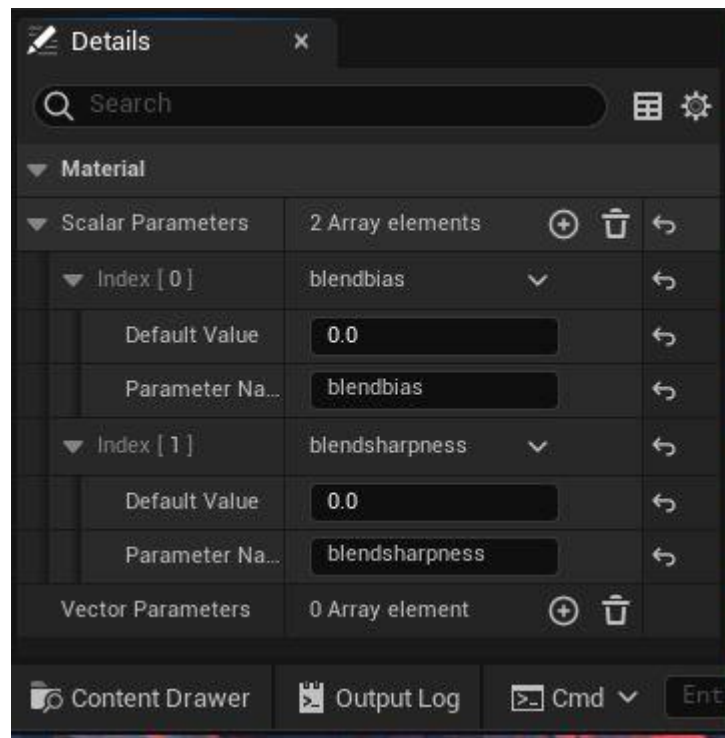


Na sličan način kreirana je i funkcija materijala za akumulaciju snijega.



Slika 6.31 Kolekcija čvorova koja simulira akumulaciju snijega na površini objekata

Snijeg se nakuplja na gornjoj površini bilo kojeg oblika. Kako bi se taj efekt postigao potrebno je pratiti vertikalnu os objekta. Čvor „World Aligned Blend“ gleda Z os u prostoru i parametrima „Blend Sharpness“ i „Blend Bias“ pomoću originalne teksture i boje snijega postavlja boju na gornji dio bilo kojeg modela.



*Slika 6.32 Kolekcija parametara materijala sniježnosti*

Kreirana je nova kolekcija parametara kako bi se kontrolirala količina akumulacija snijega na svakom materijalu kojeg snijeg dotakne.



*Slika 6.33 Oba parametra snježnosti na klupi su 0*



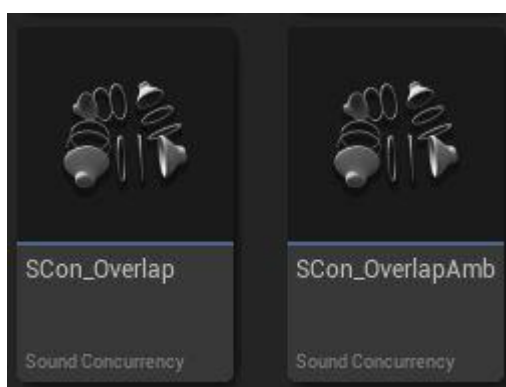
*Slika 6.34 Parametri snježnosti na klupi su podešeni*

### 6.3. Glazba i ambijentalni zvukovi

Tablica ispod prikazuje listu skladbi i ambijentalnih zvukova upotrebljenih u ovom projektu. Svi zvukovi unutar ovog projekta licencirani su putem stranice [artlist.io](http://artlist.io).

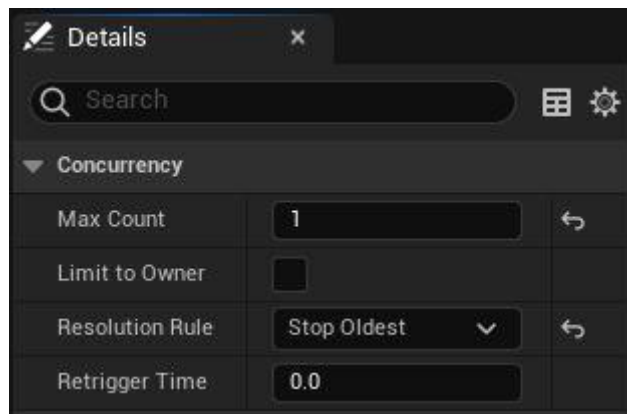
Autori:	Naslov djela:
Fable Forte	Courtyard Cakamity
Francesco Dandrea	The Doubt Pt. II
Ian Post	Mad Crows
Nobou	Strange Connection
Roie Shpinger	White Morning
Searching For Light	Under The Willow Tree
Stanley Gurvich	At Last

#### 6.3.1. Istovremenost zvuka



*Slika 6.35 Istovremenost za glazbu i zvukove ambijenta*

Istovremenost zvukova nazvan u Unreal Engine-u „Sound Concurrency“ je skup pravila koja propisuju način na koji se istovremeni zvukovi odvijaju. Stvorena su dva različita pravila koja se primjenjuju na glazbu i ambijent odvojeno.



*Slika 6.36 Pravila za zvukove*

Dva su pravila za dvije skupine zvukova, no ona će se primjenjivati na isti način. Istovremenost dopušta samo jedan zvuk, kada se novi zvuk pojavi prednost ima noviji, a ponovno pokretanje odmah kreće. Postavke su vidljive iz slike 6.36.

### 6.3.2. Prikaz vremenskih prilika i njihovih postavki

Vremenske varijante i njihove prethodno pojašnjene postavke u tablici prikazane su ispod. Odabrane je pet reprezentativnih vremenskih uvjeta. U tablici s vremenskim prilikama se nalazi više od 50 različitih varijacija.



Slika 6.37 Vedro nebo



Slika 6.38 Podaci u tablici za vedro nebo



Slika 6.39 Kiša



Slika 6.40 Podaci u tablici za kišu



Slika 6.41 Grmljavina s kišom



Slika 6.42 Podaci u tablici za kišu



Slika 6.43 Snijeg



Slika 6.44 Podaci u tablici za snijeg



Slika 6.45 Magla

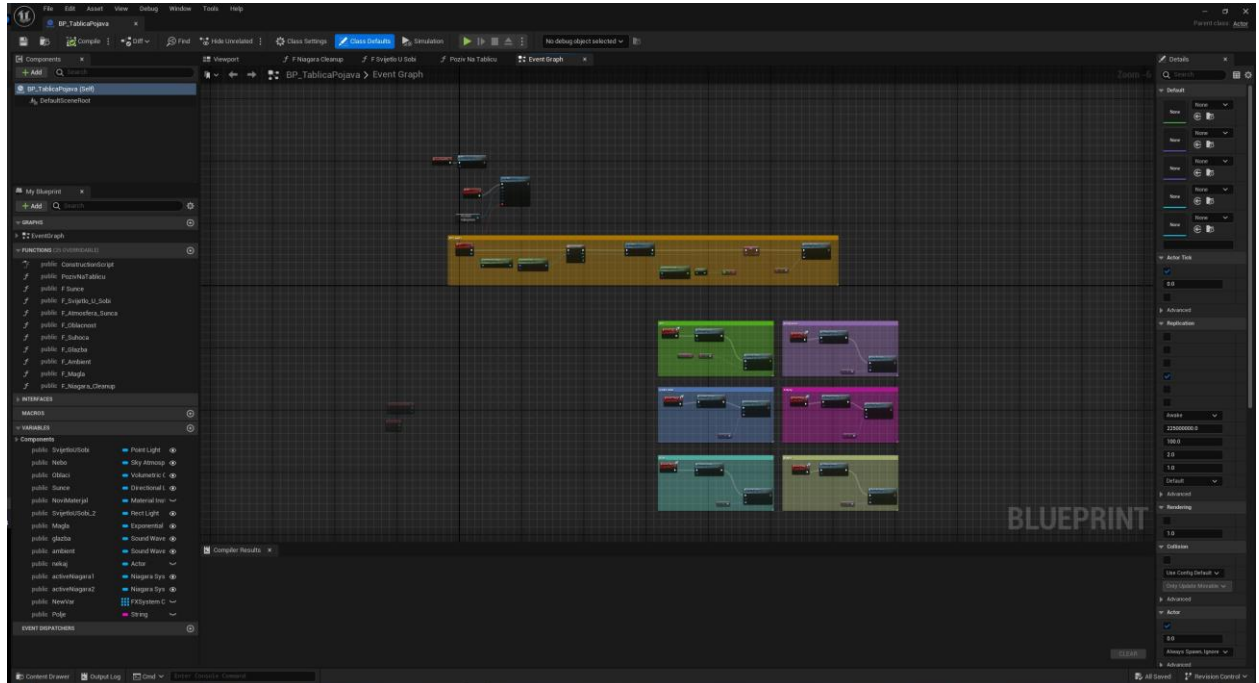


Slika 6.46 Podaci u tablici za maglu



## 7. Programiranje simulacije

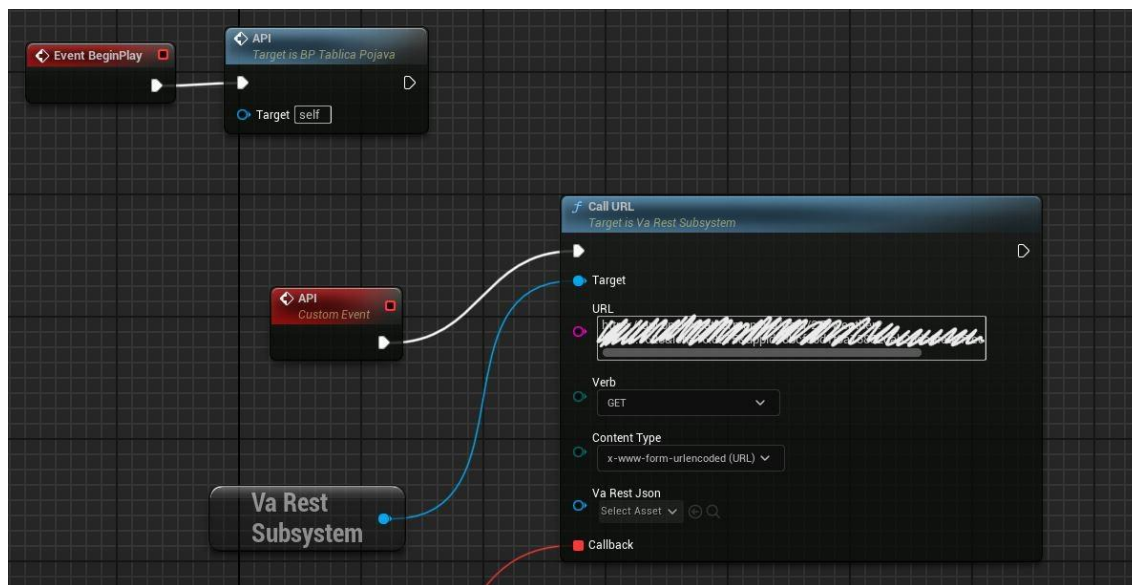
Ono što je centar ove simulacije i što čini cijeli ovaj program mogućim je Blueprint koji upravlja svim informacijama potrebnim za prikaz vremenskih pojava.



Slika 7.1 Prikaz Blueprint-a koji prikazuje vremenske pojave

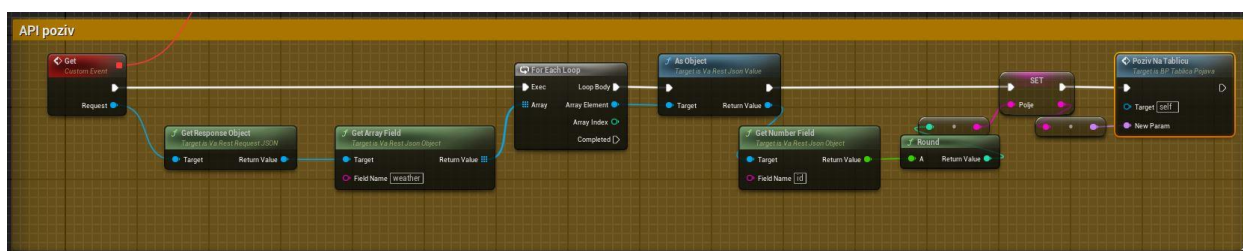


Program započinje zahtjevom pristupa podacima sa servisa.



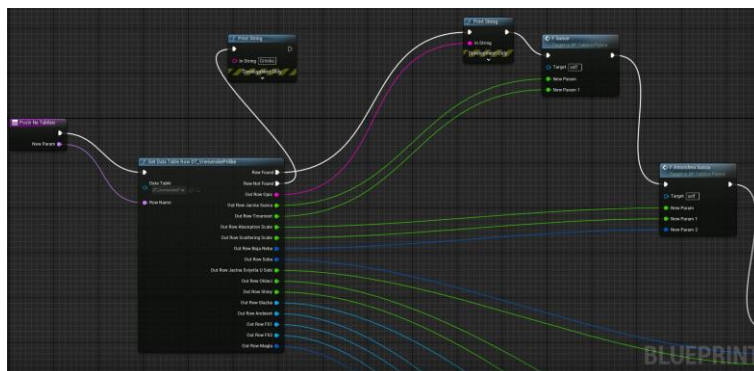
Slika 7.2 Poziv na server

Čvor „Događaj početak igre“ (*eng. Event BeginPlay*) započinje proces čim se program pokrene. Da ovaj program radi potrebno je omogućiti VaRest dodatak u Blueprint desnim klikom i dodavanjem čvora „VaRest Subsystem“. Sada je moguća komunikacija između klijenta i pružatelja usluga. Kreiran je Prilagođeni događaj (*eng. Custom event*) nazvan „API“ koji pokreće API poziv na Open Weather server pomoću URL-a do API završne točke sa potrebnim parametrima.



Slika 7.3 Dohvaćanje potrebnog polja

Novi Prilagođeni događaj naziva „Get“ dohvaća određeno polje iz dohvaćenih podataka servisa i sprema ga kao varijablu. Prvo dohvaća kompletan JSON objekt i pretvara ga u ugniježđeno polje (*eng. array*). Iteracijom prvo se traži polje „weather“ koje sadrži vrijednosti kao polje s više varijabli te je potrebno iterirati kroz njih kako bi se došlo do polja „id“. Zatim se ta vrijednost posprema u varijablu. Zbog kompatibilnosti s tipom podatka imena tablice broj se mora pretvoriti u cjelobrojnu vrijednost i konvertirati u string. Nakon toga se pokreće funkcija „Poziv na Tablicu“.



Slika 7.4 Funkcija koja dohvaća podatke iz ćelija tablice

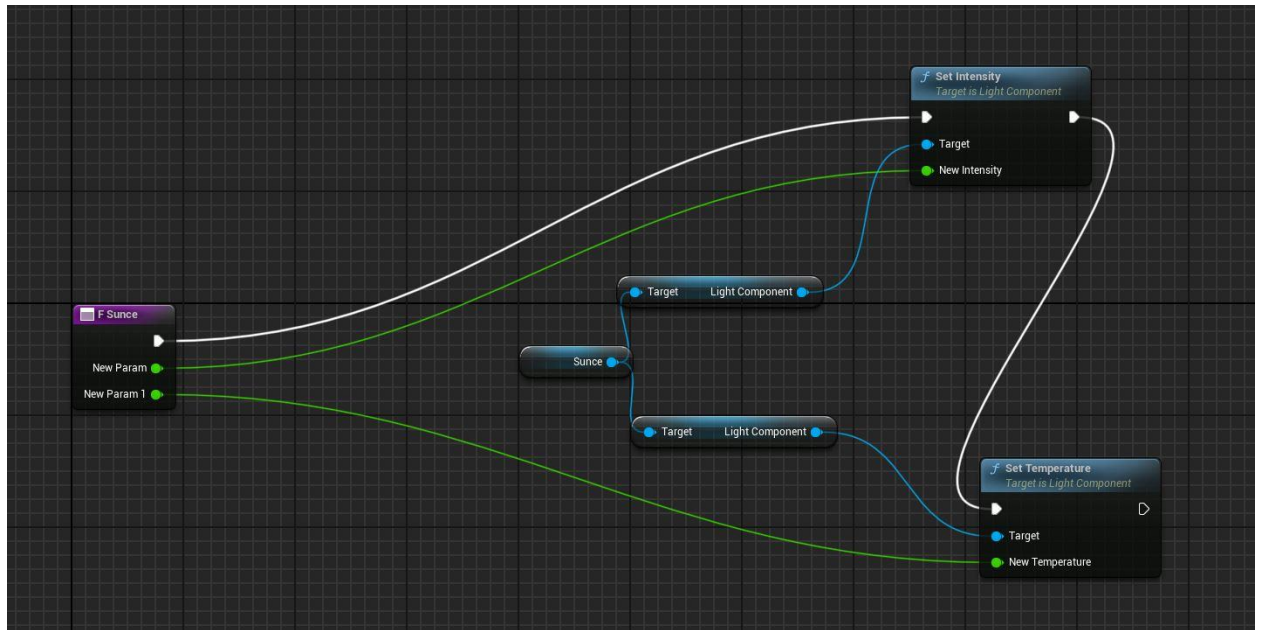
Ulazna vrijednost funkcije je podatak tipa „name“ tako da se može dohvatiti odgovarajući red u tablici. Čvor „Dohvati ime reda u tablici“ (eng. *Get table row name*) povezan je na tablicu „DT\_VremenskePrilike“ koja pruža potrebne podatke. Izlazne vrijednosti ovog čvora prikazuju svaku ćeliju koja sadrži varijable kako bi u cijelosti kreirala određenu vremensku pojavu tog reda u tablici. Na uspješno dohvaćen red u konzoli se ispisuje ime vremenske prilike i nastavlja se s daljnjim procesom, u suprotnome se u konzoli ispisuje poruka „Greska“, što se događa ukoliko je napravljeno previše poziva u kratkom roku ili korisnik nije spojen na internet.

Reakcije na tipove podataka su dalje razvrstane na ostale funkcije. Najprije se unutar izrađenog Blueprint-a postavljaju varijable čiji tipovi odgovaraju akterima na sceni koje želimo kontrolirati i koji su stalno prisutni.

Svijetlo USobi	SvijetloUSobi	▼	🔍	🖋️	↶
Nebo	SkyAtmosphere	▼	🔍	🖋️	↶
Oblaci	VolumetricCloud	▼	🔍	🖋️	↶
Sunce	DirectionalLight	▼	🔍	🖋️	↶
Svijetlo USobi 2	SvijetloUSobi2	▼	🔍	🖋️	↶

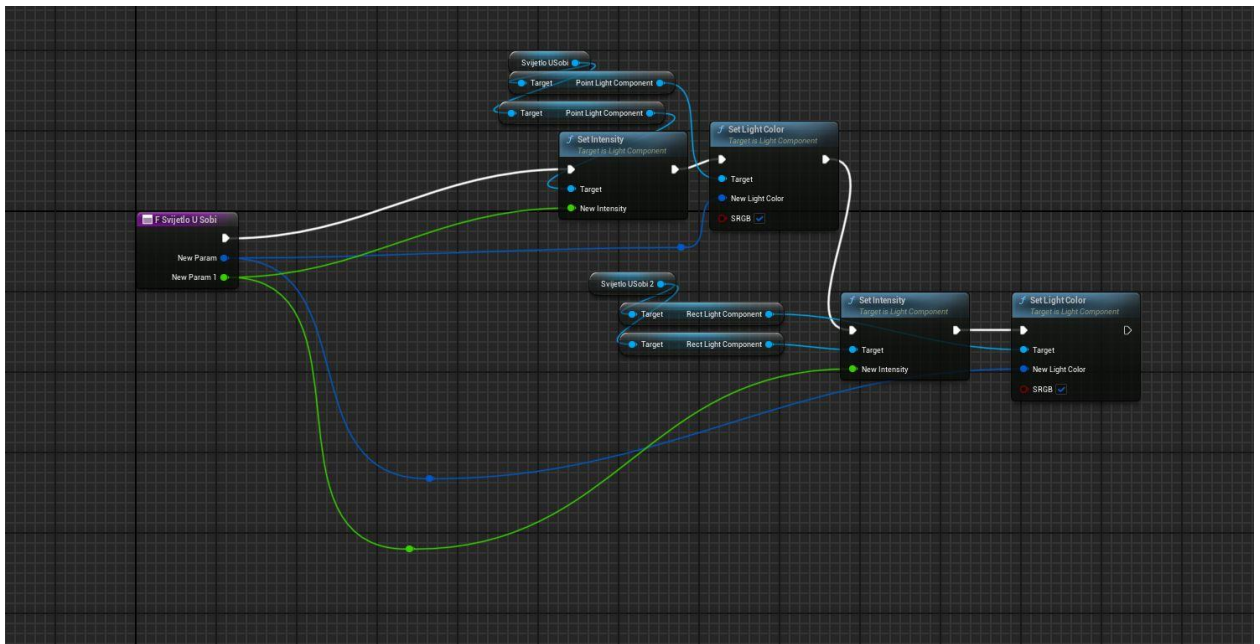
Slika 7.5 Varijable povezane s akterima na sceni

Kada su varijable povezane s odgovarajućim akterima, moguće je mijenjati njihove vrijednosti na sceni.



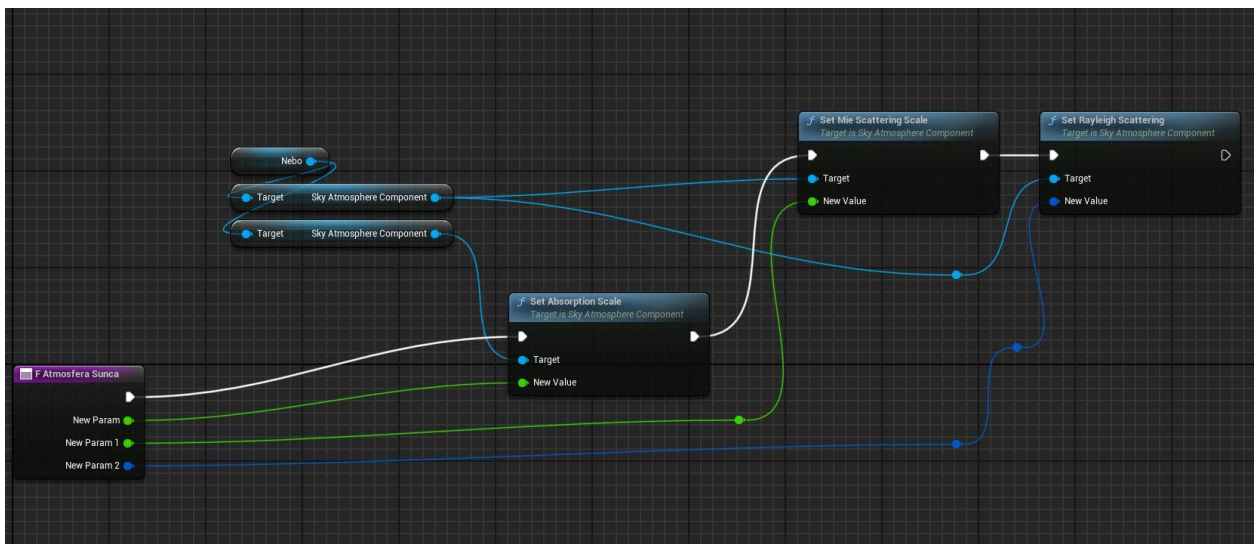
Slika 7.6 Funkcija mijenja toplinu i intenzitet sunca

Funkcija „F\_Sunce“ dohvaća ćelije iz tablice float tipa kako bi postavila intenzitet i toplinu sunčevog svijetla pomoću čvorova Set Intensity i Set Temperature koji su spojeni ciljnim pinom (eng. *Target pin*) na varijablu „Sunce“.

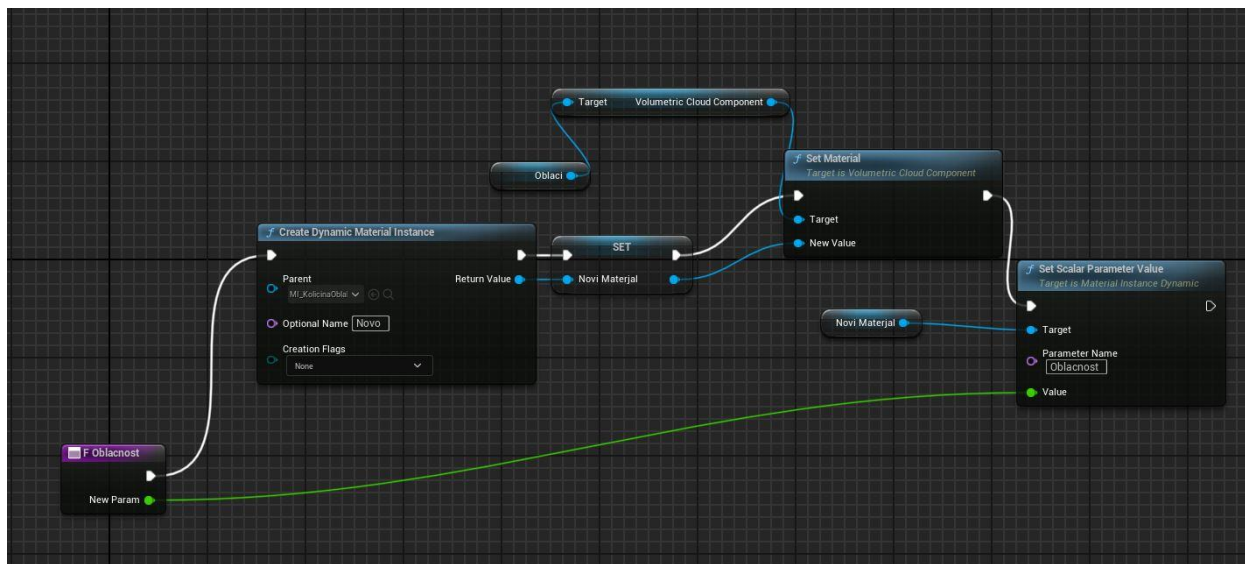


Slika 7.7 Funkcija mijenja boju i jačinu svjetlosnih komponenti

Funkcija za mijenjanje boje i intenziteta svjetla unutar sobe dohvaća podatke tipa float i linearna boja (*eng. Linear color*). Linearna boja sadrži četiri glavne komponente: količinu crvene, plave, zelene i prozirnost, na temelju toga se stvara boja koju svjetlo emitira. Čvorovi za intenzitet i boju spojeni su na dva svjetla koja su postavljena unutar sobe.

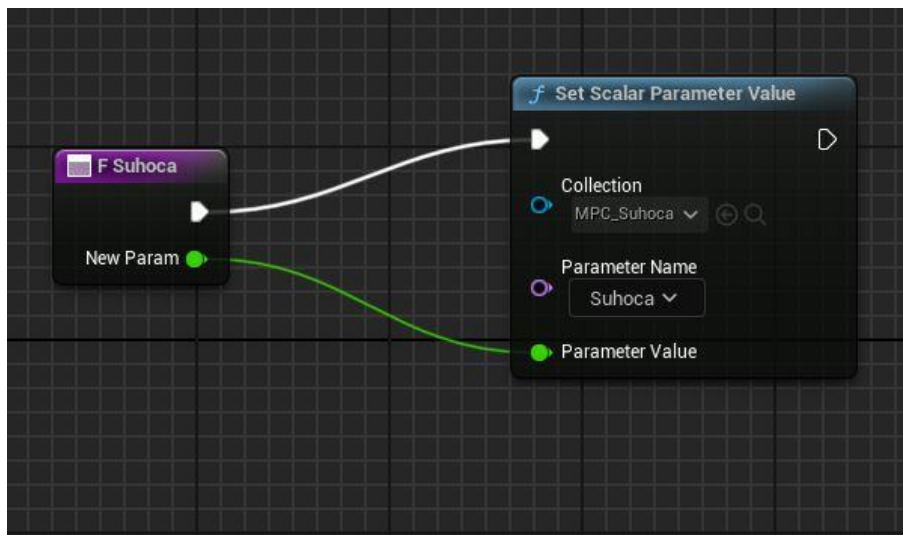


Slika 7.8 Funkcija koja mijenja način raspršenja sučeve svjetlosti



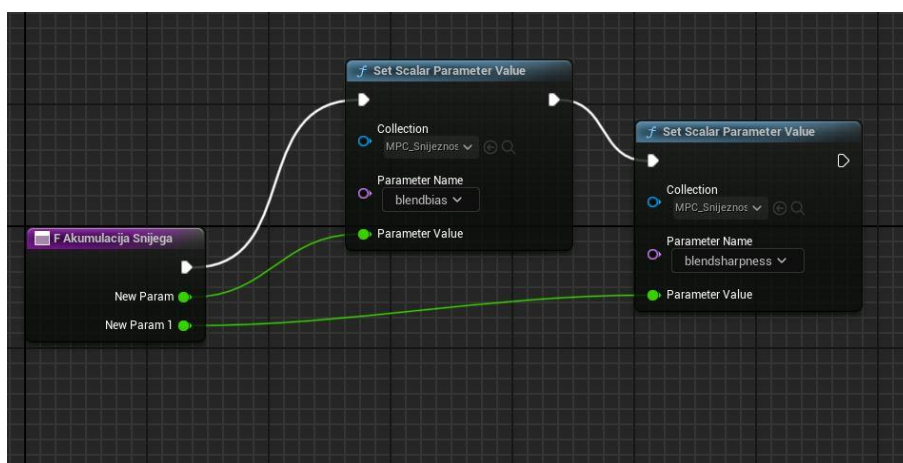
Slika 7.9 Funkcija koja kontrolira gustoću oblaka

Funkcija „F\_Oblacnost“ prvo kreira dinamičnu instancu materijala (*eng. Dynamic Material Instance*) čiji je roditelj (*eng. parent*) instanca „MI\_KolicinaOblaka\_inst“, stvara novi materijal kao varijablu i postavlja mu novu vrijednost čvorovima Set Material i Set Scalar Parameter Value. Vrijednost određuje broj iz ćelije koja određuje gustoću oblaka, koji odgovara prethodno izrađenom parametru u materijalu „M\_KolicinaOblaka“.



*Slika 7.10 Funkcija koja određuje suhoću materijala*

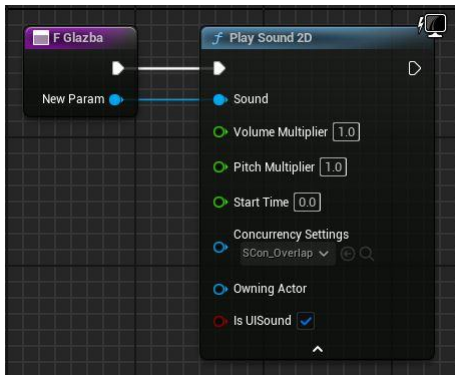
Funkcija za suhoću materijala koristi čvor za postavljanje vrijednosti parametra materijala „MPC\_Suhoca“. Ovo je jednostavnija metoda mijenjanja parametara unutar materijala.



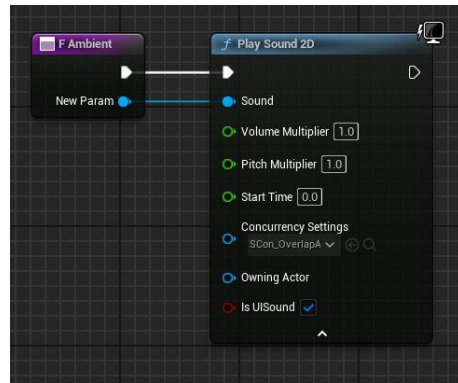
*Slika 7.11 Funkcija koja određuje akumulaciju snijega*

Funkcija za količinu nakupljanja snijega djeluje na identičan način.



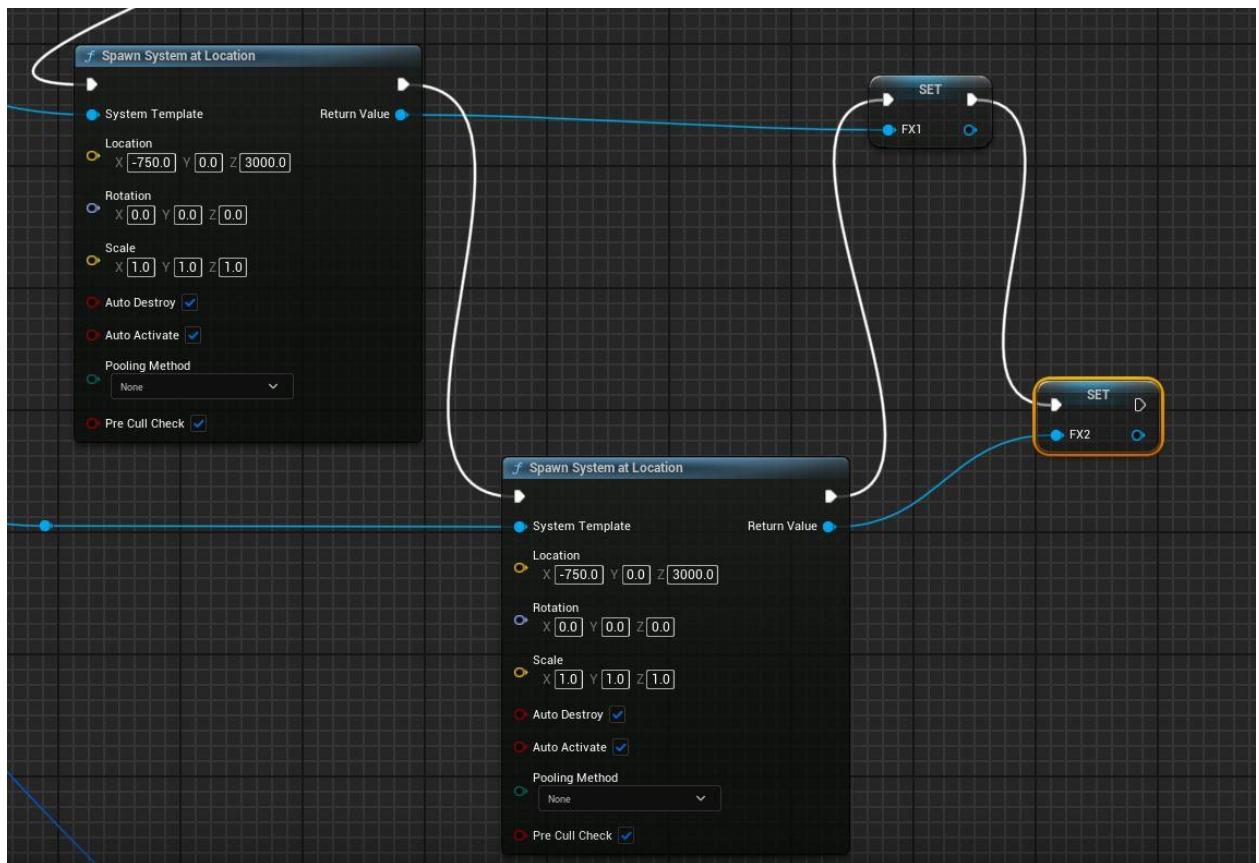


Slika 7.12 Funkcija koja kontrolira glazbu



Slika 7.13 Funkcija koja kontrolira ambient

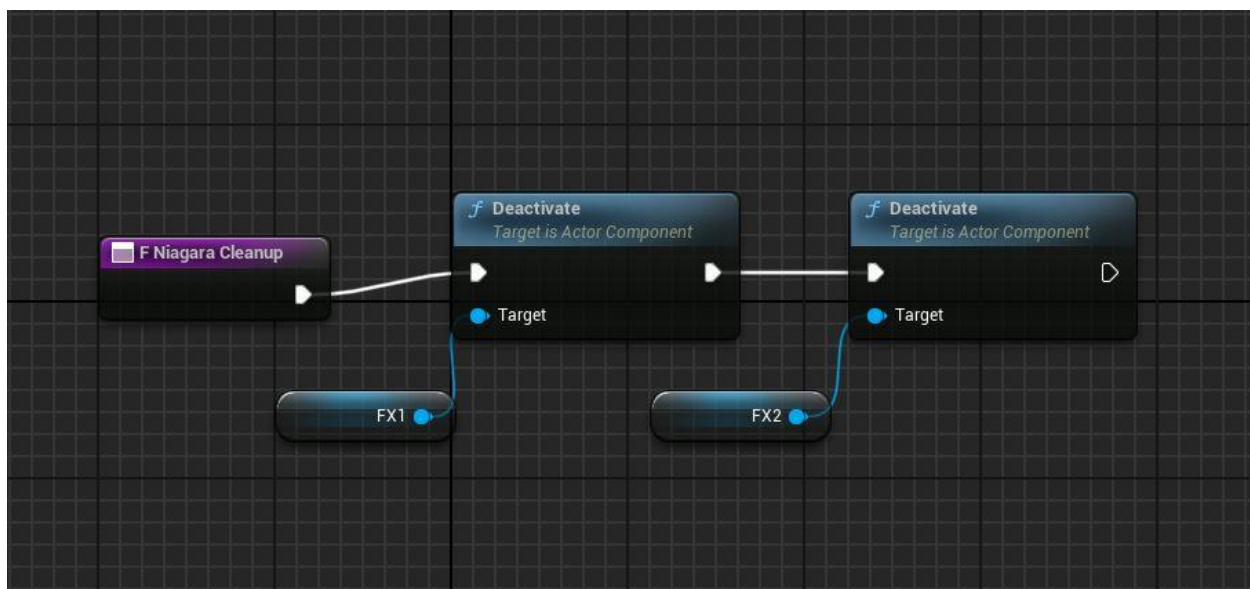
Funkcije za kontroliranje ambijentalnog zvuka i glazbe djeluju isto. Dohvaćaju podatak o kojoj wav datoteci se radi i pokreću reprodukciju koristeći pravila o istovremenosti zvuka koja su prethodno bila izrađena. Svaki zvuk koji prođe kroz svoju funkciju i primjenjuje to pravilo te s obzirom da su pravila razdvojena ne može doći do situacije da se ambijentalni zvuk i glazba međusobno poništavaju.



*Slika 7.14 Čvorovi koji dohvaćaju Niagara sustave na scenu*

Svaki red u tablici, tj. svaka vremenska prilika posjeduje dva Niagara sustava. Koriste se dva sustava jer ima situacija kad u isto vrijeme na sceni trebaju dva različita sustava čestica, na primjer kad pada kiša i grmljavina ili kad pada snijeg pomiješan sa kišom. U slučaju kad drugi sustav nije potreban, njegova funkcija emitiranja čestica se ugasi. Čvorovi koji pozivaju Niagara sustave na scenu spremaju pozvane efekte u varijable.



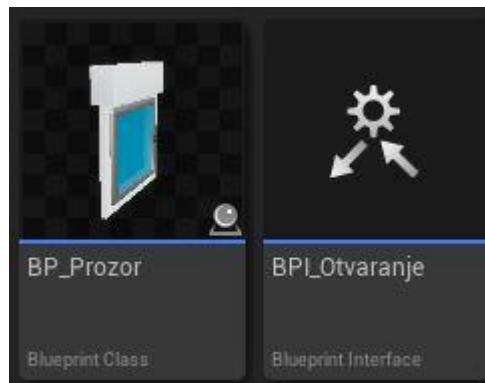


*Slika 7.15 Funkcija koja miče nepotrebne efekte*

Te varijable su potrebne kako bi varijabla „F\_Niagara\_Cleanup“ funkcionirala. Funkcija se poziva svaki puta kada se pozove novi red iz tablice kako se sustavi ne bi prikazivali jedni preko drugih.

## 8. Otvaranje i zatvaranje prozora

Soba u ovom projektu posjeduje prozor koji se može otvarati i zatvarati. Kombinacijom Blueprinta i Blueprint Sučelja (*eng. Blueprint Interface*) moguće je dati kontrolu igraču da kontrolira otvaranje i zatvaranje prozora.



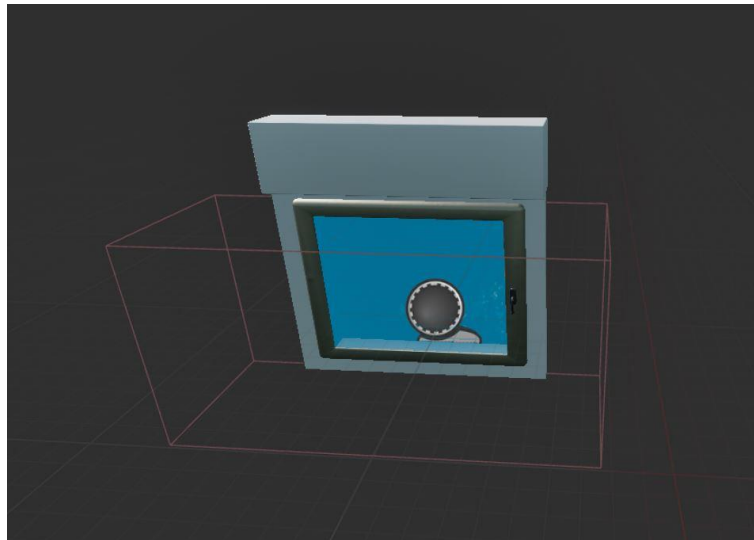
Slika 8.1 Blueprint i sučelje koji otvaraju i zatvaraju prozor pritiskom tipke igrača

Kreira se novi Blueprint koji u sebi sadrži sastavne dijelove prozora, kocku koja će služiti kao pivot i kocka koja će detektirati nalazi li se igrač unutar polja interakcije ili ne.



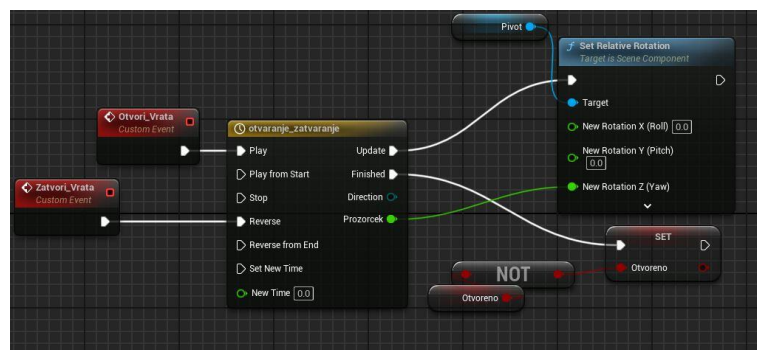
Slika 8.2 Sastavni dijelovi Blueprint-a

Dijelovi koji se moraju micati djeca (*eng. children*) su statičke mreže kocke tako da se okretanjem kocke okreću svi dijelovi prozora koje je potrebno micati.



Slika 8.3 Prikaz svih komponenti u pregledniku Blueprint-a

Kako bi animacija otvaranja i zatvaranja funkcionirala, potrebno je postaviti događaj koji to aktivira.

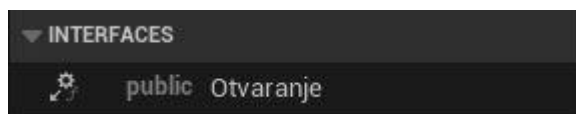


Slika 8.4 Blueprint koji otvara i zatvara prozor

Događaj se sastoji od dva prilagođena događaja „Otvori\_Vrata“ i „Zatvori\_Vrata“, animacije koja okreće pivot za 90 stupnjeva i čvora koji postavlja kut varijable Pivot koja je vezana na kocku koja otvara prozor. Događaj otvaranja pokreće animaciju i postavlja vrijednost boolean podatka zvanog „Otvoreno“ što određuje da je prozor otvoren. Događaj zatvaranja pokreće animaciju unazad i postavlja kut pivota na početnu vrijednost.

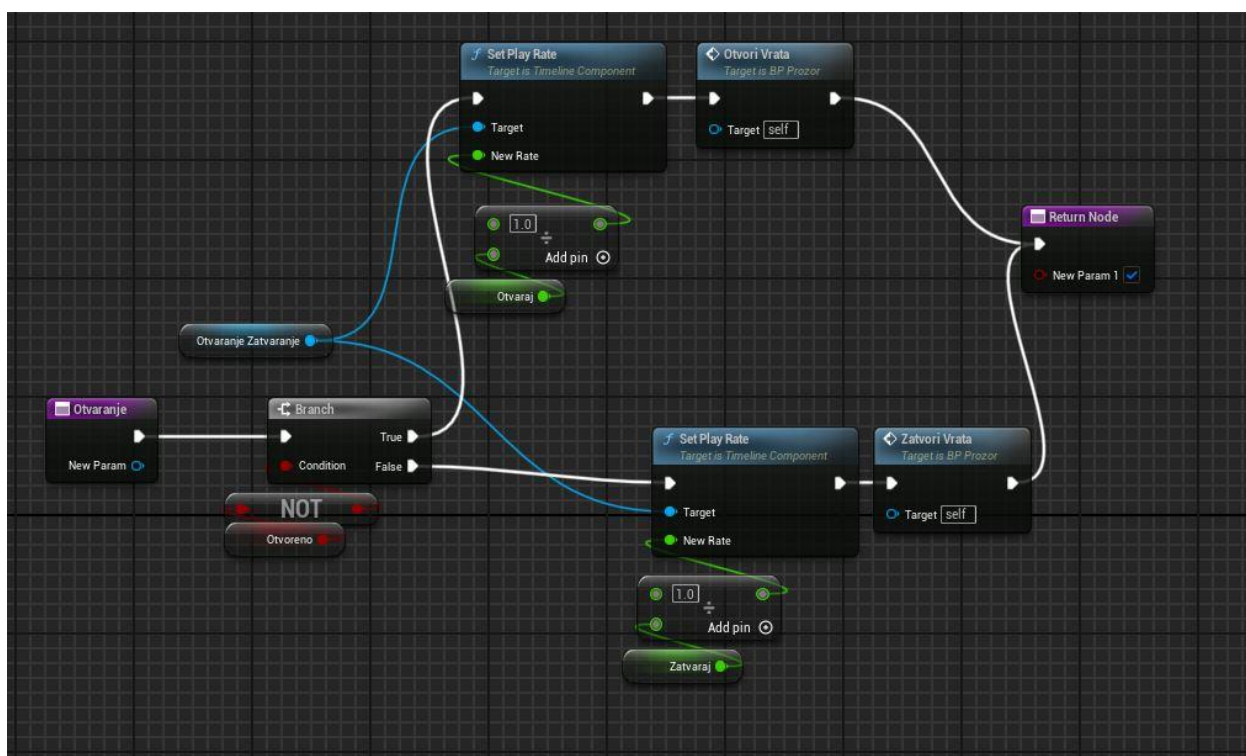
Da bi omogućili kontrolu nad prozorom, ostatak logike se obavlja u Blueprint-u igrača. Da bi se interakcija mogla uspostaviti, potrebno je izraditi sučelje prikazano na slici 8.1, „BPI\_Otvaranje“. Ova sučelja sadrže definicije funkcija bez stvarne implementacije pri čemu omogućuju korištenje različite akcije u različitim Blueprint-ima, a time se i rasterećuje program što pridonosi optimizaciji i boljim performansama.

Sučelje se poziva unutar Blueprint-a gdje se željena akcija odvija i dodjeljuje mu se funkcija. Ovdje je sučelje implementirano Blueprint-u za prozor i dodata mu je funkcija „Otvaranje“.



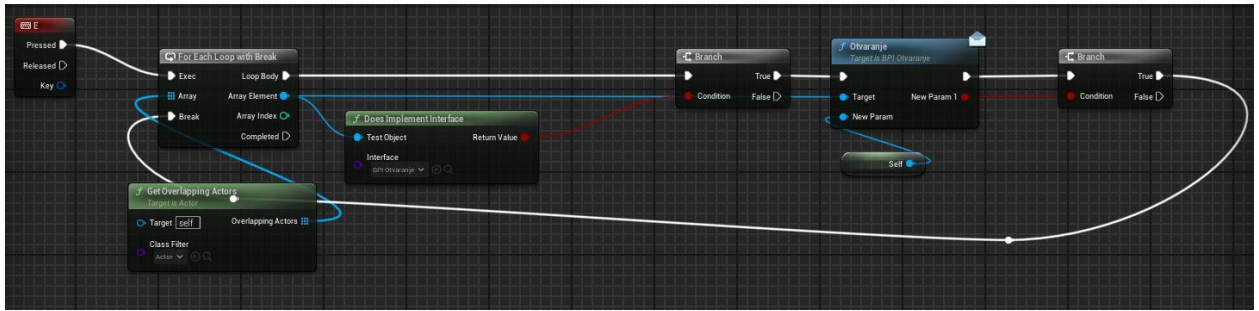
Slika 8.5 Sučelje Otvaranje povezano je s Blueprint-om za prozor

Unutar tog Blueprint-a definira se što funkcija otvaranja čini.



Slika 8.6 Funkcija otvaranja i zatvaranja prozora

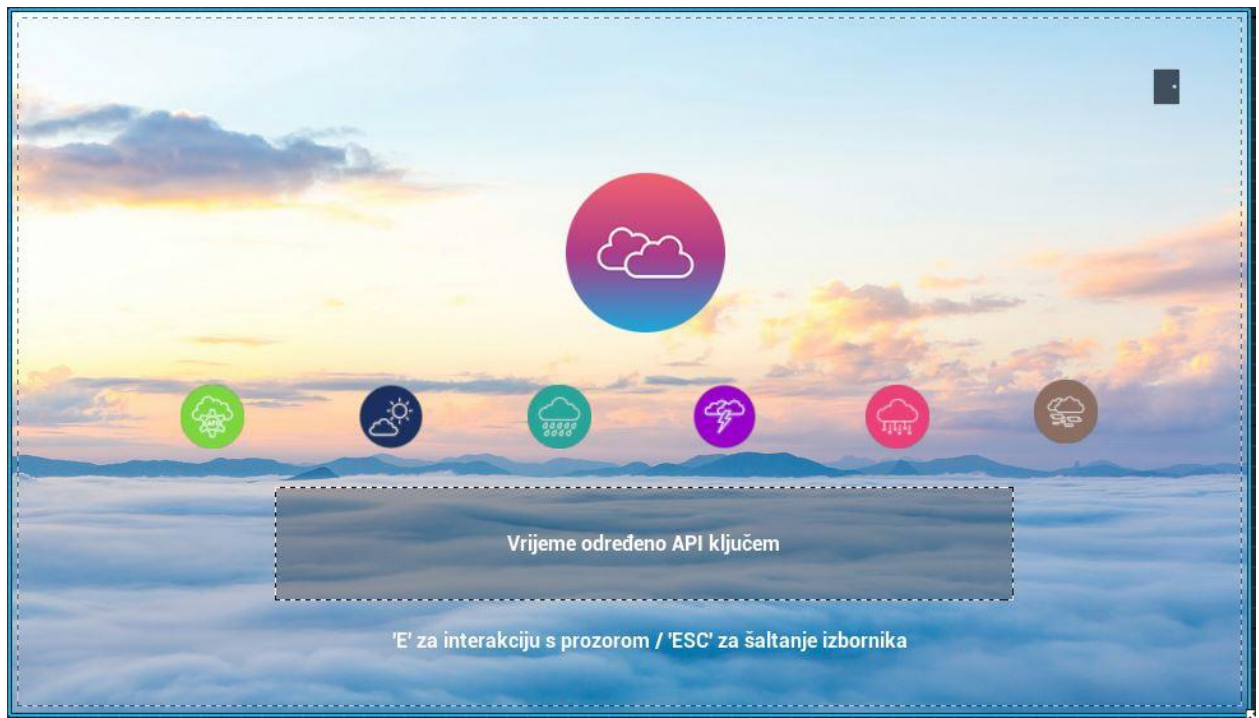
Funkcija koristi čvor grananja (*eng. branch*) da bi pokrila slučaj kada se prozor otvara i zatvara. Boolean prethodno definiran je uvjet za grananje. Ako je vrijednost istinita animacija se pokreće i poziva se događaj „Otvori\_Vrata“. U suprotnome se pokreće animacija unazad prijeko događaja „Zatvori\_Vrata“. Varijable „Otvoraj“ i „Zatvaraj“ javne su varjiable tipa float koje je moguće mijenjati kako bi se mogla postaviti brzina otvaranja i zatvaranja prozora. Dije se brojem 1 zato što sama animacija traje jednu sekundu pa će zadane vrijednosti imati trajanje u sekundama.



Slika 8.7 Kontrola nad otvaranjem i zatvaranjem prozora

Unutar igračevog Blueprint-a pritisak na tipku „e“ pokreće petlju koja provjerava je li prozor otvoren ili zatvoren, pokreće funkciju sučelja i vraća boolean vrijednost na početak petlje da događaj može pratiti je li prozor otvoren ili zatvoren.

## 9. Interaktivno sučelje



*Slika 9.1 Izgled interaktivnog sučelja*

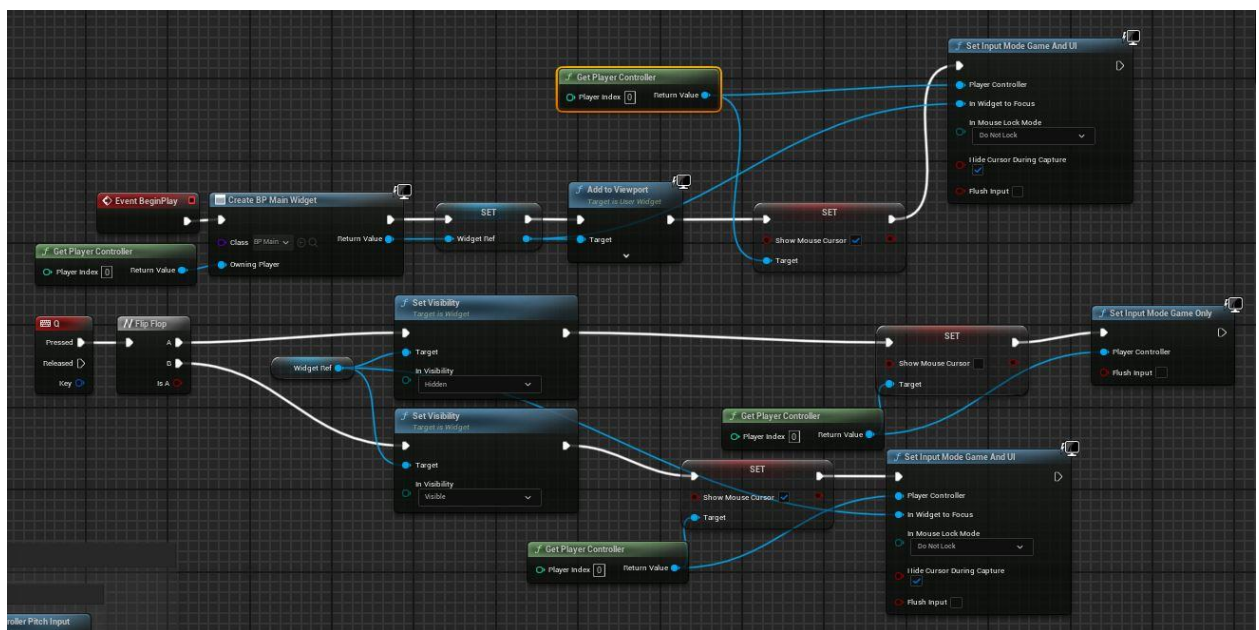
Sučelje u ovoj simulaciji služi da informira korisnika o kontrolama simulacije i o tome koji je trenutni način rada simulacije. Sastoji se od elemenata koji imaju čisto vizualnu funkciju, gumbova koji mijenjaju način rada između API poziva i prikaza ostalih vremenskih pojava, tekstualnih elemenata koji su sinkronizirani s akcijama gumba i gumba koji izlazi iz cijelog programa.





Slika 9.2 Sastavni dijelovi sučelja

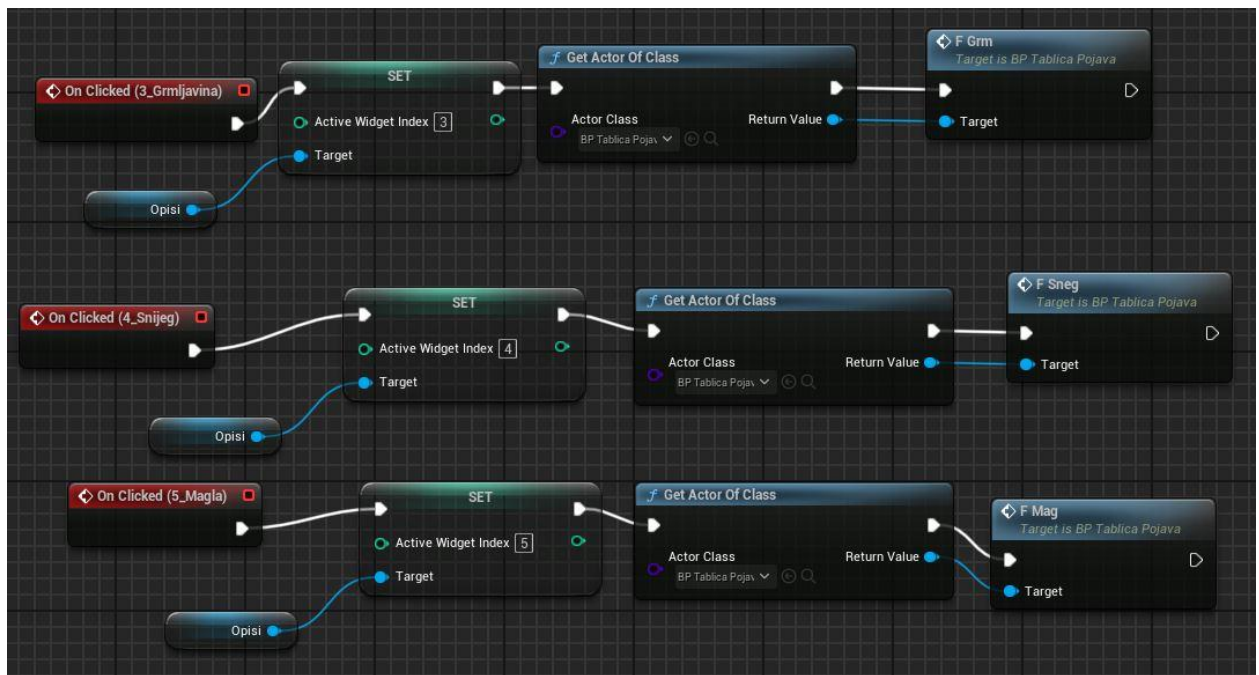
Svako interaktivno sučelje je zapravo Blueprint. Tamo je moguće kreirati sve funkcije koje gumbi izvršavaju. Postavljanje elemenata u dizajneru poprilično je jednostavno i intuitivno. Dohvaćaju se potrebni elementi i postavljaju se gdje su potrebni. Pozadine i slike dodjeljuju se u obliku tekstura.



Slika 9.3 Kontroliranje sučelja

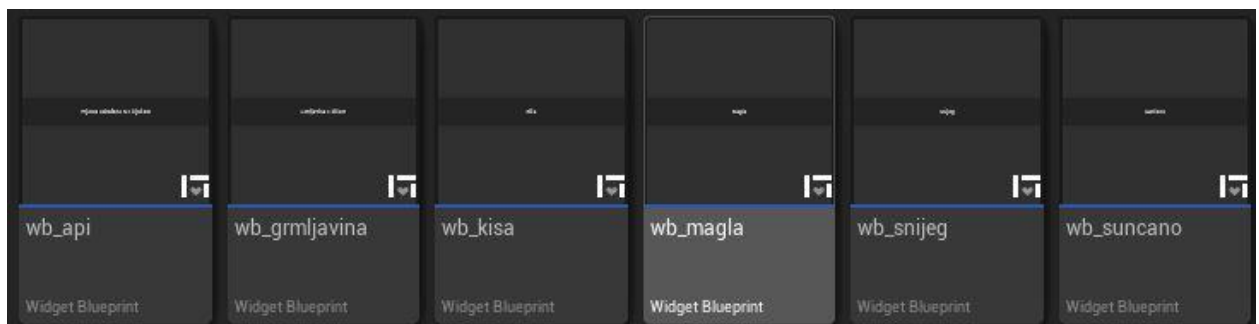
Kada se program pokrene sučelje se pojavljuje na ekranu i igračeva kontrola prebacuje se na sučelje. Pritiskom tipke „Escape“ prikaz se izmjenjuje između vidljivosti sučelja. Kada je sučelje

vidljivo igrač ima kontrolu nad sučeljem, kada je sučelje sakriveno igrač ima kontrolu nad 3D prostorom. Time je postignuto da korisnik odmah može odabrati vremenske prilike na sučelju bez da u međuvremenu pomiče kameru, a kada sučelja nema, micanje kamere potezima miša je ponovno omogućeno. Bitno je također napomenuti kada se radi o ulaznim akcijama korisnika kao što su klikovi i pritisci tipki, potrebno je koristiti čvor Get player controller, jer inače događaji pokrenuti takvim inputima nisu mogući, kontrolu igrača potrebno je referencirati.



Slika 9.4 Gumbi iz sučelja koji kontroliraju događaje

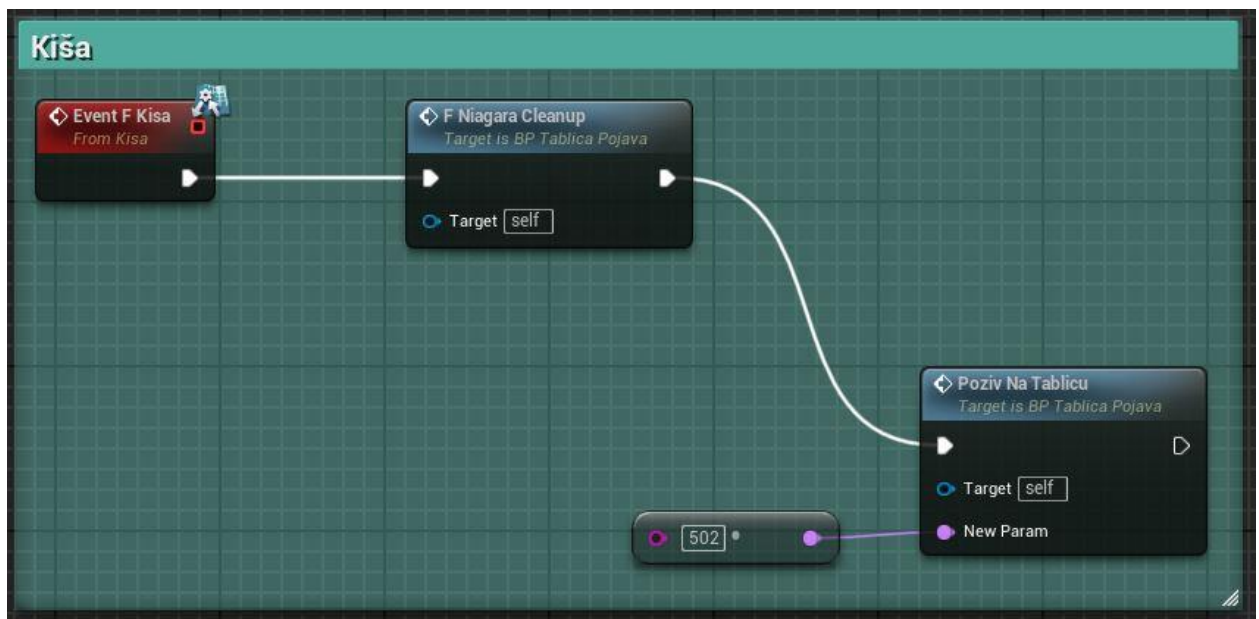
U ovom popisu elemenata stavka „opisi“ je zapravo preklopnik koji izmjenjuje sadržaj ovisno o tome koji je gumb kliknut. „Sadržaj“ unutar preklopnika su ostala sučelja kreirana zasebno kao tekst koji opisuje što određeni gumb radi. Redom kojim su ti sadržaji postavljeni dodijeljen je indeks. Indeks je cjelobrojna vrijednost koja započinje od 0 i povećava se za jedan svaki puta kada se doda jedan sadržaj.



Slika 9.5 Opisi akcija gumbova



U Blueprint dijelu potrebno je dodijeliti gumbima „on-click“ događaj. Svaki puta kada se određeni gumb klikne pogled se prebaci na pogled s odgovarajućim indeksom i izvršava ostale funkcije koje su određene vlastitim Blueprint Sučeljem. Sučelje je povezano s Blueprintom koji kontrolira dohvaćanje redova tablice. Da bi se postigla interakcija između ta dva Blueprint-a, integrirano je Blueprint Sučelje „BPI\_Api“ s funkcijama „F\_Grom“, „F\_Snijeg“ i ostale koje svaka posebno dohvaćaju svoj red tablice kako bi se prikazalo određeno vrijeme klikom miša. Zatim čvor može pozvati „BP\_TablicaPojava“ i aktivirati funkcije sučelja. Time je moguće vidjeti što program može prikazivati, da se ne prikazuje samo vrijeme dohvaćeno API pozivom.



Slika 9.6 Primjer interakcije interaktivnog sučelja s API Blueprintom

Funkcija „F\_Kisa“ pozvana je kao vlastiti događaj i izvršava najprije funkciju koja odstranjuje prethodno postavljene Niagara sustave i zatim dohvaća ostale elemente iz tablice. Vrijednost imena tablice je fiksna tako da gumb za kišu svaki puta prikazuje određenu varijaciju kiše. Isti proces je ponovljen za vedro nebo, snijeg, grmljavinu i maglu. Također postoji gumb koji vraća korisnika na prikaz vremenskih prilika učitani preko API poziva. On koristi varijablu koja je postavljena kao vrijednost na početku pokretanja programa kako bi se pozivi na server ograničili na jedan poziv po pokretanju simulacije.

## 10. Zaključak

Unreal Engine izuzetno je svestran alat koji, u kombinaciji s različitim servisima, alatima i dodacima postaje moćno sredstvo za razvoj i implementaciju projekata koji idu van okvira klasičnih igara. U proteklih nekoliko godina, Unreal Engine postao je sve pristupačniji čime se njegova upotreba proširila. Unreal Engine se osim svojom tehničkom moći također ističe i bogatom kolekcijom dodataka i resursi koji su dostupni korisnicima besplatno, kao dodatak VaRest. Brojni besplatni dodaci omogućuju proširivanje funkcionalnosti i prilagodbu različitim projektima.

Unreal Engine zajednica, posebno na njegovim službenim forumima, izrazito je aktivna i uvijek spremna pomoći ili barem raspraviti problem ili ideju koju netko predloži. To su sve faktori koji doprinose brzom razvoju znanja unutar samog programa i predstavlja veliku priliku za učenje, što Unreal Engine čini neprocjenjivim alatom za svakog pojedinačnog korisnika. Tome također u velikom djelu pridonosi i vizualni sustavi za skriptiranje Blueprint.

Korištenjem API poziva u kombinaciji s VaRest dodatkom, Unreal komunicira s vanjskim servisima u stvarnom vremenu. Implementacijom API-a otvara se širok spektar mogućnosti za dinamičke aplikacije, igre i simulacije. To može pridonijeti boljem i jedinstvenom doživljaju korisnika. Bitno je naglasiti da API ne dohvaća samo podatke o vremenu kao u ovom projektu, već je moguće koristiti raznovrsne podatke dostupne putem raznih API sučelja koja se mogu iskoristiti u različite svrhe. Dobar primjer su besplatni NASIN TLE API koji pruža informaciju o trenutnom položaju i orbiti objekata koji kruže oko Zemlje i SpaceX API koji nudi podatke o različitim tipovima raketa, lansiranjima i detaljima o stelitima sa Starlink mreže. Ovakva vrsta integracije podataka stvara novi potencijal za projekte stvorene unutar Unreal Engine-a koji mogu biti zabavnog i/ili edukativnog karaktera.

## 11. Literatura

- [1] VaRest dokumentacija dostupna na:  
<https://ufna.notion.site/VaRest-UE4-Plugin-40b98c54fc184033b60a42e0e4753536>, rujan 2024.
- [2] Open Weather servis dostupan na:  
<https://openweathermap.org/>, rujan 2024.
- [3] Updating Projects to Newer Versions of Unreal Engine dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/updating-projects-to-newer-versions-of-unreal-engine?application\\_version=5.3](https://dev.epicgames.com/documentation/en-us/unreal-engine/updating-projects-to-newer-versions-of-unreal-engine?application_version=5.3), rujan 2024.
- [4] Quixel Bridge Plugin for Unreal Engine dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/quixel-bridge-plugin-for-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/quixel-bridge-plugin-for-unreal-engine?application_version=5.4), rujan 2024.
- [5] JSON viewer online servis dostupan na:  
<https://jsonformatter.org/json-viewer>, rujan 2024.
- [6] Dana Table dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/BlueprintAPI/EditorScripting/DataTable?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/BlueprintAPI/EditorScripting/DataTable?application_version=5.4), rujan 2024.
- [7] Material Instance dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/BlueprintAPI/Interchange/Node/MaterialInstance?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/BlueprintAPI/Interchange/Node/MaterialInstance?application_version=5.4), rujan 2024.
- [8] Material Parameter Collections dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/using-material-parameter-collections-in-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/using-material-parameter-collections-in-unreal-engine?application_version=5.4), rujan 2024.
- [9] Niagara System and Emitter Module Reference dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/system-and-emitter-module-reference-for-niagara-effects-in-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/system-and-emitter-module-reference-for-niagara-effects-in-unreal-engine?application_version=5.4), rujan 2024.
- [10] Sound Concurrency Reference Guide dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/sound-concurrency-reference-guide?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/sound-concurrency-reference-guide?application_version=5.4), 2024.
- [11] Blueprint Interface dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/blueprint-interface-in-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/blueprint-interface-in-unreal-engine?application_version=5.4), rujan 2024.
- [12] Implement Interface in Blueprint dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/implementing-blueprint-interfaces-in-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/implementing-blueprint-interfaces-in-unreal-engine?application_version=5.4), rujan 2024.
- [13] Editing Timelines dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/editing-timelines-in-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/editing-timelines-in-unreal-engine?application_version=5.4), rujan 2024.
- [14] Widget Blueprints dostupno na:  
[https://dev.epicgames.com/documentation/en-us/unreal-engine/widget-blueprints-in-umg-for-unreal-engine?application\\_version=5.4](https://dev.epicgames.com/documentation/en-us/unreal-engine/widget-blueprints-in-umg-for-unreal-engine?application_version=5.4), rujan 2024.

## Popis slika

Slika 1.1 Logotip Unreal Engine-a .....	10
Slika 2.1 Kreiranje projekta.....	14
Slika 2.2 Načini rada u Unreal Engine-u .....	15
Slika 2.3 Polygon alat.....	15
Slika 2.4 Novi oblik nastao podloču Polygon alata.....	16
Slika 2.5 Izrađen model sobe.....	16
Slika 2.6 Kreiranje terena Landscape mode-om.....	17
Slika 2.7 Quixel Bridge kolekcija resursi .....	18
Slika 2.8 Preuzimanje modela s Quixel Bridge-a u projekt .....	18
Slika 2.9 Uvođenje modela u projekt .....	19
Slika 2.10 Model kreveta unutar scene i preglednika sadržaja.....	20
Slika 2.11 Instance modela na ulici .....	21
Slika 2.12 Instance postavljeni na sceni .....	22
Slika 3.1 Mijenjanje verzije programa .....	23
Slika 3.2 Odabir verzije programa.....	23
Slika 4.1 Sučelje za kreiranje ključa.....	24
Slika 4.2: Rezultat pretraživanja Koprivnice.....	24
Slika 4.3 URL za poziv API-u.....	24
Slika 4.4 JSON odgovor na stranici JSON viewer .....	25
Slika 4.5 Dio liste vremenskih prilika Open Weather stranice.....	26
Slika 5.1 Struktura tablice s varijablama .....	27
Slika 5.2 Tablica vremenskih pojava.....	28
Slika 6.1 Izgled neba sa standardnim raspršenjem i apsorpcijom .....	30
Slika 6.2 Izgled neba kada se pojača raspršenje .....	30
Slika 6.3 Izgled neba kada se pojača apsorpcija.....	31
Slika 6.4 Skala apsorpcije i raspršenja .....	31
Slika 6.5 Instanca materijala za oblake .....	32
Slika 6.6 Dio logike materijala oblaka .....	32
Slika 6.7 Parametar oblačnosti je podešen na -0.3 .....	33
Slika 6.8 Parametar oblačnosti je podešen na 0.05.....	34
Slika 6.9 Sustav čestica za snijeg .....	35
Slika 6.10 Opcije emitera .....	36
Slika 6.11 Kreirani emiter s postavkama.....	37

Slika 6.12 Oblik lokacije emitera .....	37
Slika 6.13 Brzina i smjer čestica .....	38
Slika 6.14 Trajanje čestica.....	38
Slika 6.15 Ažuriranje čestica .....	39
Slika 6.16 Sustav čestica za kišu .....	39
Slika 6.17 Veličina čestice.....	40
Slika 6.18 Smjer i brzina kretanja čestica kiše .....	40
Slika 6.19 Materijal za odbijanje kapi kiše.....	40
Slika 6.20 Događaj za kišu .....	41
Slika 6.21 Događaj za raspršenje.....	41
Slika 6.22 Pojavljivanje groma.....	42
Slika 6.23 Zvuk grmljavine .....	43
Slika 6.24 Materijal za maglu.....	44
Slika 6.25 Sustav čestica za maglu.....	45
Slika 6.26 Postepeno pojavljivanje i nestajanje čestica.....	45
Slika 6.27 Kolekcija parametara materijala suhoće.....	46
Slika 6.28 Primjena parametra suhoće na materijal .....	47
Slika 6.29 Suhoća vrijednosti 1.0 .....	47
Slika 6.30 Suhoća vrijednosti 0.5 .....	47
Slika 6.31 Kolekcija čvorova koja simulira akumulaciju snijega na površini objekata .....	48
Slika 6.32 Kolekcija parametara materijala sniježnosti.....	49
Slika 6.33 Oba parametra sniježnosti na klupi su 0.....	50
Slika 6.34 Parametri sniježnosti na klupi su podešeni.....	50
Slika 6.35 Istovremenost za glazbu i zvukove ambijenta.....	51
Slika 6.36 Pravila za zvukove.....	52
Slika 6.37 Vedro nebo .....	53
Slika 6.38 Podaci u tablici za vedro nebo.....	53
Slika 6.39 Kiša.....	53
Slika 6.40 Podaci u tablici za kišu .....	53
Slika 6.41 Grmljavina s kišom .....	53
Slika 6.42 Podaci u tablici za kišu .....	53
Slika 6.43 Snijeg.....	54
Slika 6.44 Podaci u tablici za snijeg .....	54
Slika 6.45 Magla.....	54
Slika 6.46 Podaci u tablici za maglu.....	54

Slika 7.1 Prikaz Blueprint-a koji prikazuje vremenske pojave .....	55
Slika 7.2 Poziv na server .....	56
Slika 7.3 Dohvaćanje potrebnog polja.....	56
Slika 7.4 Funkcija koja dohvaća podatke iz ćelija tablice .....	57
Slika 7.5 Varijable povezane s akterima na sceni.....	57
Slika 7.6 Funkcija mijenja toplinu i intenzitet sunca.....	58
Slika 7.7 Funkcija mijenja boju i jačinu svjetlosnih komponenti.....	59
Slika 7.8 Funkcija koja mijenja način raspršenja sučeve svjetlosti.....	59
Slika 7.9 Funkcija koja kontrolira gustoću oblaka .....	60
Slika 7.10 Funkcija koja određuje suhoću materijala.....	61
Slika 7.11 Funkcija koja određuje akumulaciju snijega .....	61
Slika 7.12 Funkcija koja kontrolira glazbu.....	62
Slika 7.13 Funkcija koja kontrolira ambijent .....	62
Slika 7.14 Čvorovi koji dohvaćaju Niagara sustave na scenu .....	63
Slika 7.15 Funkcija koja miče nepotrebne efekte.....	64
Slika 8.1 Blueprint i sučelje koji otvaraju i zatvaraju prozor pritiskom tipke igrača .....	65
Slika 8.2 Sastavni dijelovi Blueprint-a .....	65
Slika 8.3 Prikaz svih komponenti u pregledniku Blueprint-a.....	66
Slika 8.4 Blueprint koji otvara i zatvara prozor.....	66
Slika 8.5 Sučelje Otvaranje povezano je s Blueprint-om za prozor .....	67
Slika 8.6 Funkcija otvaranja i zatvaranja prozora .....	67
Slika 8.7 Kontrola nad otvaranjem i zatvaranjem prozora .....	68
Slika 9.1 Izgled interaktivnog sučelja.....	69
Slika 9.2 Sastavni dijelovi sučelja .....	70
Slika 9.3 Kontroliranje sučelja .....	70
Slika 9.4 Gumbi iz sučelja koji kontroliraju događaje .....	71
Slika 9.5 Opisi akcija gumbova.....	71
Slika 9.6 Primjer interakcije interaktivnog sučelja s API Blueprintom .....	72