



PCS3539
Tecnologia de Computação Gráfica

Hopper Island

Disciplina: PCS3539 - Tecnologia de Computação Gráfica

Prof.: Ricardo Nakamura

Data: 04/07/2023

Prof.: Romero Tori

Membros:

11302626 - Caio da Costa Gossi

11260832 - Roberta Boaventura Andrade

11259715 - Vanderson da Silva dos Santos

São Paulo
2023

Sumário

Sumário.....	2
1 - Introdução.....	3
2 - Descrição.....	3
3 - Objetivo.....	5
4 - Manual do Usuário.....	5
4.1 - Controles principais.....	5
4.2 - Controles de Depuração.....	6
4.3 - Personagens e Habilidades.....	6
4.4 - Interface do Usuário.....	6
5 - Ferramentas Utilizadas.....	10
6 - Processo de Desenvolvimento.....	10
6.1 - Meshes Modeladas.....	11
• Ilha principal.....	11
• Ilha gelo.....	12
• Ilha Vulcão.....	13
• lago.....	14
6.2 - Meshes Texturizadas.....	19
• Ilha principal.....	20
• Ilha gelo.....	22
• Ilha Vulcão.....	22
• lago.....	23
6.3 - Meshes Importadas.....	25
6.4 - Personagem.....	31
6.5 - Iluminação.....	32
6.6 - Filtros, post-processing e partículas.....	34
6.7 - Lógica de jogo e programação.....	37
6.8 - Otimizações.....	39
7 - Objetivos Atingidos.....	40
8 - Possíveis Melhorias.....	41
8 - Referências.....	42
9 - Links.....	43

1 - Introdução

Buscando explorar e expandir sobre os conceitos aprendidos durante a primeira parte da disciplina, o grupo optou por implementar uma "demo" de um jogo de mundo aberto, no qual o jogador poderá andar e explorar o mapa do jeito que bem entender, descobrindo por conta própria aquilo que o jogo tem a oferecer.

A escolha de um mundo aberto baseou-se principalmente nos jogos *The Legend of Zelda: Breath of the Wild* e *The Legend of Zelda: Tears of the Kingdom*, que são mundos abertos de alta complexidade e liberdade ao jogador. Tais jogos abrem possibilidades de navegar para onde quiser do mapa com conta e responsabilidade do jogador.

Tal conceito foi escolhido pois, além da grande presença de jogos deste gênero na indústria dos videogames, essa é uma boa forma de permitir a exploração de diversos conceitos gráficos diferentes que podem ser usados para compor um mundo inteiro, neste caso com destaque a uma paisagem natural, com foco em estruturas como montanhas, vales, florestas, rios e afins.

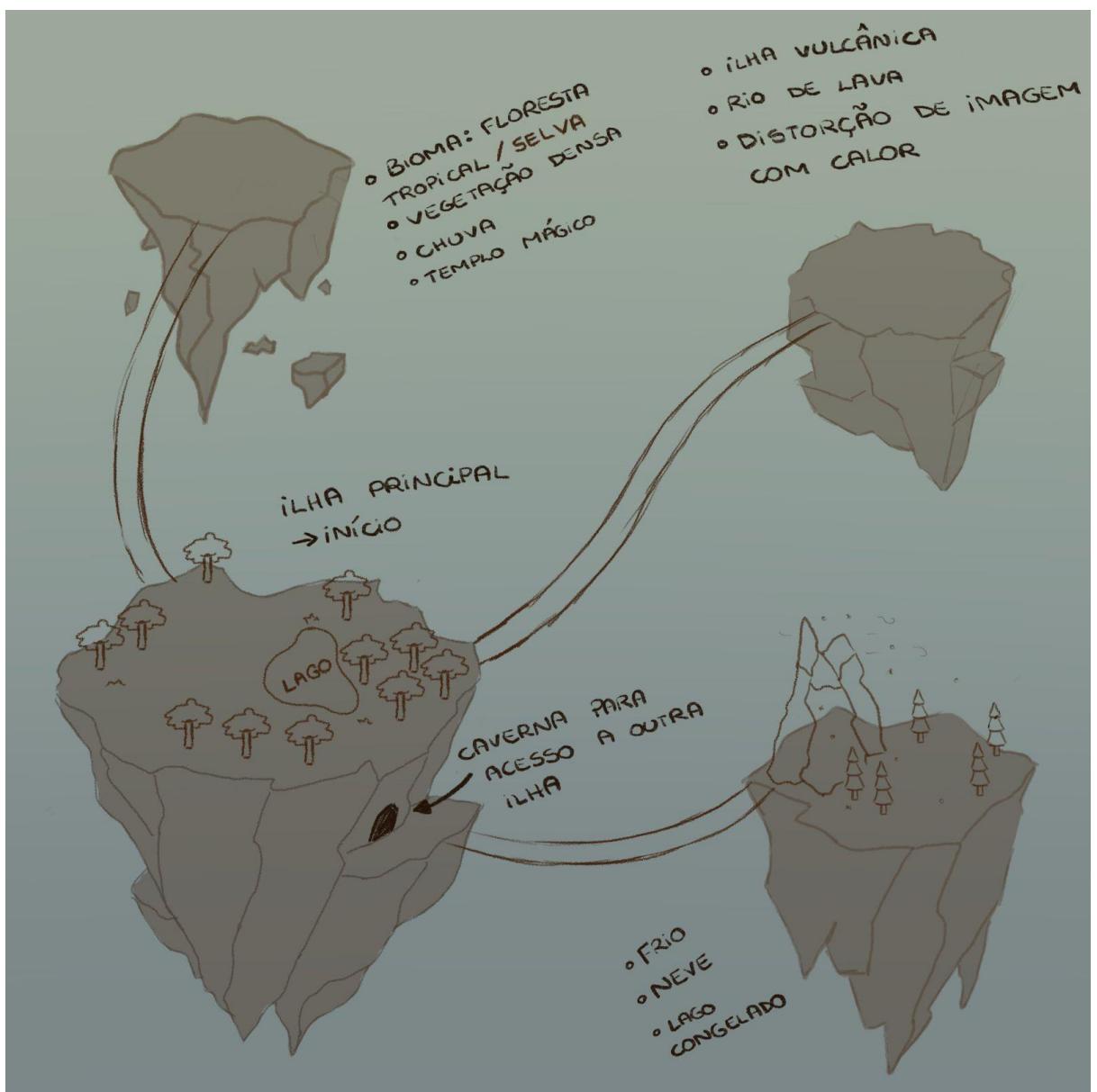
A partir deste trabalho, então, podem ser explorados, além do projeto de criação de um jogo, diversas diferentes técnicas de modelagem, texturização de vários materiais, shaders e efeitos de partícula e iluminação. O futuro trabalho e seus componentes serão descritos a seguir.

2 - Descrição

A demonstração do jogo possuirá um grande mapa principal, que carregará em si os elementos para a demonstração. O mapa desenvolvido foi um sistema de ilhas flutuantes, de modo que cada uma possui seu próprio bioma para que pudéssemos aplicar os conceitos aprendidos em sala de aula por meio de vários efeitos e ambientes tematizados.

O design inicial do mapa foi pensado como uma ilha principal na qual o jogador iria nascer e, a partir dela, poderia explorar as demais. Desse modo, todas as ilhas estariam conectadas à ilha principal a partir de pontes, como foi idealizado

no primeiro modelo desenhado, que foi utilizado como base para o desenvolvimento do projeto:



No entanto, devido à dificuldade de implementar um bom modelo de pontes extensas, assim como o fato de que a exploração se tornou demorada e repetitiva por conta do tamanho do mapa, foi decidido permanecer apenas com três das quatro ilhas originais: a principal, a de gelo e a vulcânica. Do mesmo modo, foi implementado um sistema de teleporte com o foco em melhorar a experiência do usuário ao jogar.

Assim, o mapa final consiste de uma ilha central, com uma diversidade de árvores, vegetação rasteira e um rio que a corta no meio. Acima dela há a ilha gelo, com temática fria, efeito de nevasca, pinheiros e montanhas altas. Nessa ilha há um lago que se desfaz em uma cachoeira. Um pouco para baixo da ilha principal, há a ilha vulcânica, com tema quente e rochoso, vegetação morta e um vulcão do qual escorre lava.

3 - Objetivo

O propósito do jogo desenvolvido é servir como uma “demo” de um estilo de mundo aberto, focado na exploração tanto no quesito visual como nos recursos de jogabilidades para tornar certas áreas do mapa acessíveis. Com isso, a finalidade desse projeto é desenvolver as habilidades com criação de jogos, além da aplicação dos conceitos vistos em sala de aula acerca de computação gráfica ao longo do semestre. O jogador tem o objetivo de coletar determinados baús de tesouro dispostos pelo mapa para desbloquear habilidades que tornem mais fácil a exploração do mundo, enquanto se aventura no cenário elaborado.

4 - Manual do Usuário

Para inicializar o jogo, é preciso fazer download dos arquivos disponíveis no GitHub e abrir o “StartWorld.uproject” a partir do programa “Unreal Engine 5.2.0”. A partir dele, deve-se iniciar o jogo apertando o botão de play. Na tela inicial do jogo, é possível escolher em que ilha o jogador irá nascer e a jogabilidade é definida a partir de alguns comandos listados abaixo.

4.1 - Controles principais

Controles	Descrição
-----------	-----------

w	Andar para frente
a	Andar para a esquerda
s	Andar para trás
d	Andar para direita
m	Abre/Fecha o Mapa
espaço	Pulo
SHIFT + espaço	Super pulo
CTRL + w a s d	Alta velocidade
esc	Abre/Fecha menu pause

4.2 - Controles de Depuração

Controles	Descrição
Q	Habilita/desabilita passagem do tempo
P	Muda velocidade de passagem do tempo
R	Muda multiplicador de velocidade do personagem

4.3 - Personagens e Habilidades

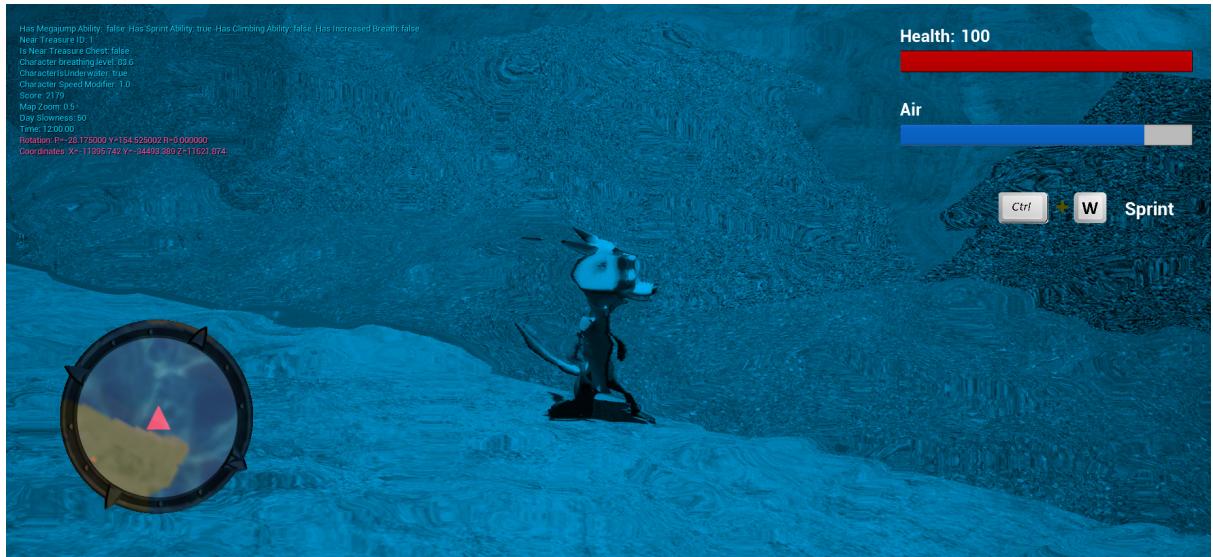
O jogo em questão possui um protagonista jogável chamado Nick Wilde, inspirado no personagem do filme Zootopia. Durante o decorrer do jogo, Nick tem a capacidade de adquirir habilidades especiais, como um super salto e uma velocidade de corrida aumentada.

4.4 - Interface do Usuário

O jogo possui várias interfaces com o usuário, incluindo o menu principal, o menu de escolha de ilhas, o menu de pausa, o menu de fim de jogo e o mapa.

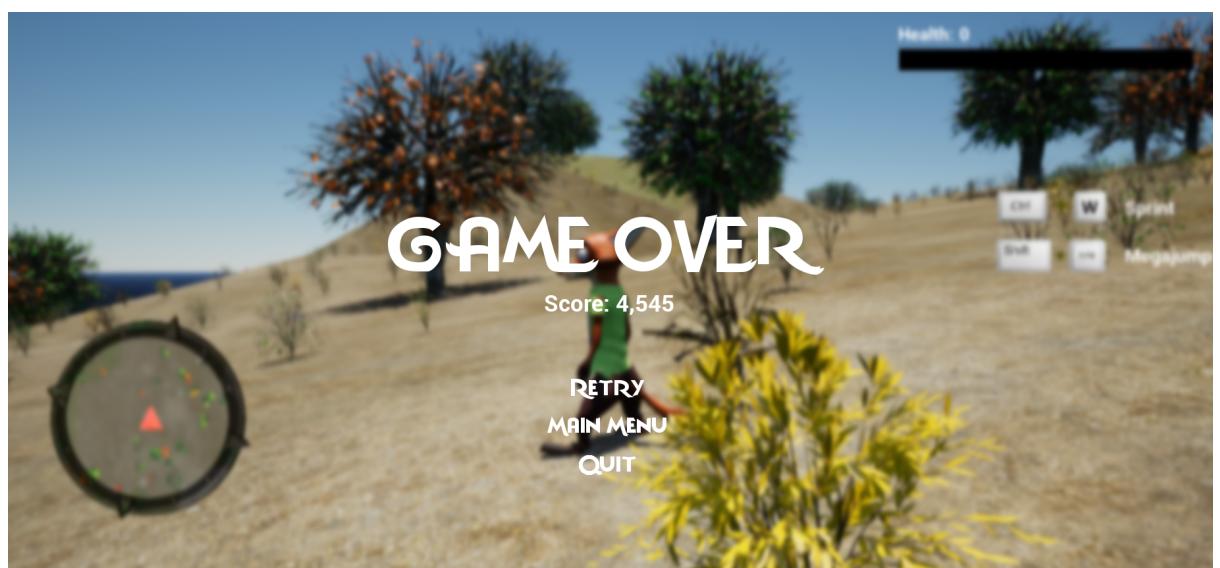
Além destes, existe também a interface principal, exibida durante o gameplay, contendo a vida do personagem, um minimapa, que se comporta de maneira dinâmica de acordo com a rotação da câmera e a posição do jogador no mapa, um medidor de respiração, que aparece enquanto o

personagem está submerso em água, e, finalmente, os controles das habilidades especiais, que aparecem conforme estas são adquiridas.

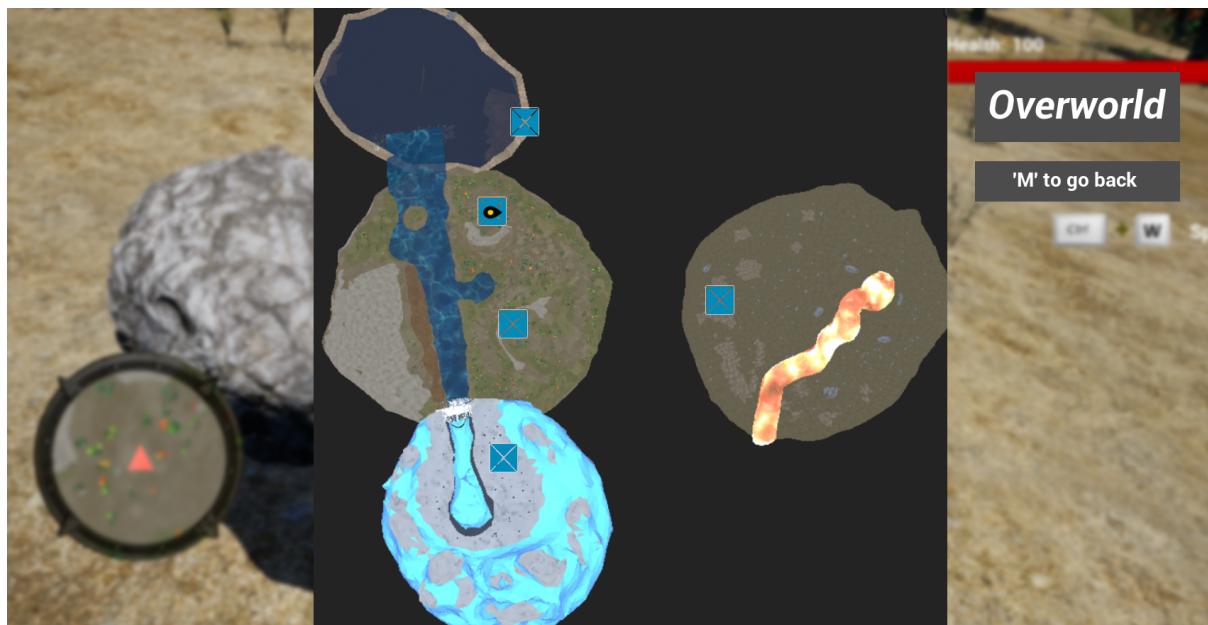


Os menus principais, de ilhas, de pausa e de fim de jogo são semelhantes, apresentando uma série de botões que realizam ações específicas.





O mapa oferece funcionalidades adicionais, como a capacidade de teletransportar entre pontos do mapa e localizar o jogador nas diferentes ilhas em que ele se encontra, utilizando um marcador, semelhante ao encontrado em diversos jogos atuais.



5 - Ferramentas Utilizadas

As ferramentas utilizadas para desenvolvimento do jogo foram os softwares “Blender 3.5” para modelagem e texturização dos objetos, além do “Unreal Engine 5.2.0” da Epic Games como game engine para a devida criação do mundo, com foco na montagem do mapa e nos comandos para permitir a jogabilidade.

6 - Processo de Desenvolvimento

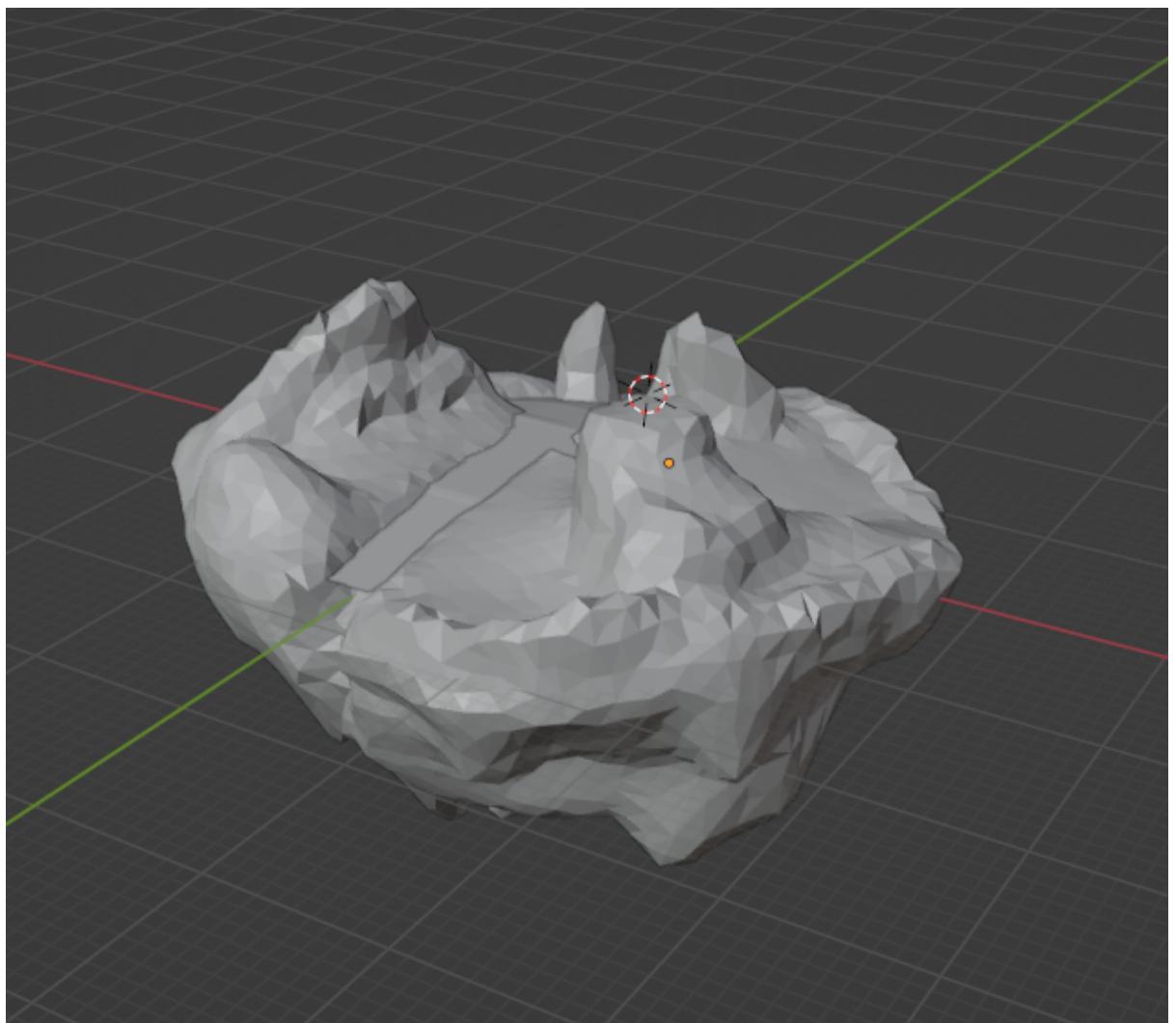
O processo de desenvolvimento foi pautado no cronograma pré-estabelecido enviado junto com a proposta do jogo. A criação decorreu a partir da modelagem das ilhas no Blender e da implementação delas no Unreal à medida em que as meshes ficavam prontas num sistema de esculpi-las, texturizá-las e importá-las no mapa. Após esse processo, foi dado um foco na aplicação da dinâmica dos fluidos como a cachoeira, o rio e a lava escorrendo, assim como o ajuste de iluminação. Finalizando, foi estruturada a jogabilidade com sistema de horário de dia e “power-ups” para avanço do personagem.

6.1 - Meshes Modeladas

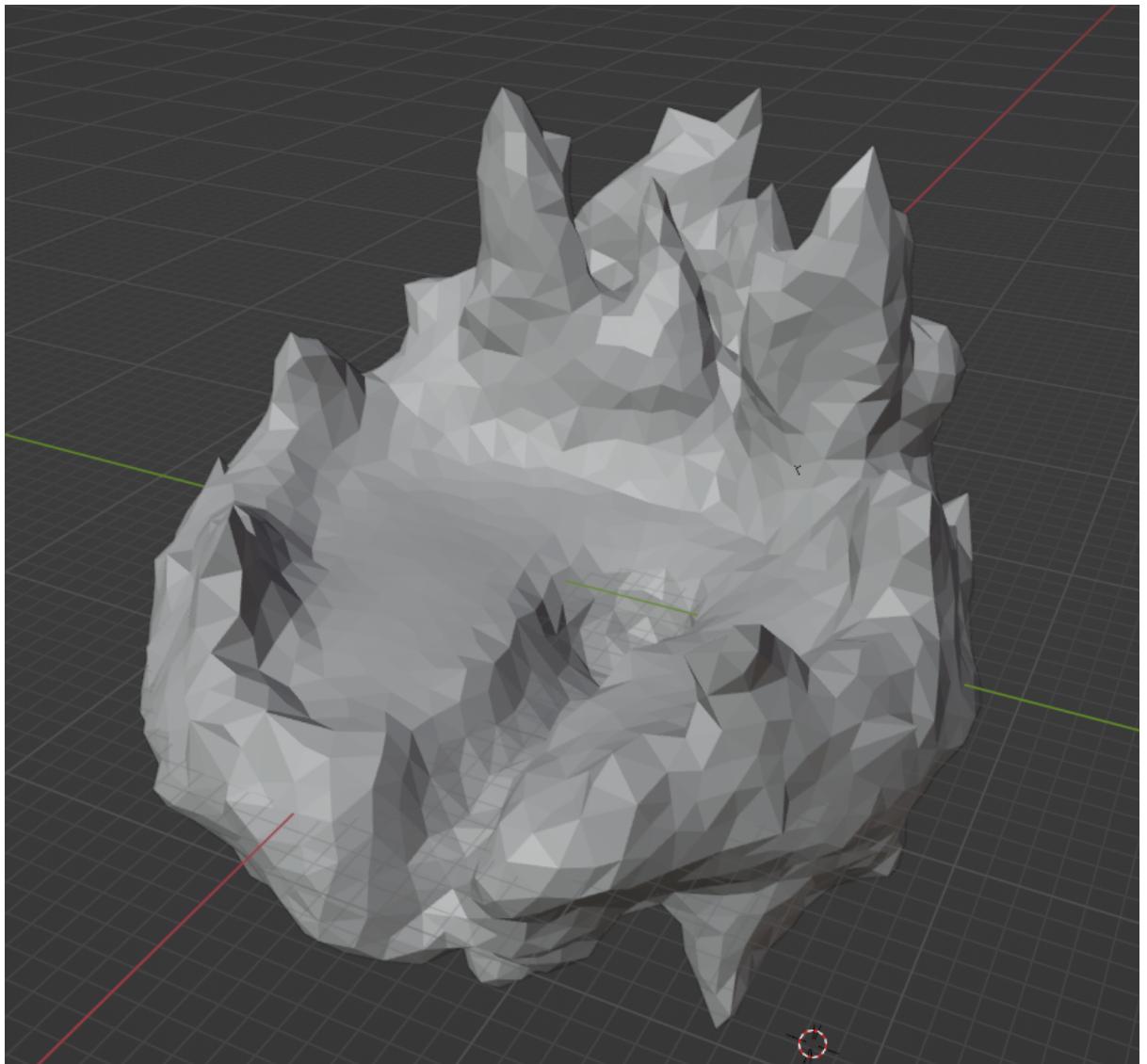
A modelagem de objetos foi feita através do programa “Blender 3.5”, de modo que o grupo esculpiu algumas assets que foram utilizadas para criar os ambientes desejados com o fim de que cada ilha possuísse um visual único e diferenciado das outras. Entre as meshes geradas, podemos destacar todas as ilhas presentes, além das árvores.

Lista de meshes modeladas:

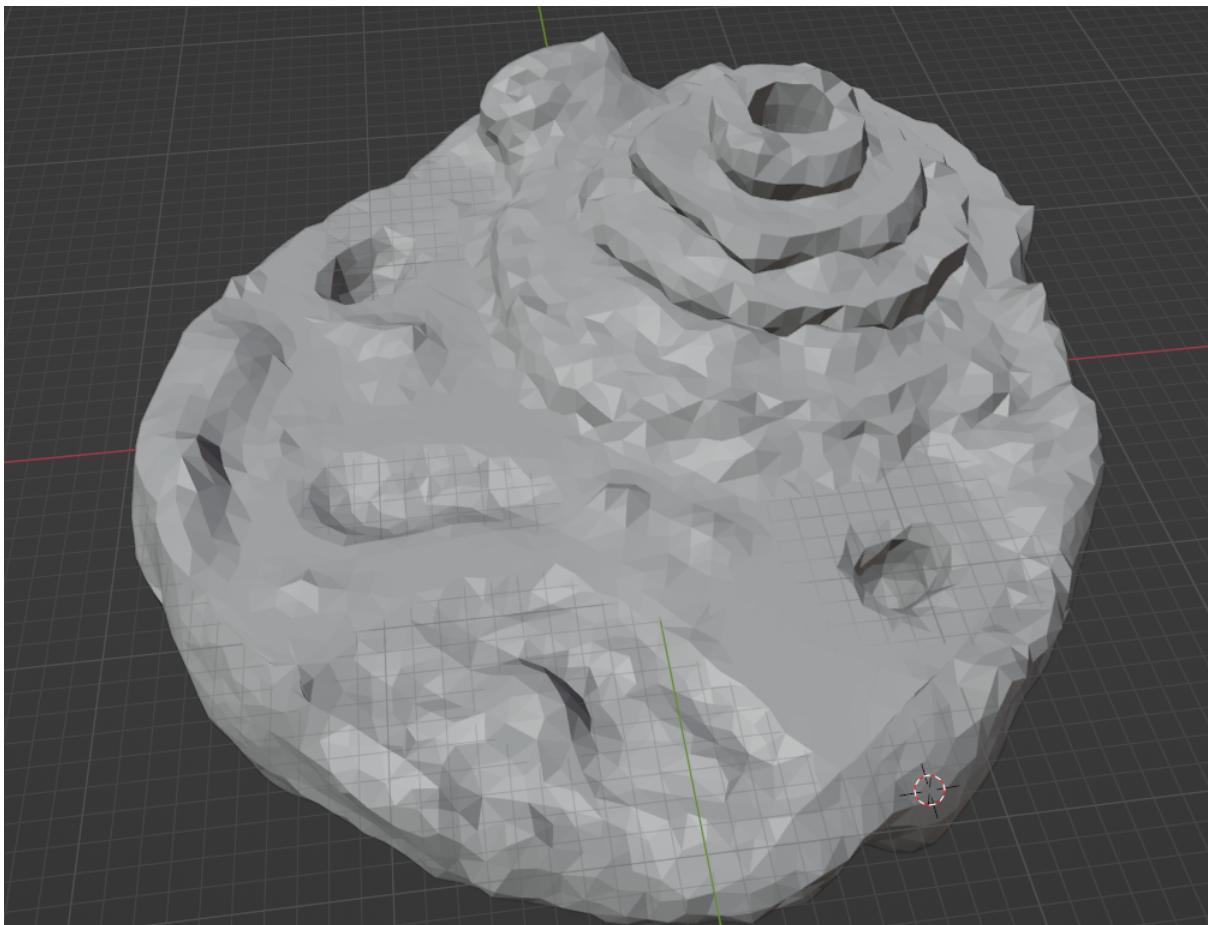
- Ilha principal



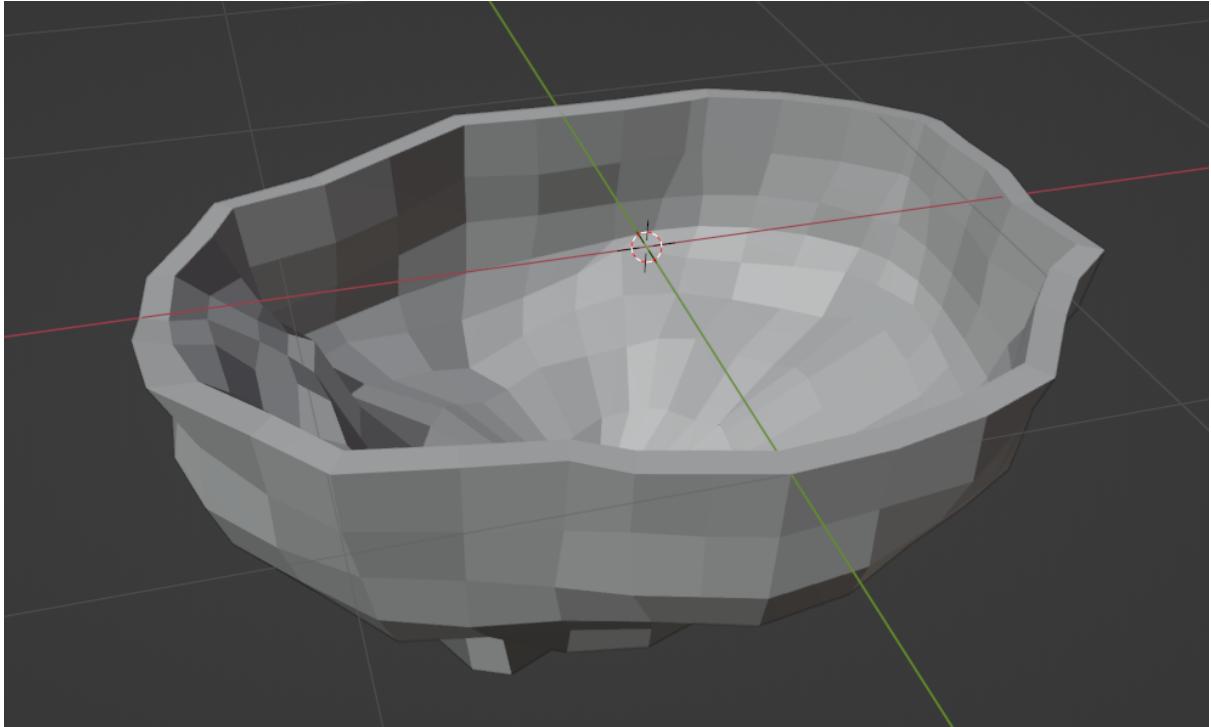
- Ilha gelo



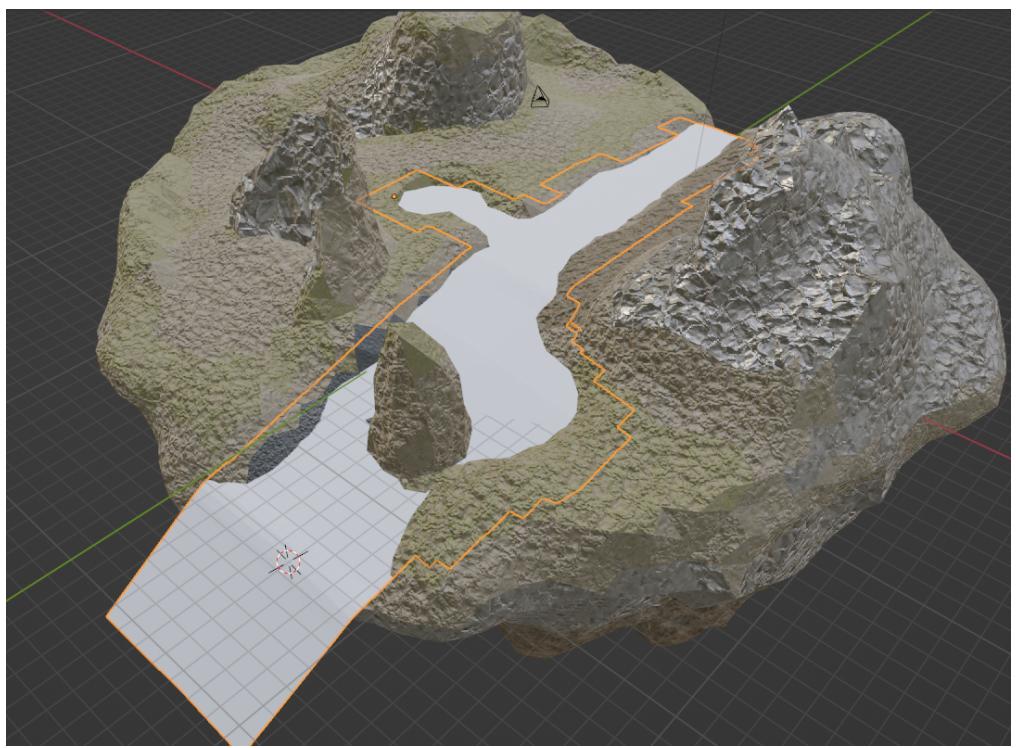
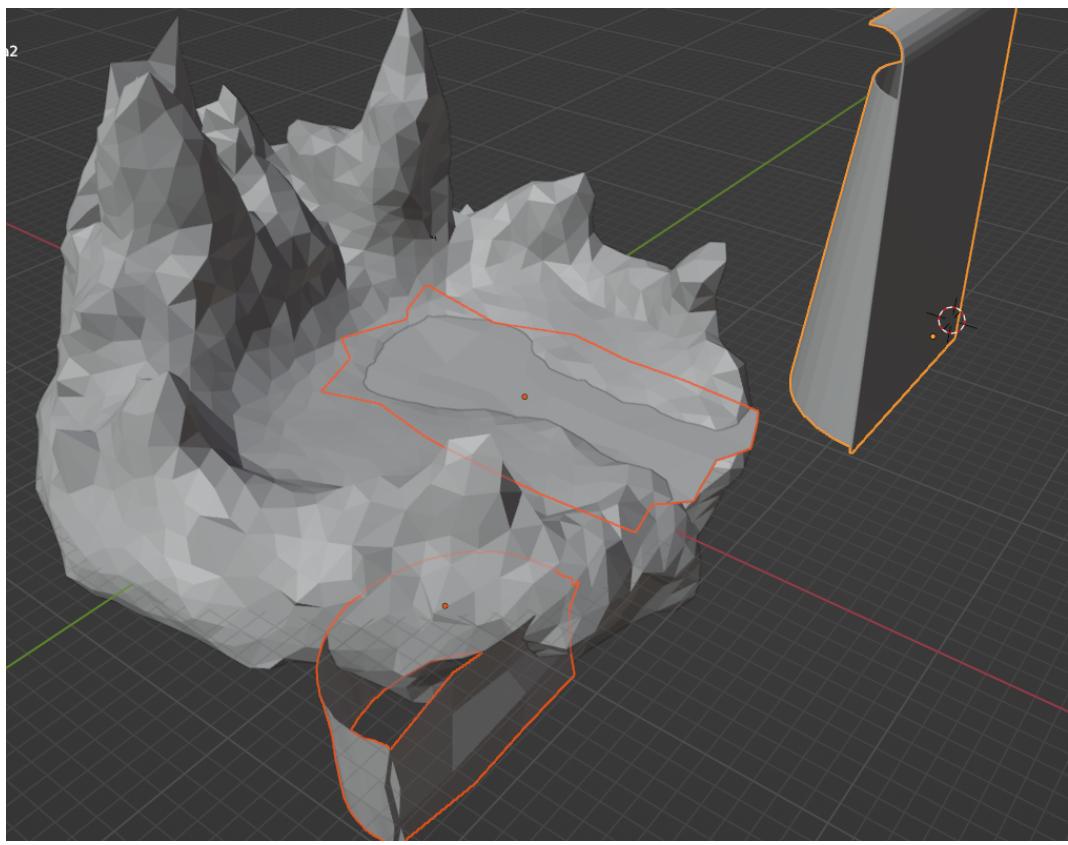
- Ilha Vulcão

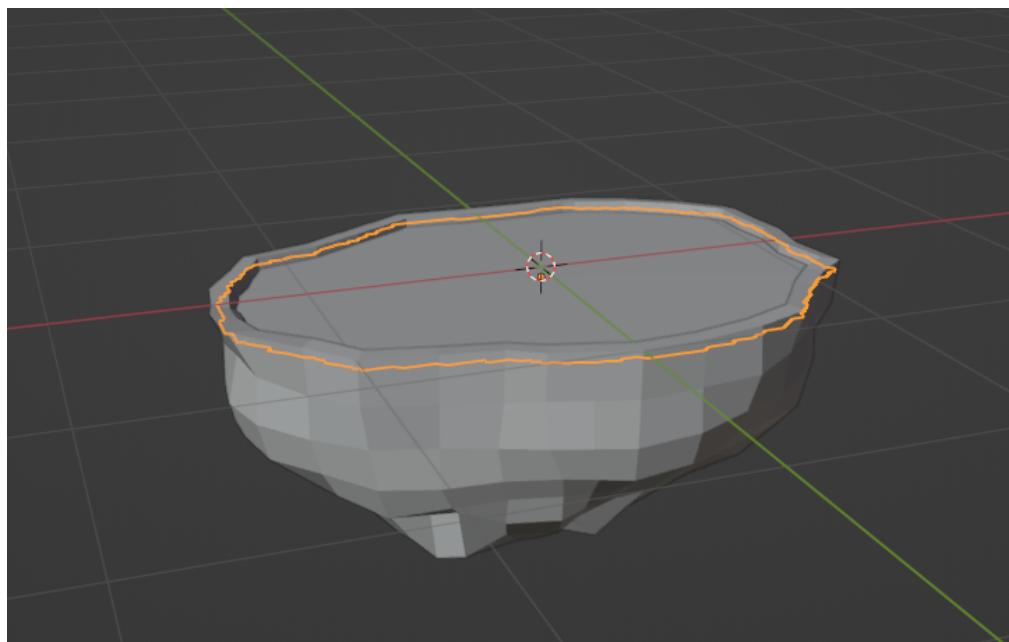
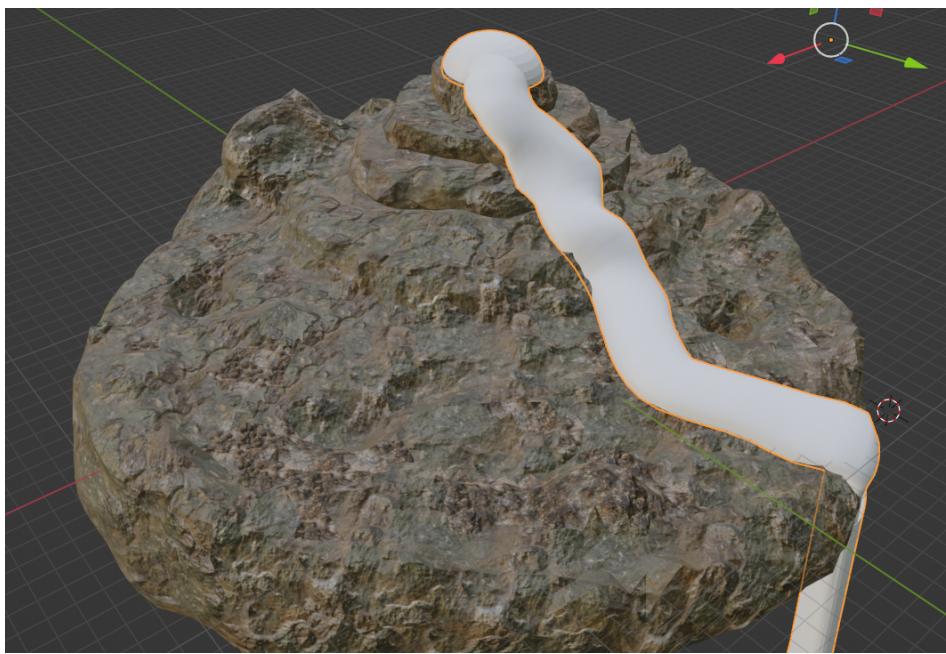


- lago

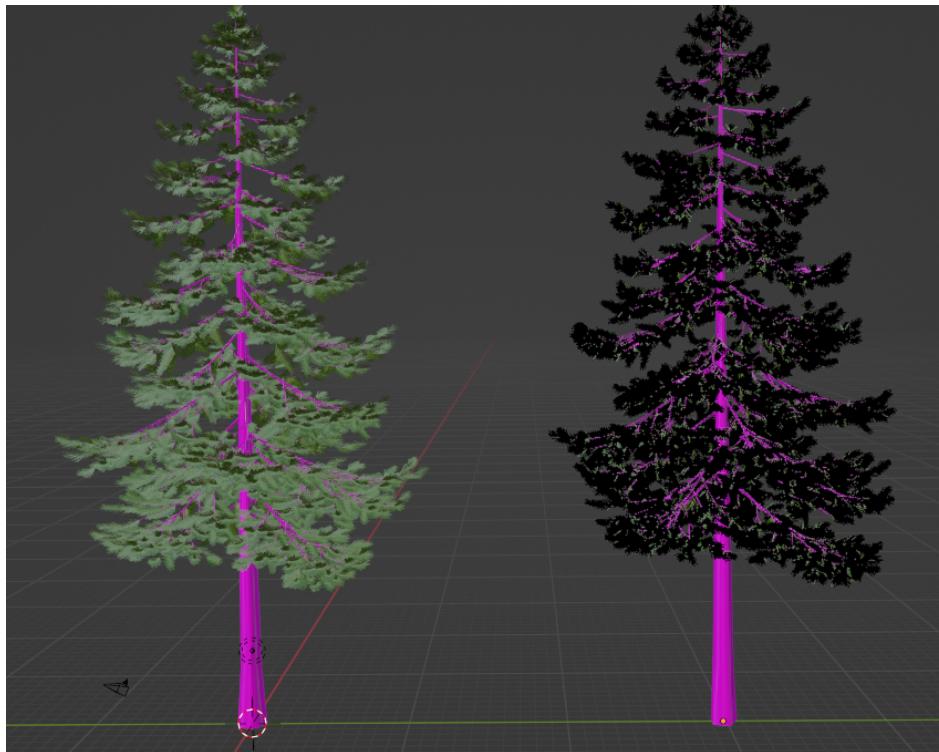
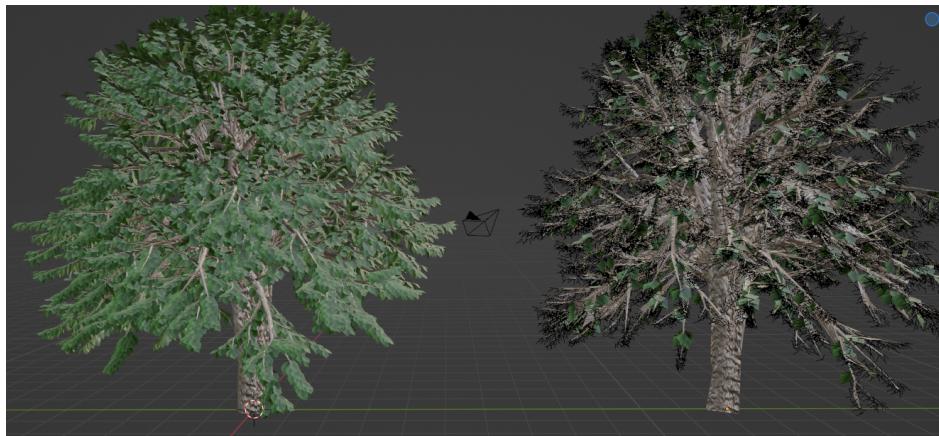
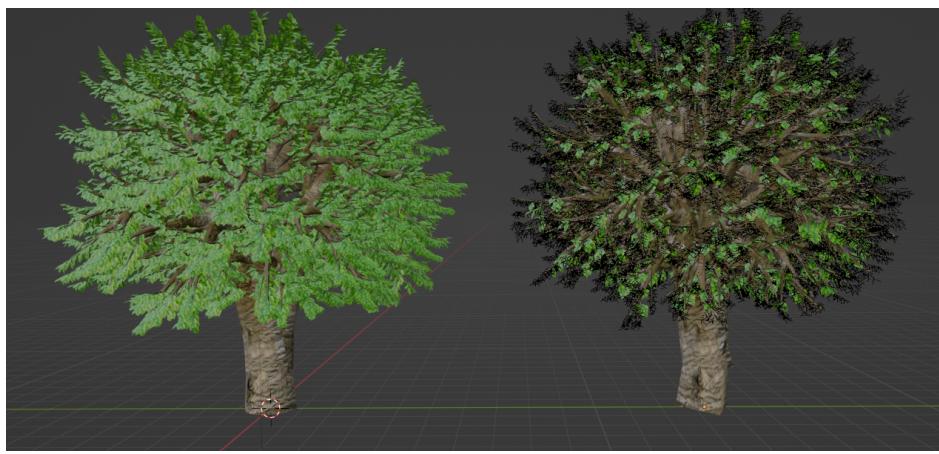


- Superfícies de água e lava (destacados nas figuras abaixo)

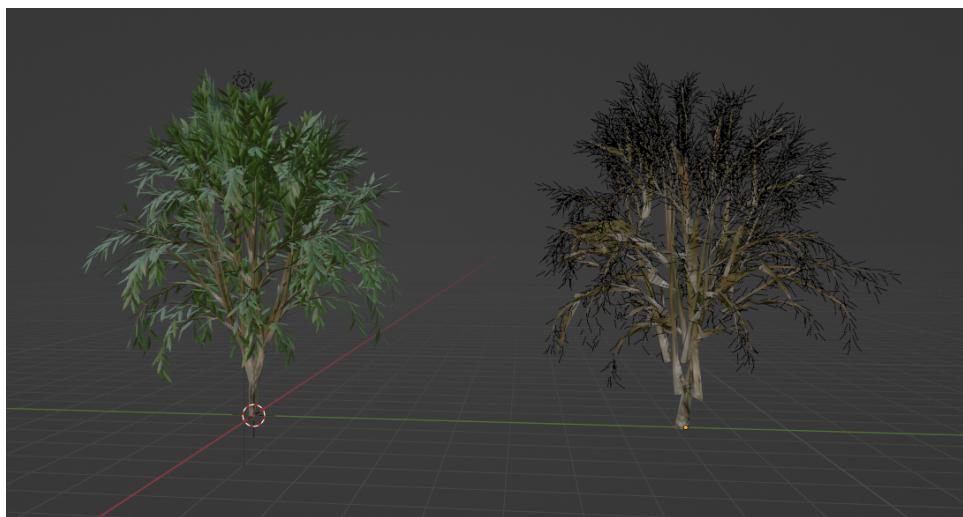


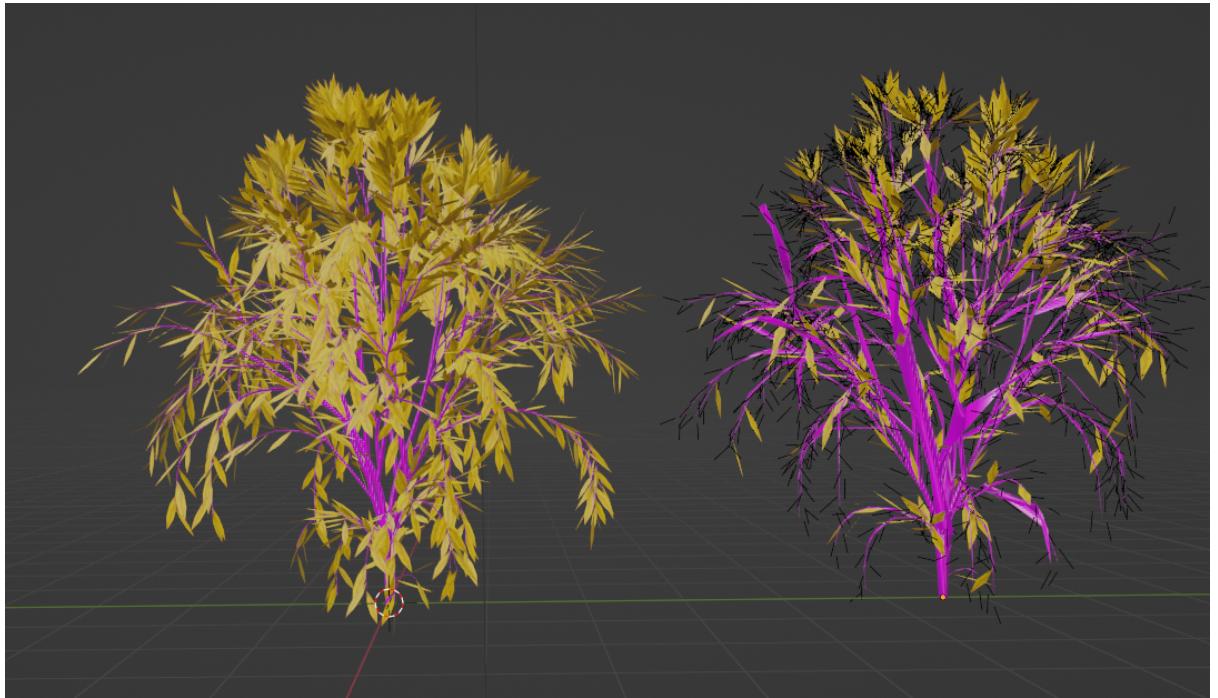


- Árvores grandes: 12 meshes diferentes, com LOD0 e LOD1 para cada árvore

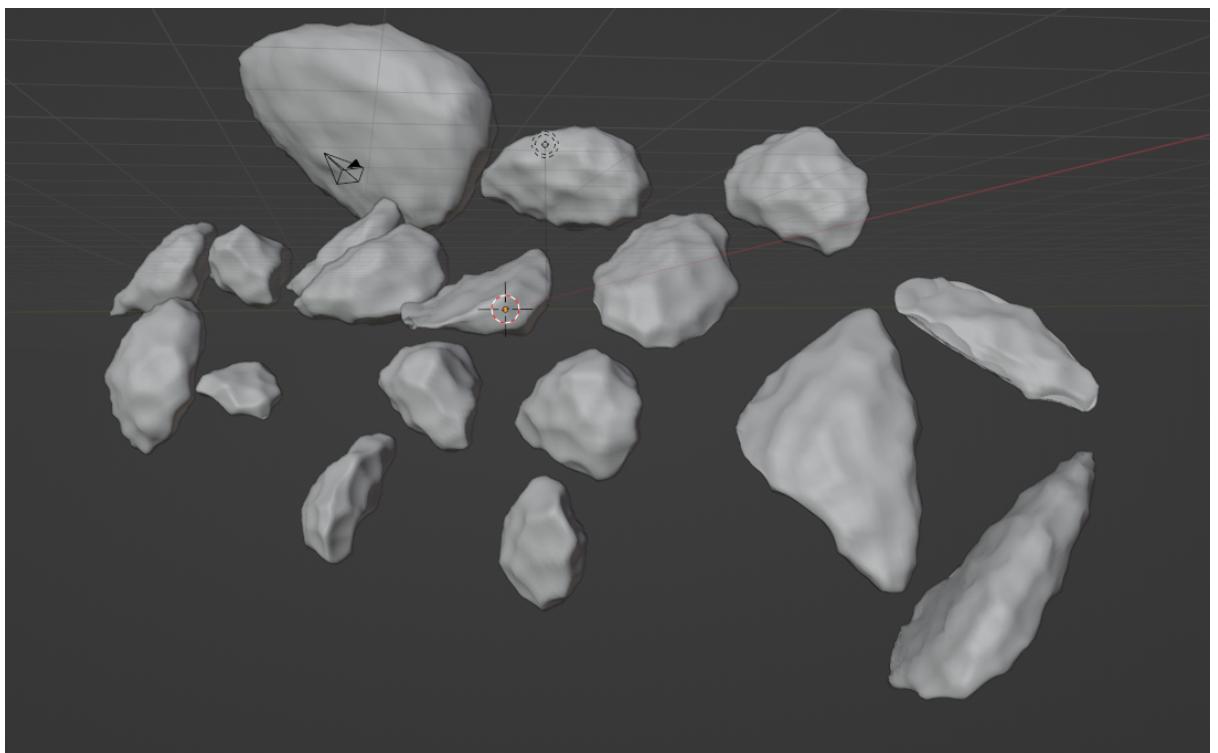


- Vegetação pequena: 11 meshes diferentes, com LOD0 e LOD1 para 7 destas





- Pedras: 17 meshes diferentes

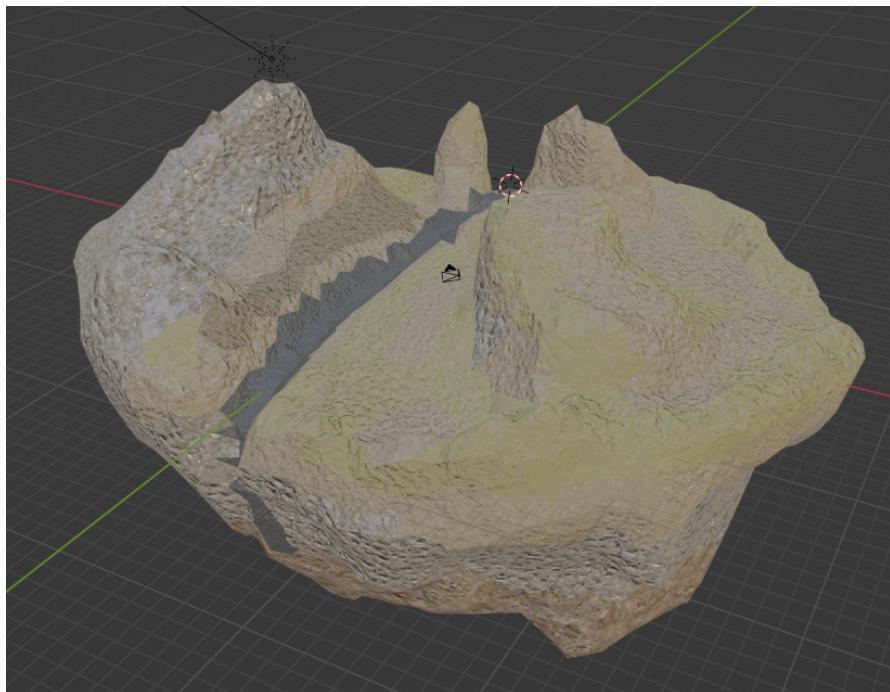


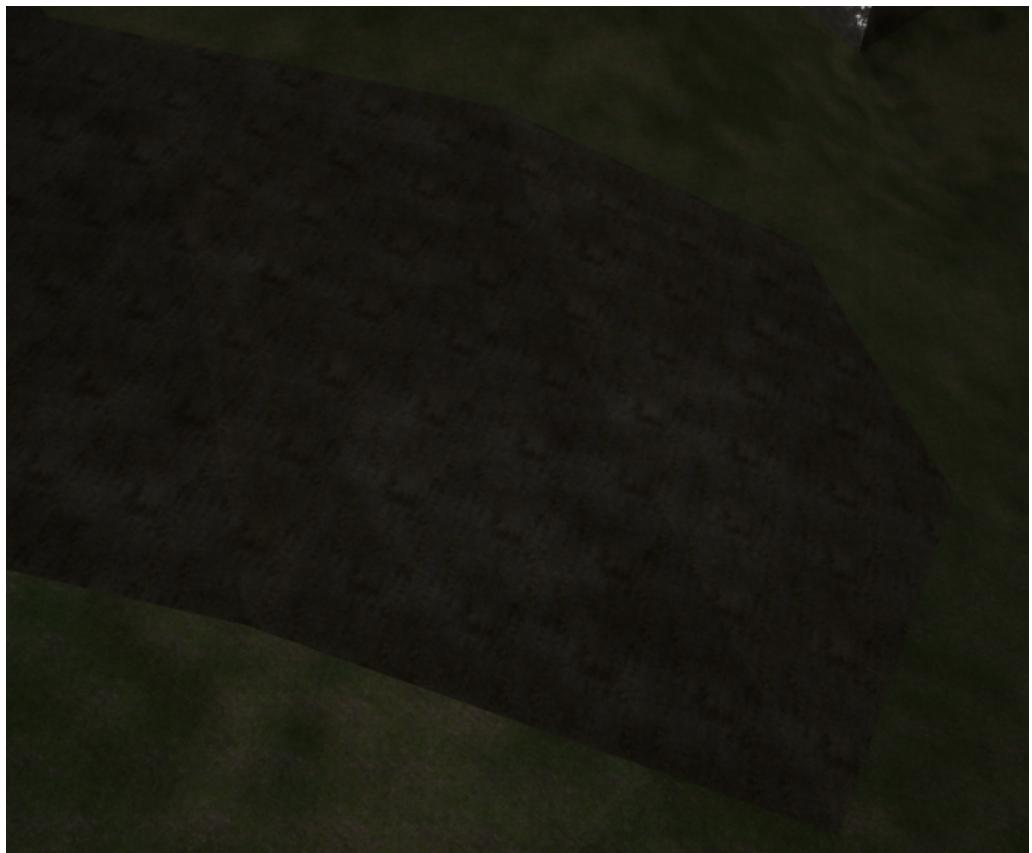
6.2 - Meshes Texturizadas

Para cada mesh modelada, foram criados diversos materiais do tipo PBR, partindo-se de texturas prontas obtidas gratuitamente online. Com estes materiais, então, e criando um mapa UV para cada modelo, foi possível texturizar cada mesh.

Sem dúvidas, um dos maiores desafios durante a criação de materiais e texturização das ilhas foi a aparente repetição das texturas. Já que mapas abertos são incrivelmente extensos e um dos objetos principais deste jogo é poder observar as paisagens à distância, muitas vezes o “tiling” ficava muito aparente. Assim, para tentar evitá-lo, foram utilizadas algumas técnicas rudimentares como macro e micro variação de textura utilizando tinting, e alguma rotação dos tiles pelo mapa utilizando funções embutidas na Unreal Engine. Os resultados foram satisfatórios.

- Ilha principal

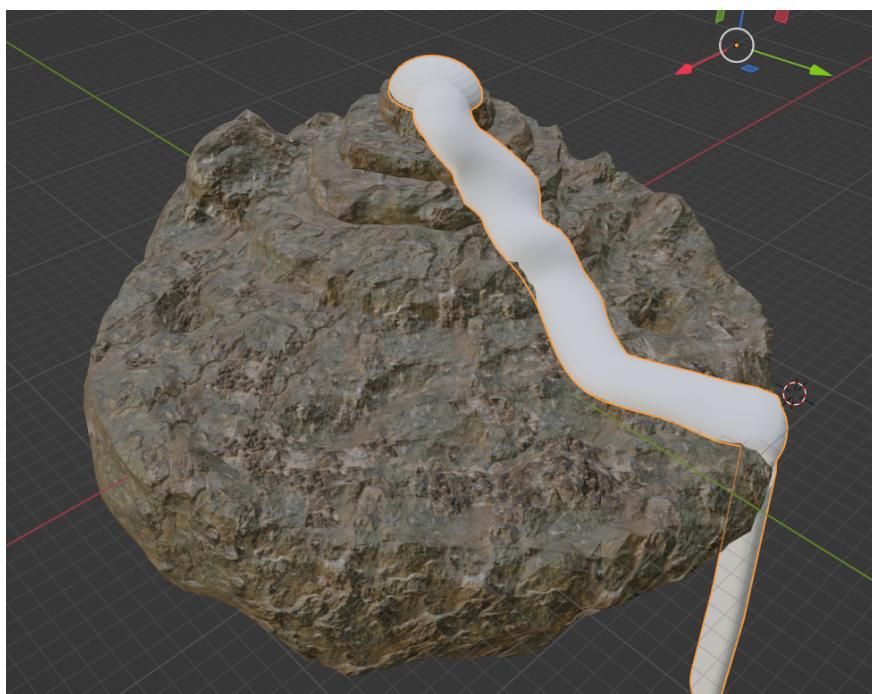




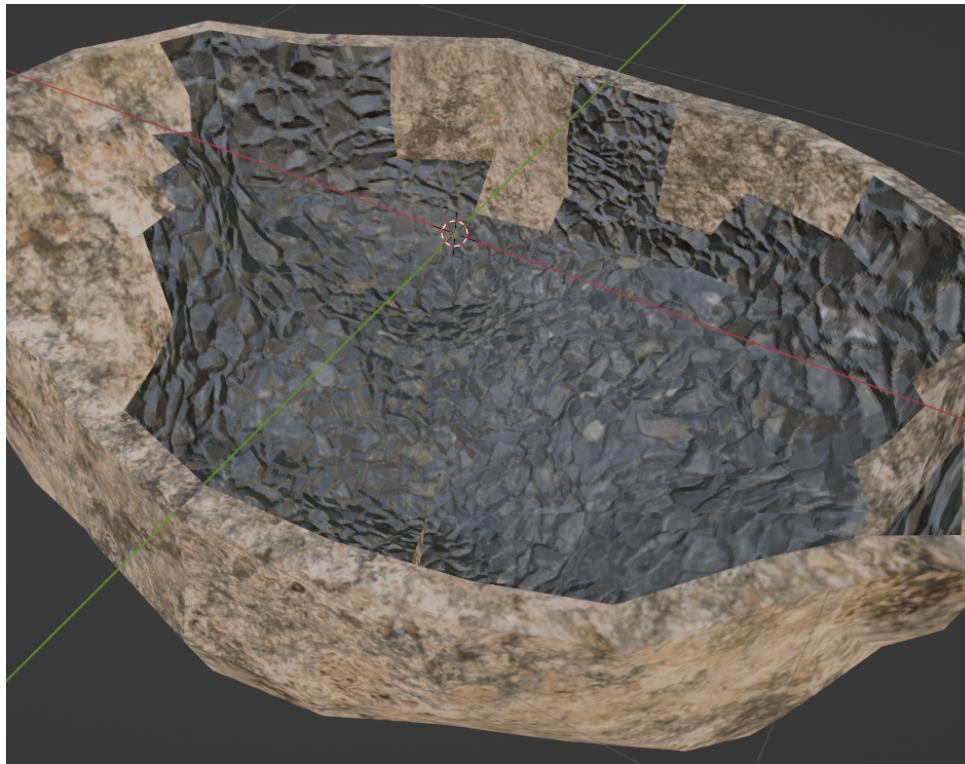
- Ilha gelo



- Ilha Vulcão

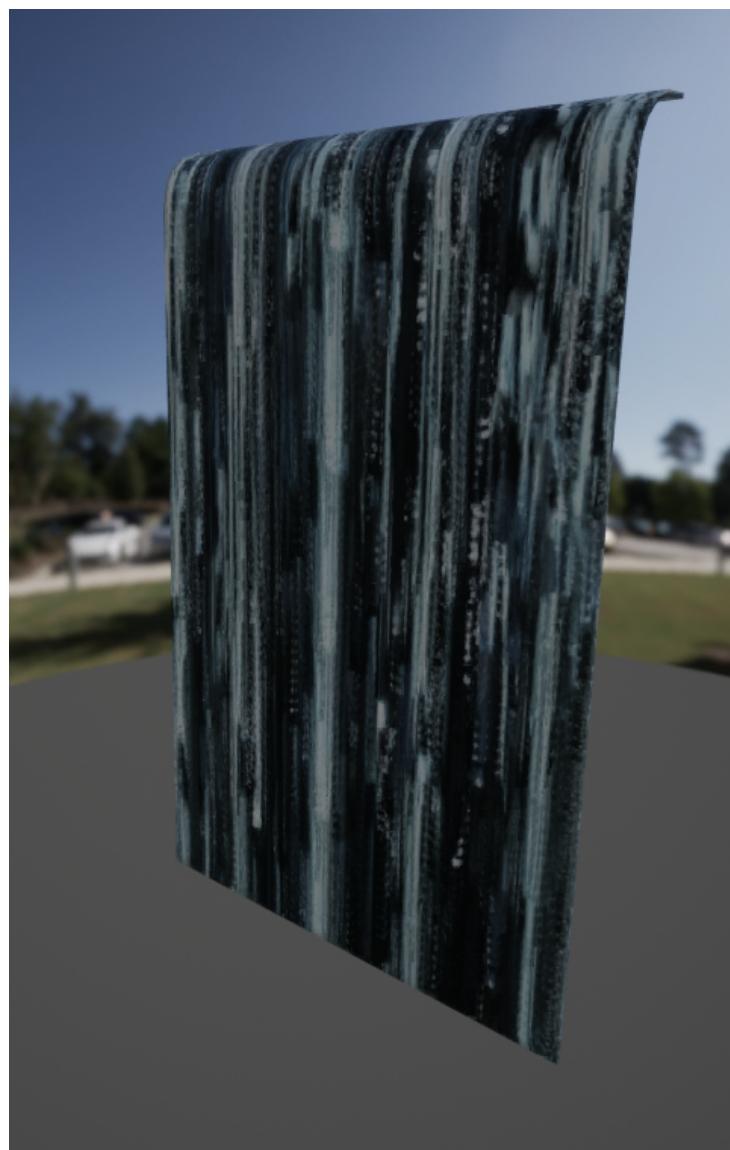
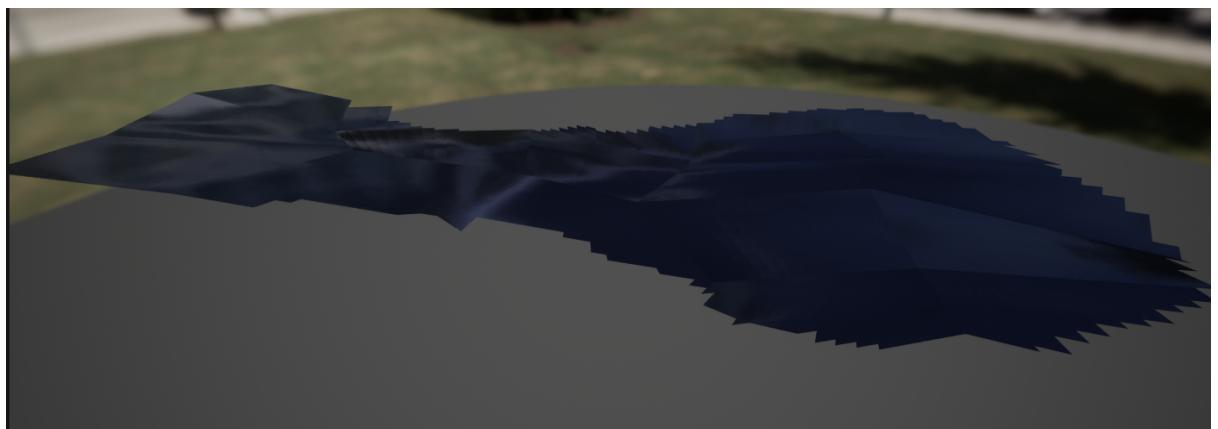


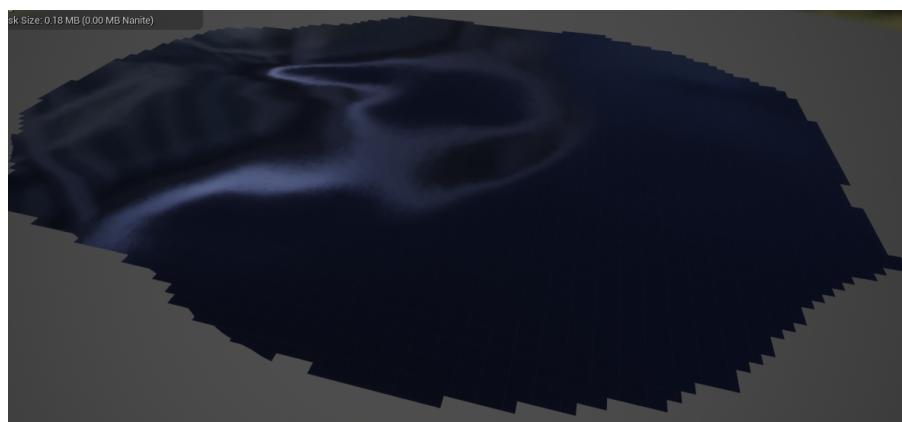
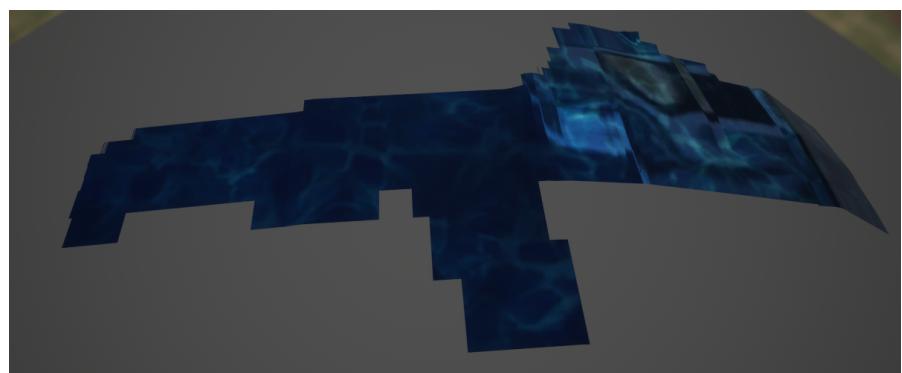
- lago



- Superfícies de água

Os materiais utilizados na texturização dos modelos de superfície de água foram baseados nos materiais originais disponibilizados pela Unreal Engine, com certas modificações para adicionar realismo. Já o material utilizado para a cachoeira foi desenvolvido inteiramente do zero.

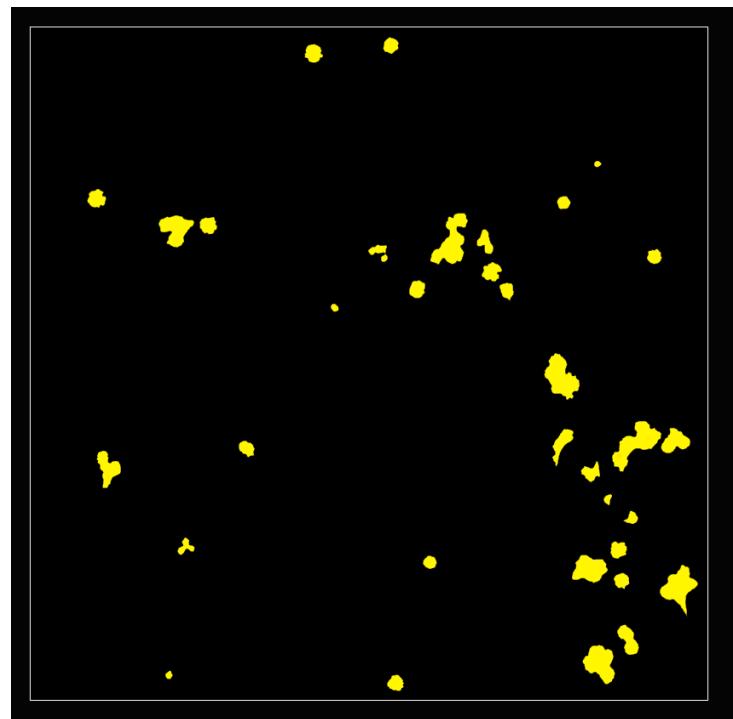
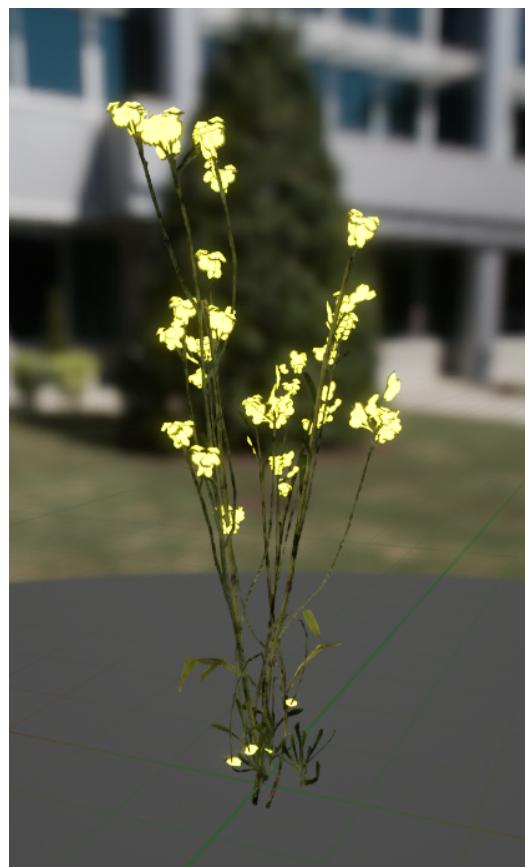


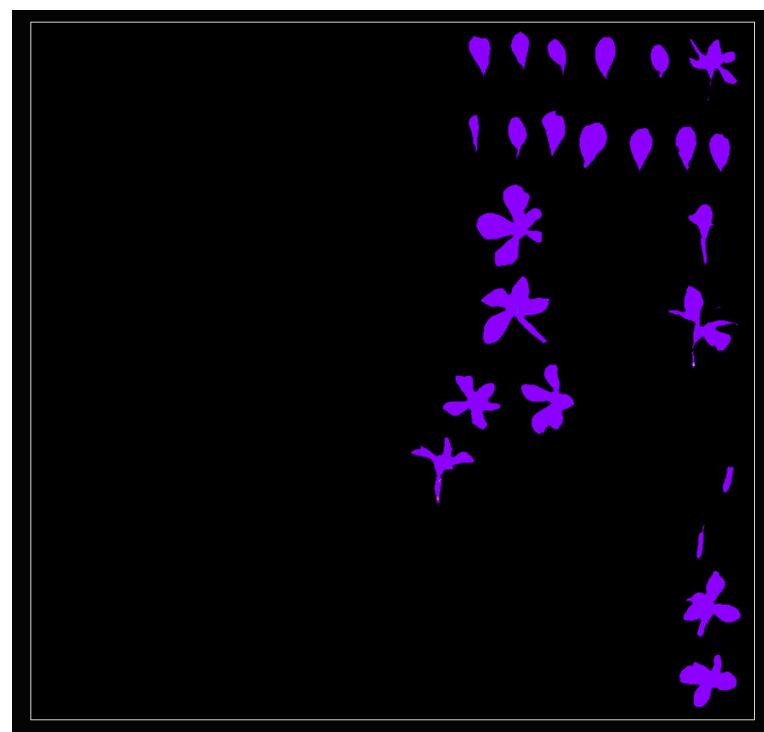


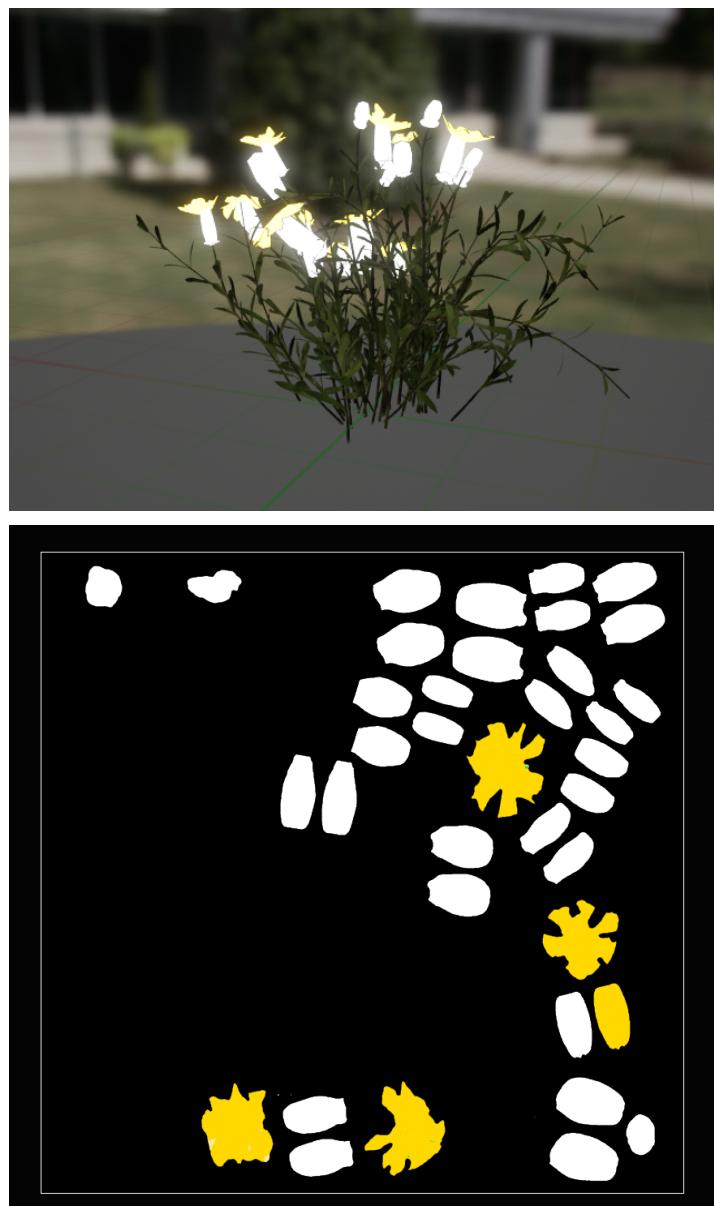
6.3 - Meshes Importadas

- Flores

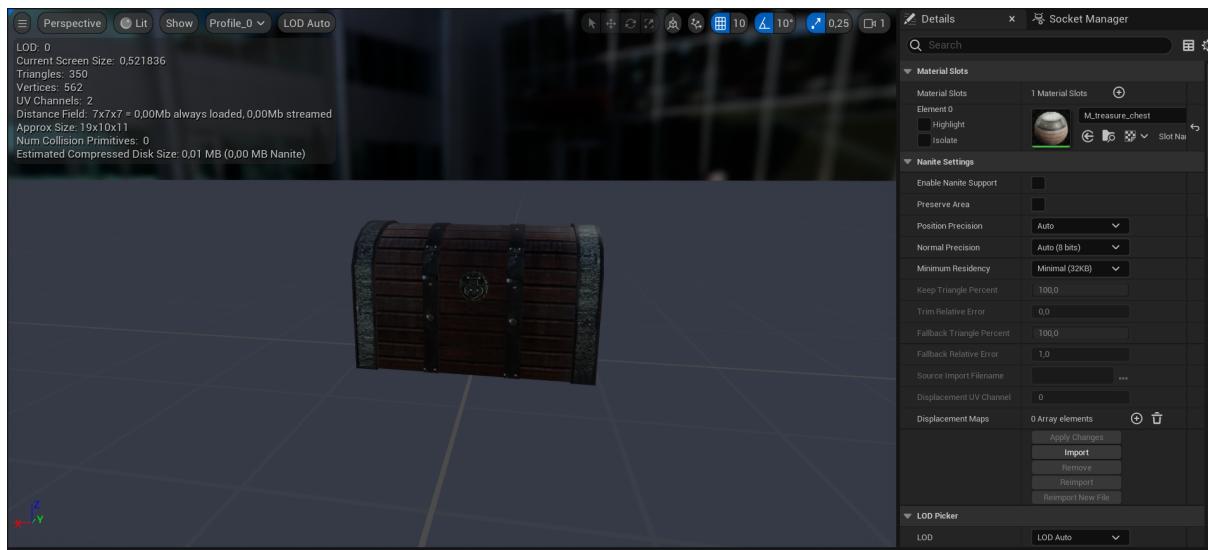
As meshes das flores foram baixadas através da plataforma integrada à Unreal Engine, Quixel Bridge. No entanto, foi, para cada tipo de flor, desenhada uma máscara de luminosidade responsável pela criação do material luminoso utilizado no projeto, o qual foi uma modificação do material original disponibilizado com os modelos originais:



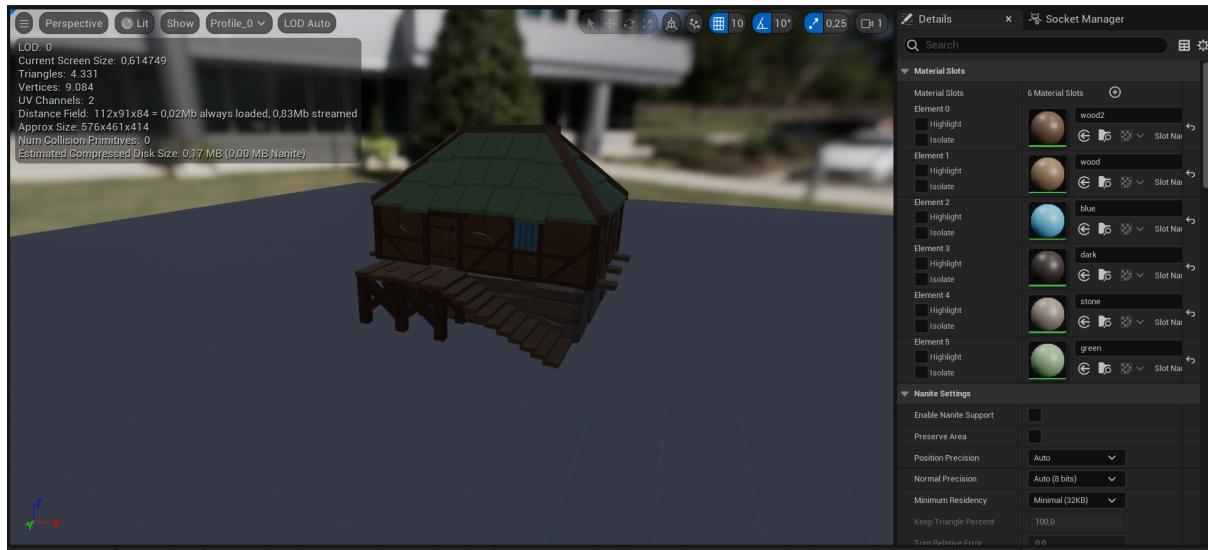




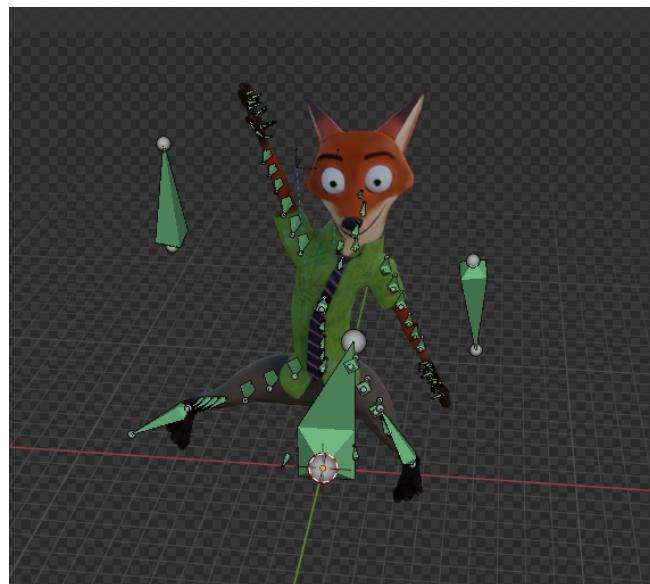
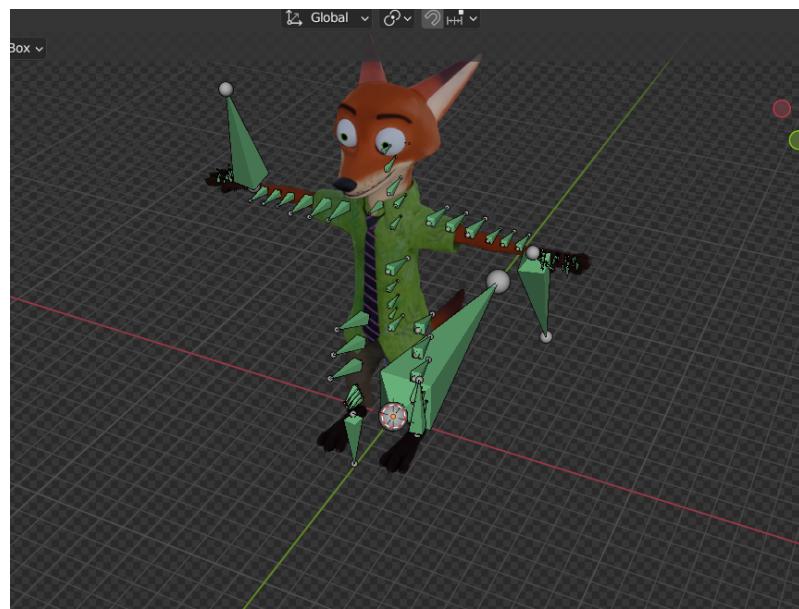
- Baú



- Casa



6.4 - Personagem



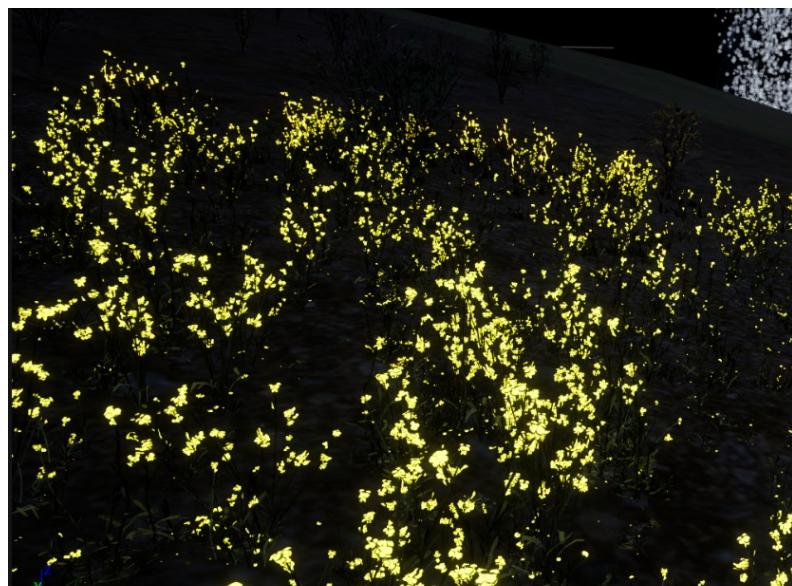


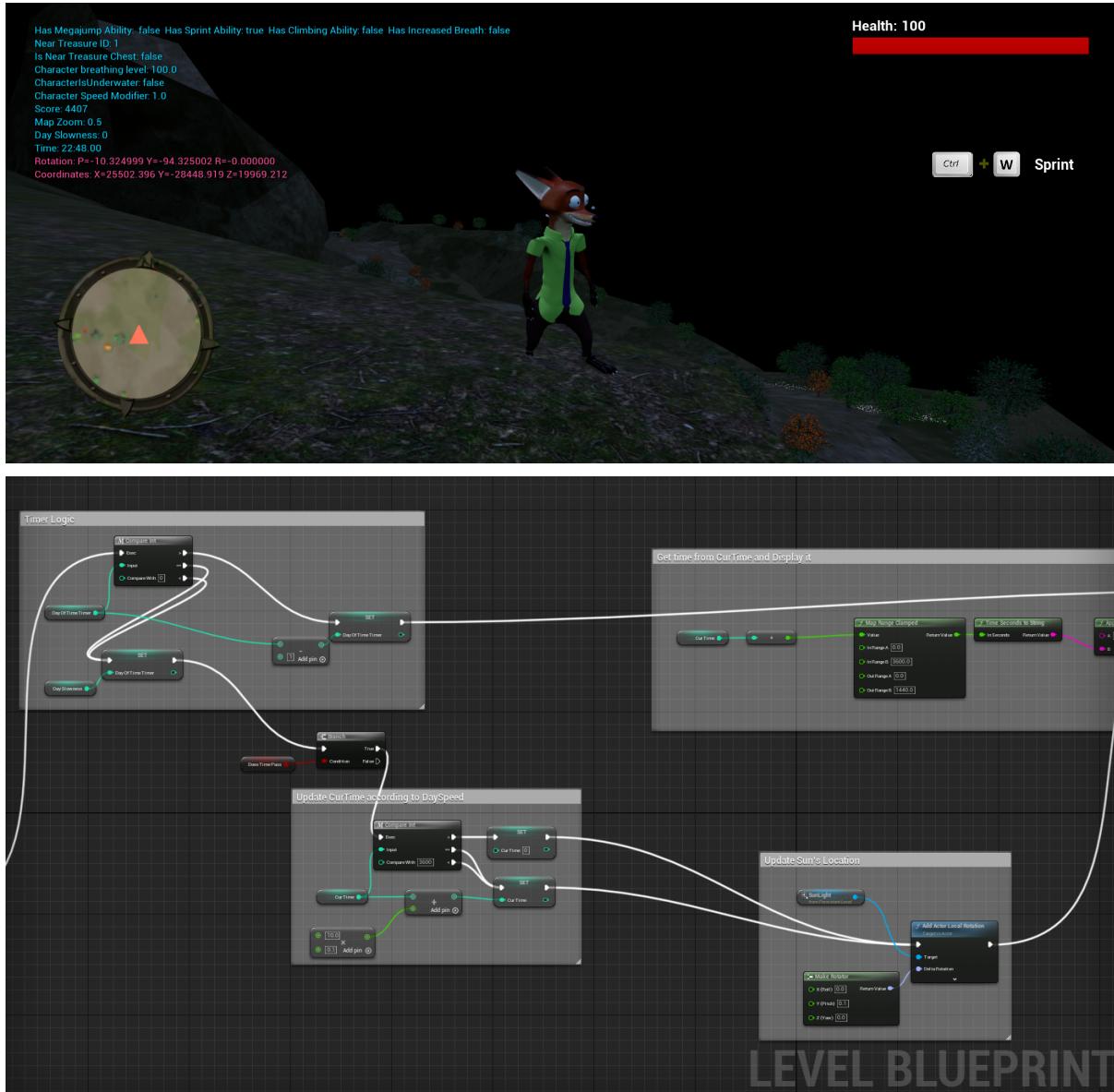
6.5 - Iluminação

Para a iluminação do mapa principal, foram utilizadas duas fontes de luz paralelas, uma para simular a iluminação do sol e outra, a iluminação da lua. A programação do ciclo dia e noite foi realizada do mesmo modo que o resto da lógica do jogo, ou seja, utilizando blueprints.

Além disso, destacam-se alguns materiais trabalhados para complementar a iluminação principal do jogo: a lava, a água e as flores. A lava e as flores são

materiais emissivos, ou seja, possuem iluminação própria, enquanto que a água reflete a iluminação de seu ambiente.



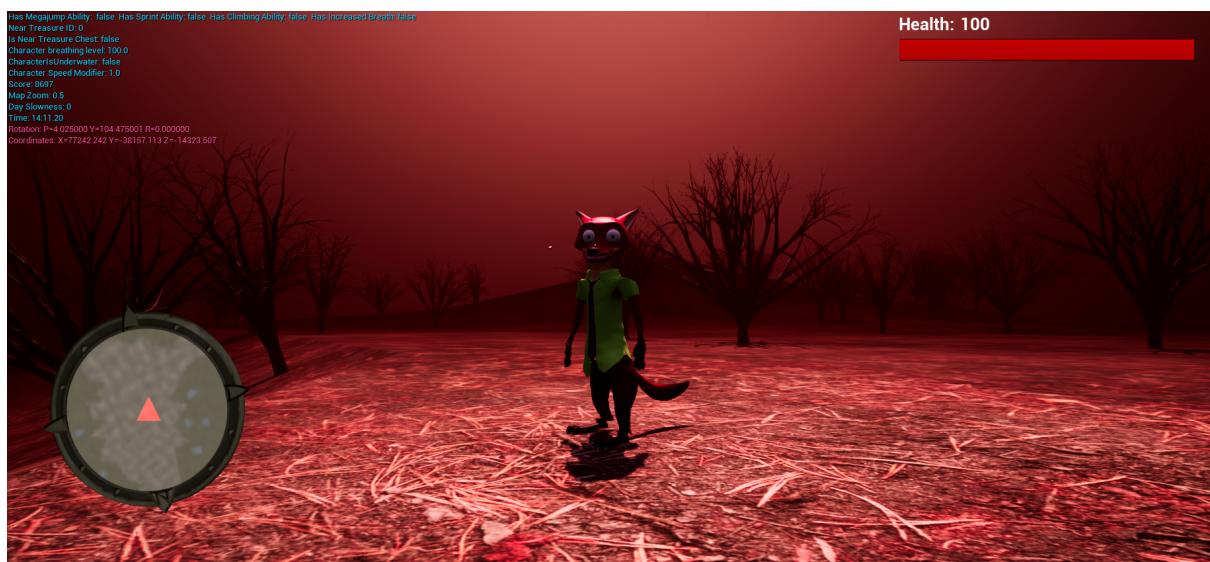


6.6 - Filtros, post-processing e partículas

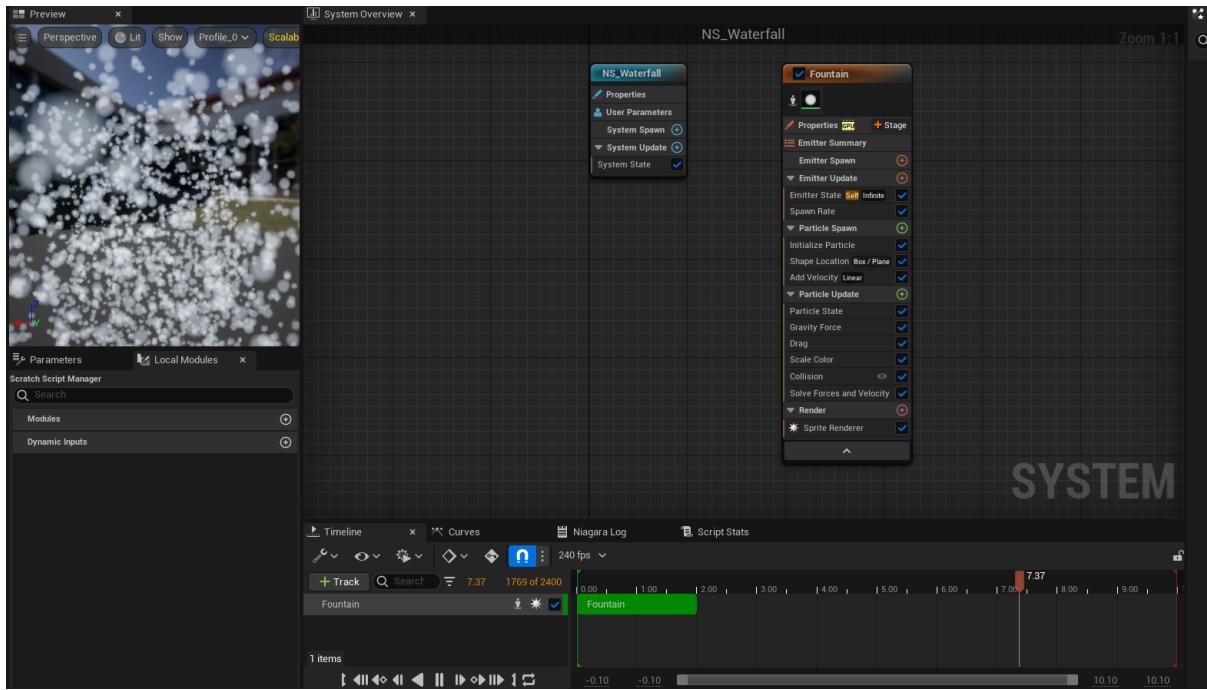
Para complementar a imersão do jogo, foram também utilizados diversos filtros e post-processings para simular cada região do jogo, bem como o ambiente subaquático. Os post-processings de cada região modificam algumas características da câmera como exposição, temperatura de cor, saturação e tinting. Já o post-processing subaquático utiliza um material próprio para adicionar o filtro observado durante o jogo, material este implementado dentro da Unreal Engine.

Além disso, o jogo também utiliza fog volumétrico diferenciado tanto na ilha de neve como na ilha vulcânica, o que contribui para a imersão. Ambos os fogs são ativados por triggers dentro do jogo.

Finalmente, para complementar os efeitos descritos, também foram criados sistemas de partículas próprios da Unreal Engine (Niagara Particle System), com o intuito de simular a água da cachoeira, a água que chega no lago e a neve. Os resultados são exibidos abaixo:



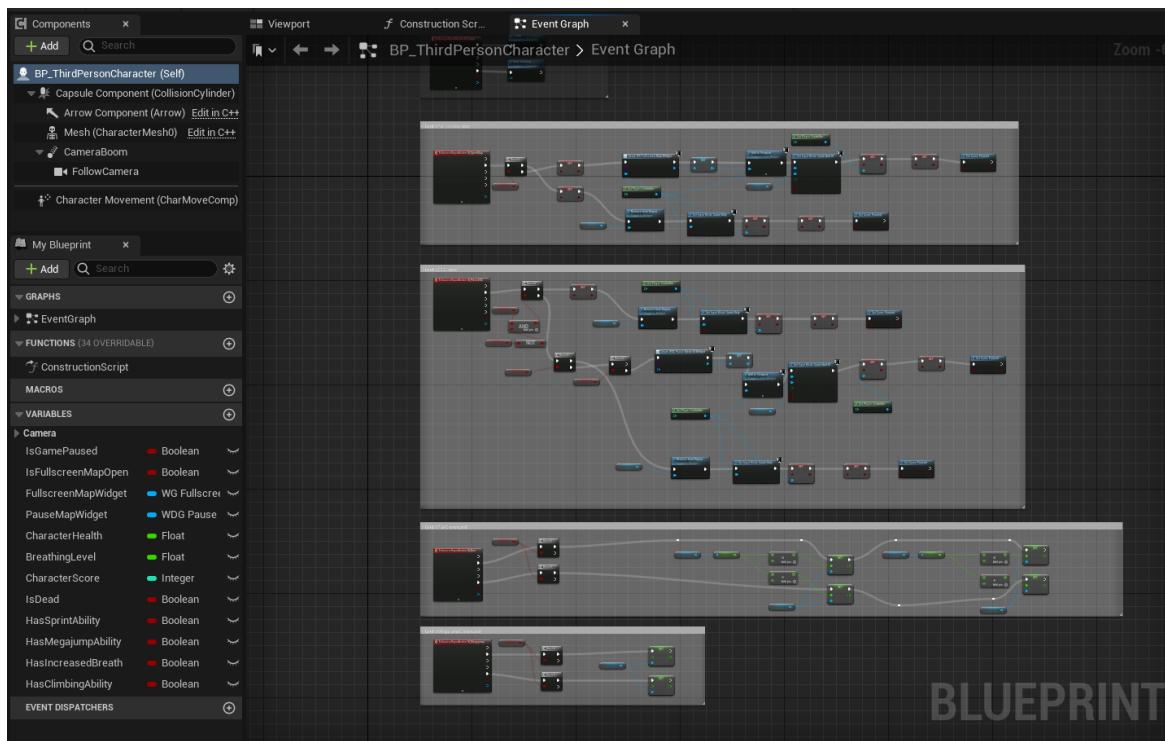




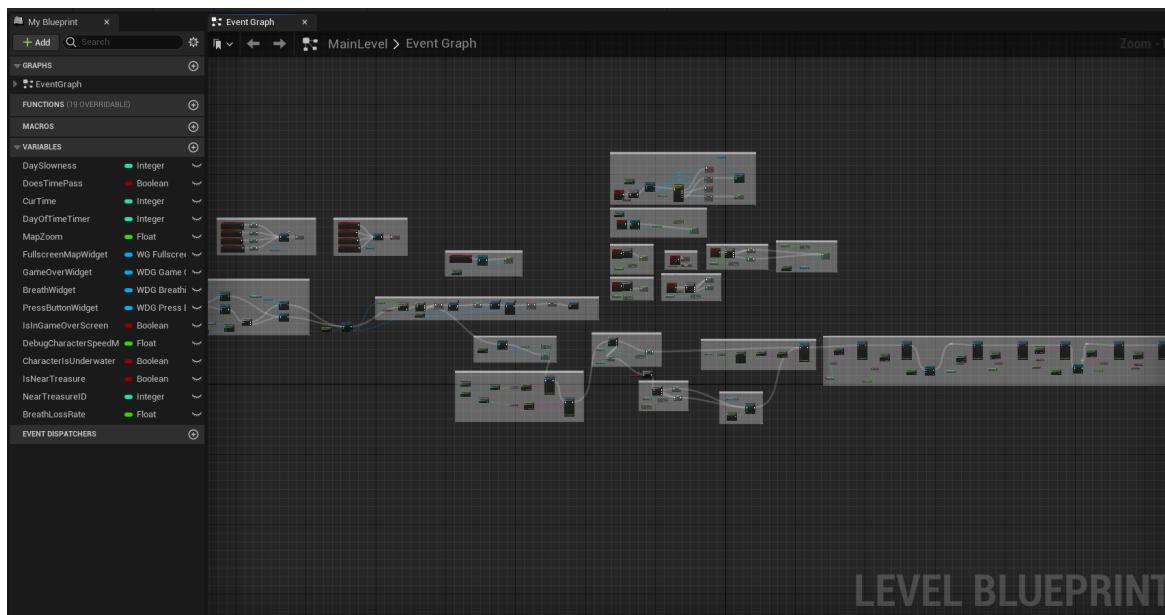
6.7 - Lógica de jogo e programação

Para a programação de toda a lógica de jogo, foi utilizada a linguagem nativa da Unreal Engine, as blueprints. Em suma, dado o pouco tempo disponível para o projeto, foram trabalhadas apenas algumas poucas classes, como a classe do próprio mapa principal e a classe responsável pelo controle do personagem em terceira pessoa. Além disso, também foram programados os elementos responsáveis pela UI, os widgets.

Assim, a partir do uso das blueprints, foi possível implementar o ciclo dia e noite, triggers para aparecimento do *fog* em áreas específicas, os níveis de vida e respiração do personagem, um sistema básico de pontuação e, finalmente, os poderes possíveis de serem adquiridos durante a gameplay. Também através das blueprints foram implementados os widgets dinâmicos como os controles de cada poder, que só aparecem após a obtenção do respectivo poder, e a barra de respiração, que só aparece quando o personagem está submerso.



Blueprint do personagem em terceira pessoa, mostrando suas variáveis



Blueprint do mapa principal, mostrando as variáveis utilizadas



6.8 - Otimizações

Uma vez que o jogo de mundo aberto possui um grande mapa e, levando em consideração que este jogo foi implementado utilizando apenas um grande nível, diversas formas de otimização foram necessárias com o intuito de viabilizar a quantidade de assets e detalhes necessários para a criação de cada ilha.

A otimização mais significativa e prevalente no jogo foi a utilização de diversos níveis de Level of Detail (LODs) para cada mesh criada (árvores grandes e pequenas e flores, principalmente, uma vez que estas existem aos milhares pelo

mapa). A existência de centenas de milhares de polígonos em cada árvore demanda um poder de processamento demasiado grande e, portanto, modelos mais simples e leves tiveram que ser criados para substituir os modelos mais detalhados quando a câmera não estiver tão perto deles. Estes LODs foram, então, integrados ao jogo a partir de funcionalidades já existentes na Unreal Engine que possibilita editar cada característica (tamanho e materiais, por exemplo) de cada LOD de cada mesh.

Outra otimização necessária de ser tomada foi a implementação de um controle para a Culling distance, ou Draw distance, isto é, a distância necessária para que o motor gráfico pare de renderizar certo assets com o objetivo de economizar memória e processamento (os assets mais afetados foram as árvores, que demandavam muito processamento quando a câmera pegava uma porção muito grande das ilhas ao mesmo tempo).

7 - Objetivos Atingidos

Grande parte dos objetivos estabelecidos no início do desenvolvimento do projeto foi atingida. Tanto a implementação de filtros específicos para cada região, como o trabalho com partículas para neve foram executados, bem como o uso de materiais emissivos e reflexivos, o trabalho com materiais de água e lava e a implementação de diversos tipos de vegetação.

Alguns objetivos, no entanto, com o curto prazo do projeto, tiveram que ser cortados, como a exploração de iluminação interior de cavernas, a implementação de animação e partículas de fogo, e a implementação de chuva. Vários assets inicialmente desejados também inevitavelmente foram descartados, como chalés para a região de neve e a existência de mobs agressivos.

Em conclusão, muitos aspectos da área de processamento gráfico foram explorados e aprendidos, sempre integrados ao ambiente e processo de desenvolvimento de jogos, ambos andando lado a lado para fornecer ao usuário final um mundo criativo, expressivo e imersivo, que possibilitasse a expansão dos limites da criatividade e da interpretação artística, ao mesmo tempo proporcionando uma experiência de usuário simples, prática e prazerosa.

8 - Possíveis Melhorias

Entre as possíveis melhorias que poderiam ser aplicadas futuramente para completar o jogo, podemos citar a implementação de um sistema de combate com elaboração de inimigos para dificultar a exploração, um sistema de níveis para que houvesse um sentimento de progressão por parte do usuário enquanto joga e o aumento do mapa com possíveis novas ilhas para aprimorar o cenário. Além disso, seria possível introduzir uma história que trouxesse sentido ao personagem e complementasse o jogo ao apresentar uma motivação para o jogador.

Além disso, o mapa principal poderia ter sido dividido em mais níveis, o que daria um desempenho muito melhor para o jogo. Manter o jogo com um único level iria inviabilizar a escalabilidade do jogo.

8 - Referências

CG GEEK. Low Poly Island | Beginner | Blender 2.8 Tutorial. Disponível em: <<https://www.youtube.com/watch?v=0lj643VmTsg>>. Acesso em: 4 jul. 2023.

CG GEEK. How to Create a Low Poly Tree in 1 Minute. Disponível em: <<https://www.youtube.com/watch?v=y7PdiGXbrD0>>. Acesso em: 4 jul. 2023.

Configurando a colisão para uma malha estática. Epic Developer Community. Disponível em: <<https://dev.epicgames.com/documentation/pt-br/uefn/configuring-collision-for-a-static-mesh-in-unreal-editor-for-fortnite>>. Acesso em: 4 jul. 2023.

Unreal Engine 5 Documentation. Unrealengine.com. Disponível em: <<https://docs.unrealengine.com/5.0/en-US/>>. Acesso em: 4 jul. 2023.

Creating and Using LODs. Unrealengine.com. Disponível em: <<https://docs.unrealengine.com/5.0/en-US/creating-and-using-lods-in-unreal-engine/>>. Acesso em: 6 jul. 2023.

9 - Links

- Link para o repositório do GitHub:

<https://github.com/VanderSant/PCS3539-ComputacaoGrafica>

- Link para o Drive (com o vídeo, relatório e apresentação na pasta "Entrega Final"):

https://drive.google.com/drive/folders/1AjYYSnLEHa2uDUPZ_bS2ekL4aZHgk6g?usp=sharing

- Vídeo da demonstração:

 Hopper Island - PCS3539 - Demonstração