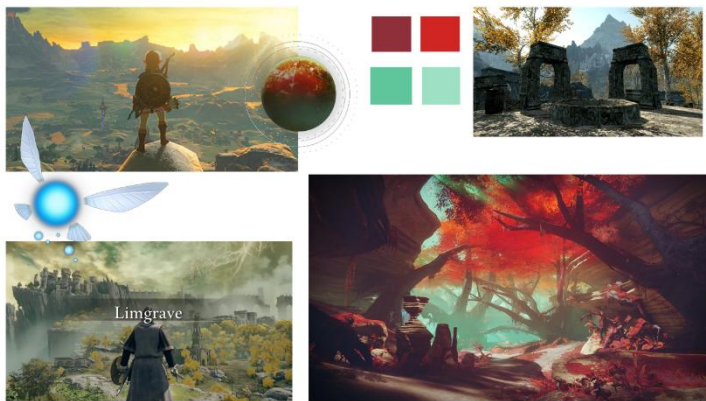


MPIE REPORT

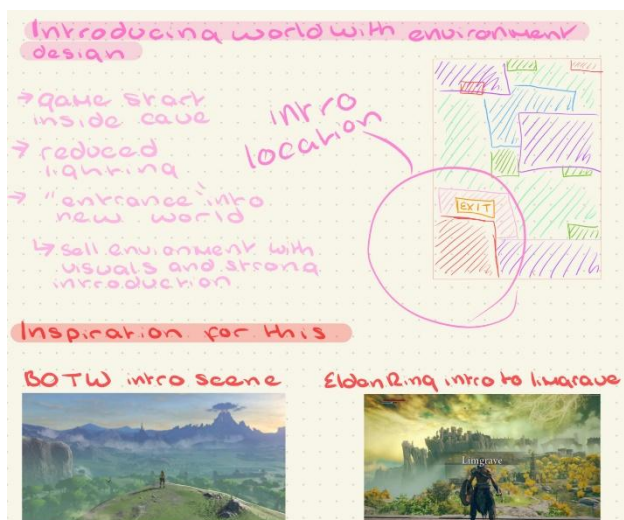
Background

For my project, I aimed to create a walkthrough story environment, using collectable items to bring focus to specific areas in the world space, hoping to tell a story through visual means rather than intricate game mechanics. Drawing from games I enjoyed, I created a mood board to collect my ideas and inspirations.

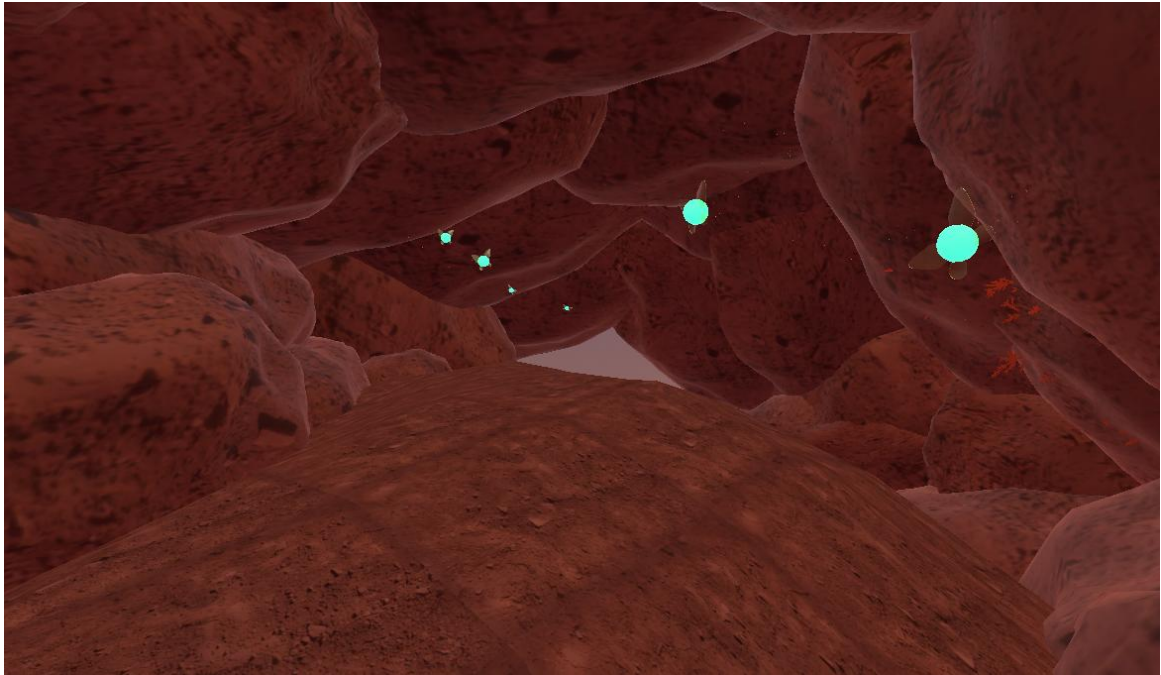


1 Mood board

I considered how the world was presented to the player when gathering reference images to draw from in my project. I was particularly drawn to the use of warm reds and oranges for vegetation and plant life, and contrasting this against the green colours. Each of the experiences I drew from had cinematic quality in establishing the environment. This was something I planned to replicate.



2 Introduction plan



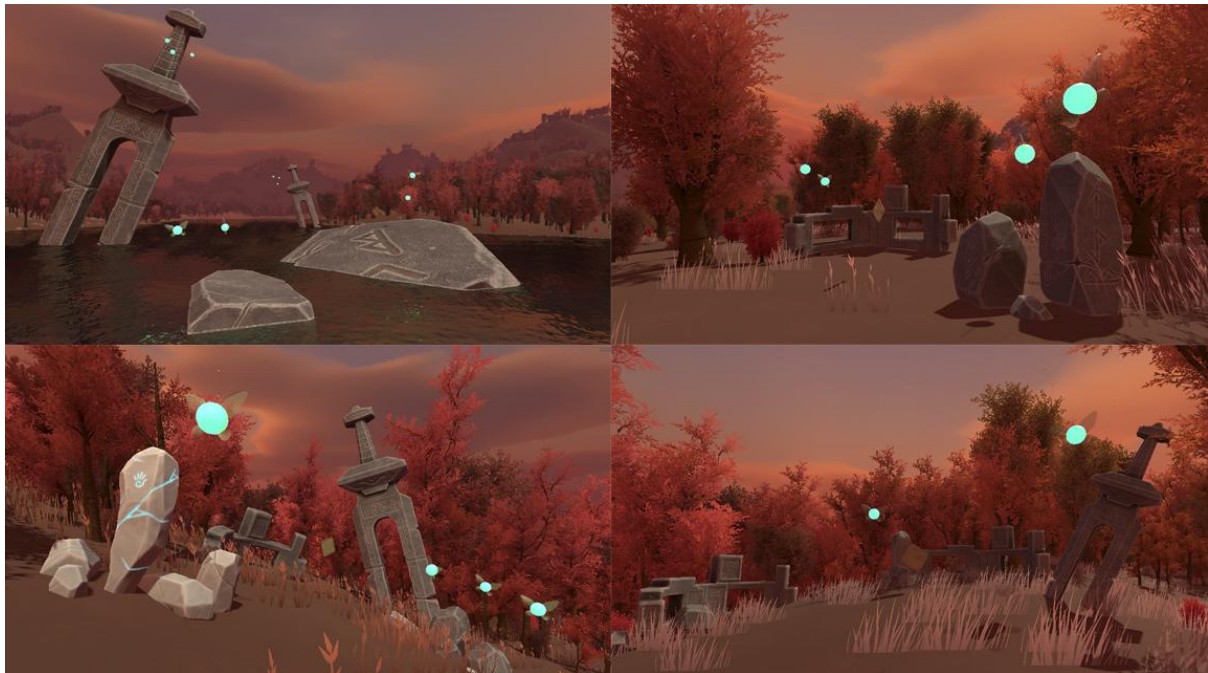
4 Start view

Players can follow this trail to the exit and see their first impression of the world. They can then collect the first item.



5 View from cave exit

The player can continue to follow the paths lit up by the fairies to collect four page items from different parts of the map and refill the diary they collected.

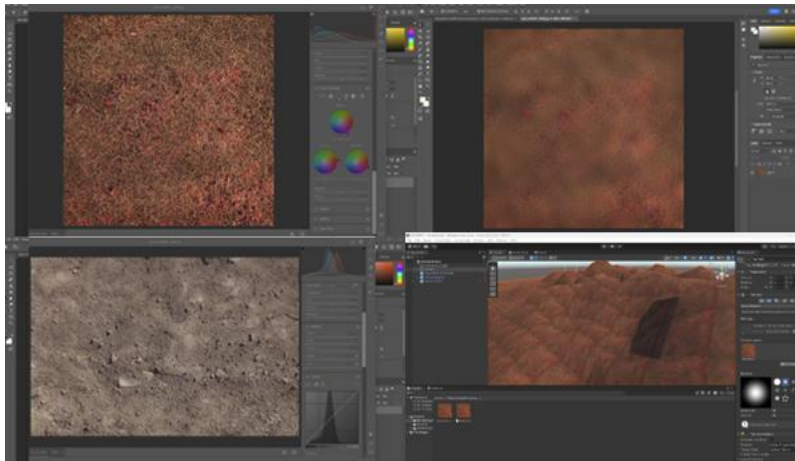


6 Different item locations

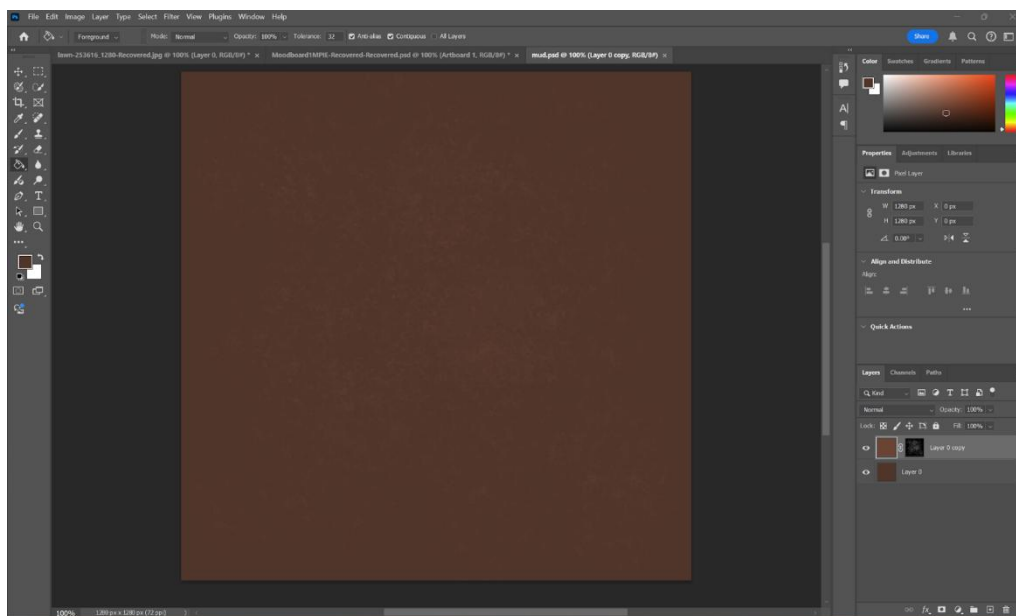
The objective is complete when the player has collected all of the items in the world space.

Description of Technical Implementation

I began by importing the Starter Asset – FPS (Unity Technologies, 2024) . I created a terrain game object and used the Unity terrain editor to sculpt the terrain. Once I was pleased with the terrain shape, I gathered images to edit in Photoshop, changing the colour values to fit the red and pink tones I wanted to achieve.

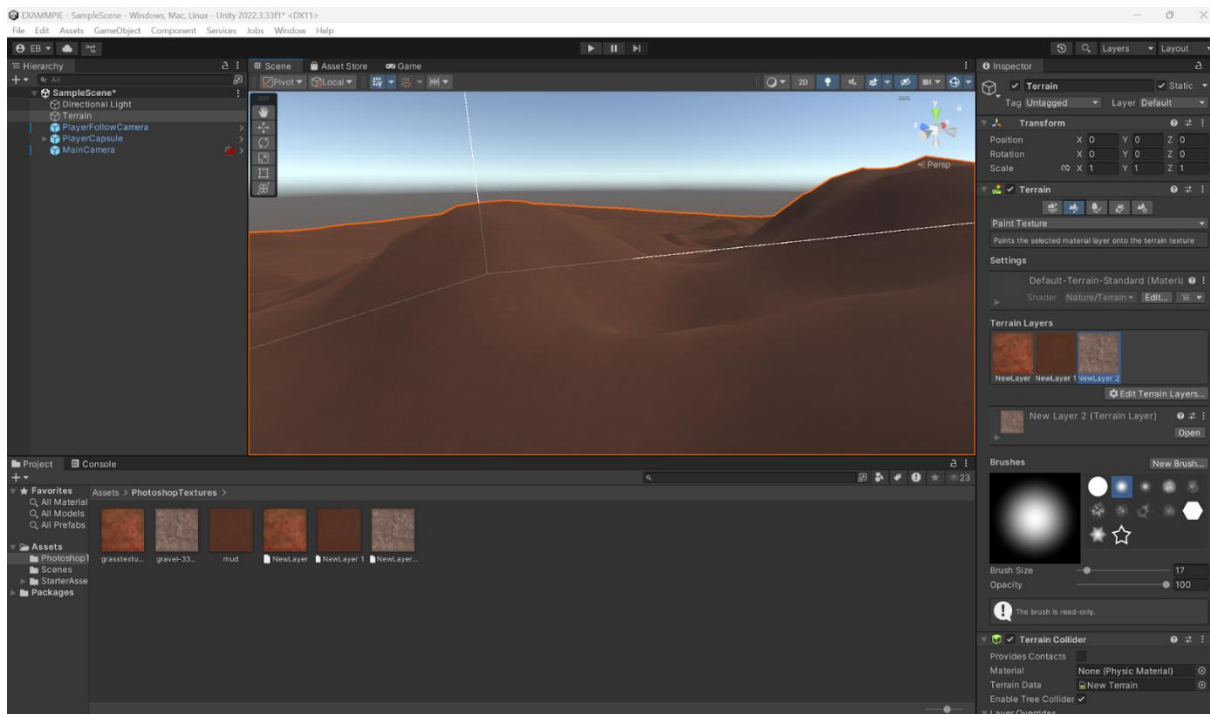


7 different textures in Photoshop and usage on terrain

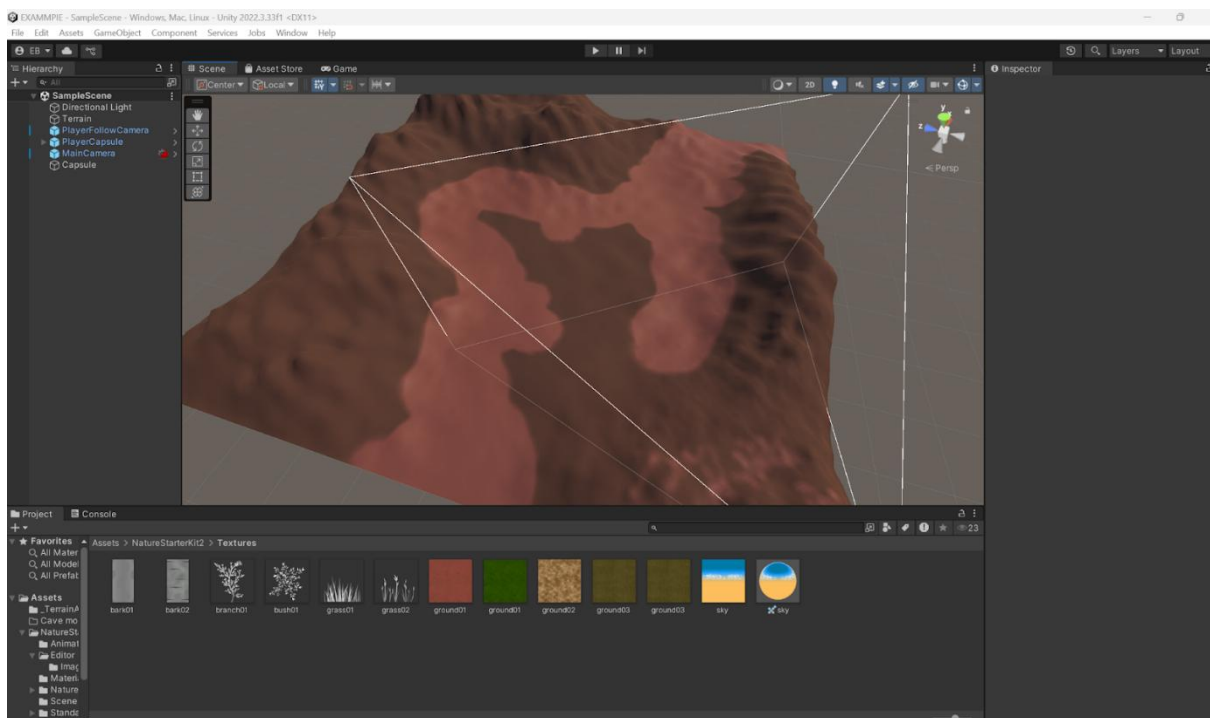


8 Dirt texture using layer mask

I created ground textures using layer masks to add subtle details and hints of colour on the surface for depth. These were used in combination with floor textures from the NatureStarterKit2 assets (Shapes, 2016) to create the ground texture.

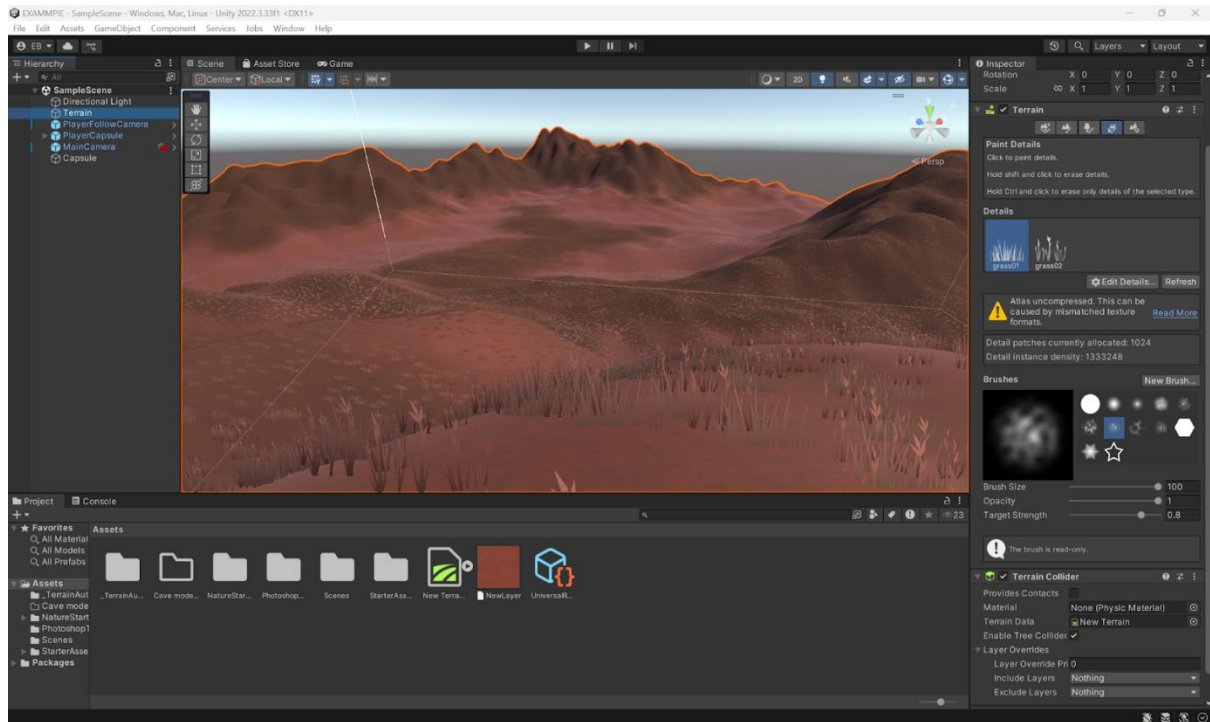


9 Dirt texture view on terrain



10 Using different textures

I added two NatureStarterKit2 (Shapes, 2016) grass textures and painted them onto the terrain. I chose two different colours for the grass directly from the terrain textures using the colour picker to keep consistency in my environment.

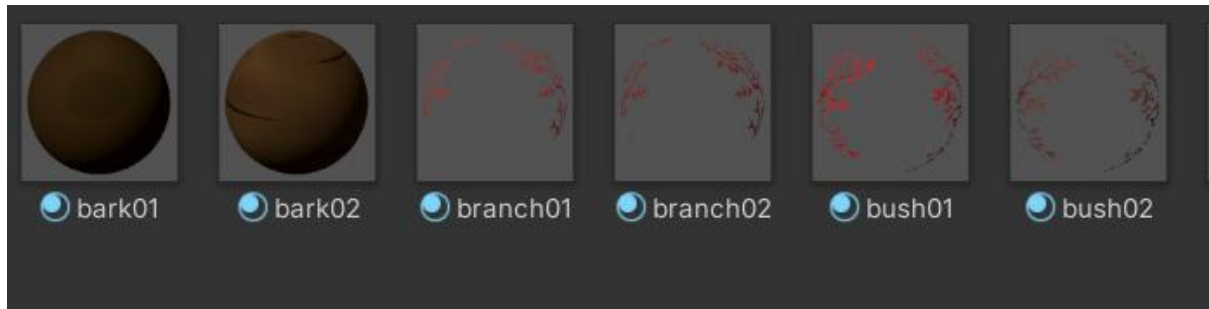


11 Terrain view with grass

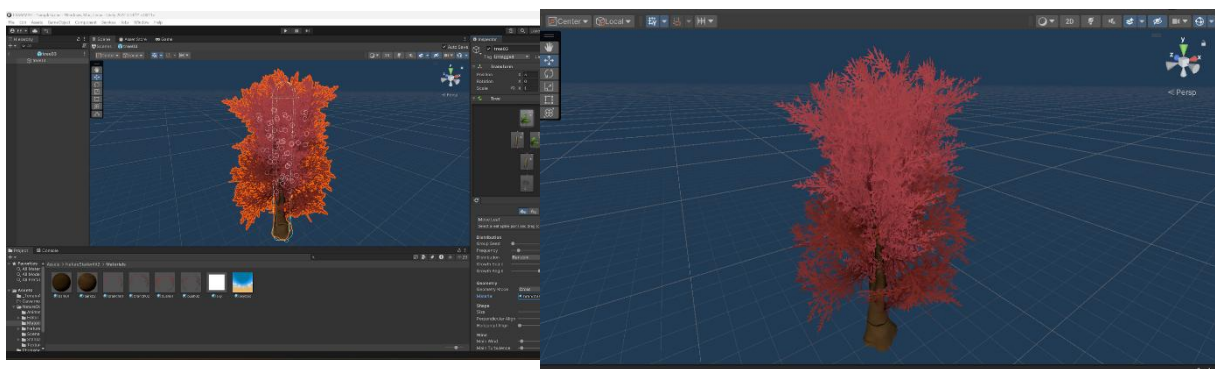


12 Grass settings

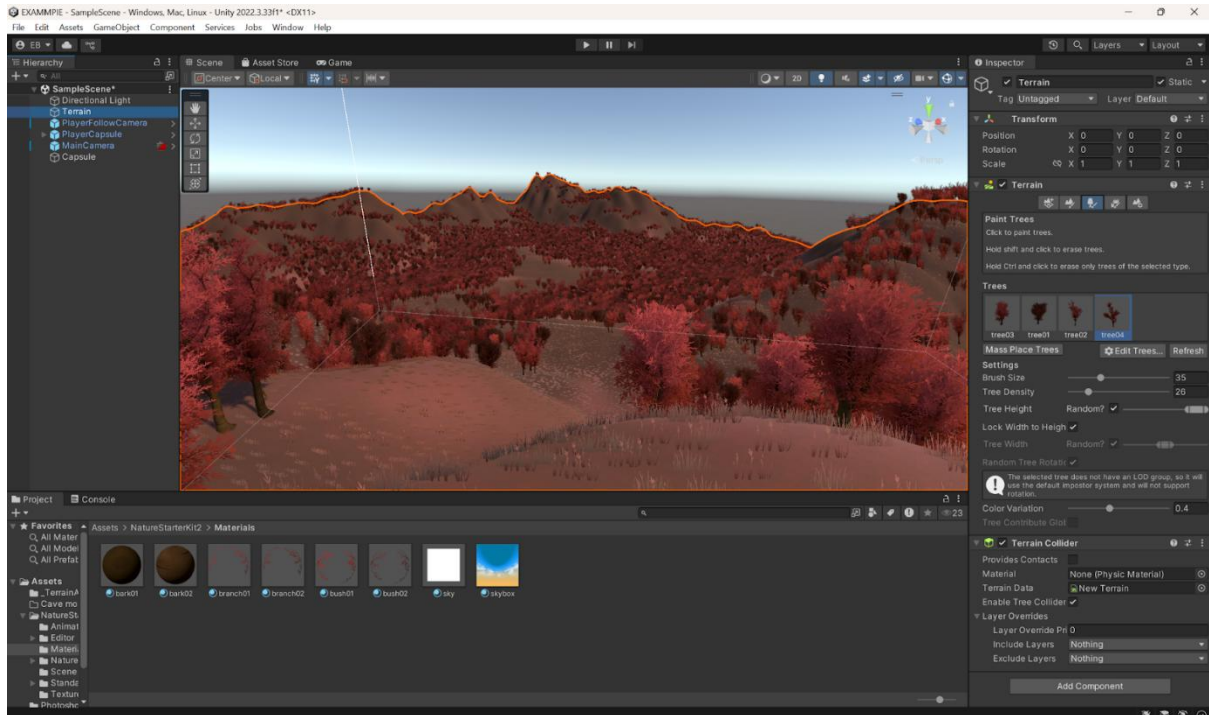
Using the paint trees tool, I created a dense forest. I used all available NatureStarterKit2 (Shapes, 2016) models to create variation in the plant life. I edited the branch materials that came assigned to the different trees and changed the colours to match my environment.



13 Changed to asset materials

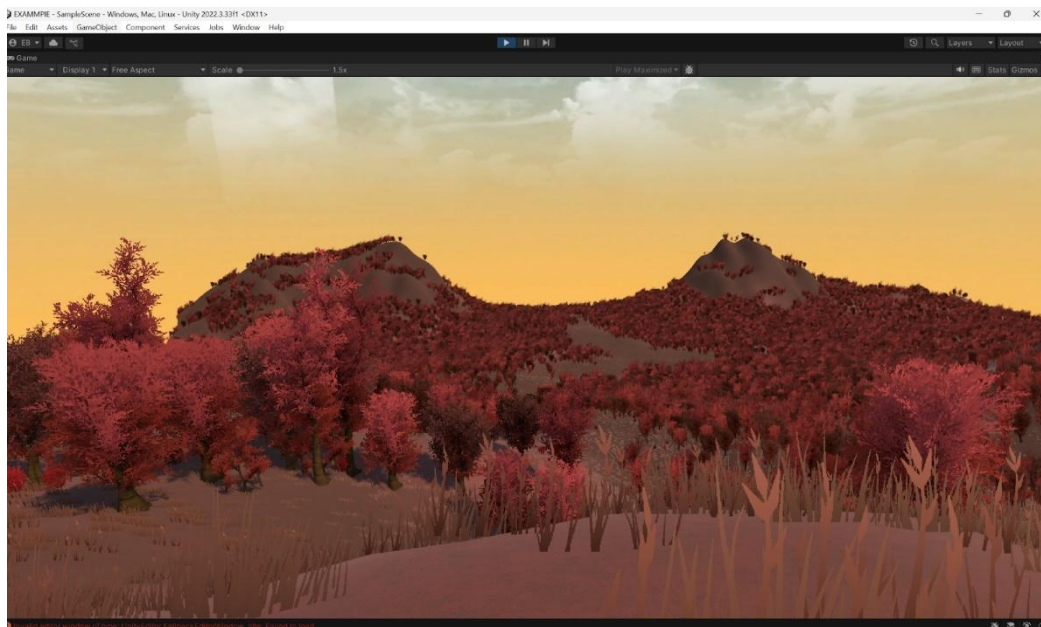


14 Applied changed to asset



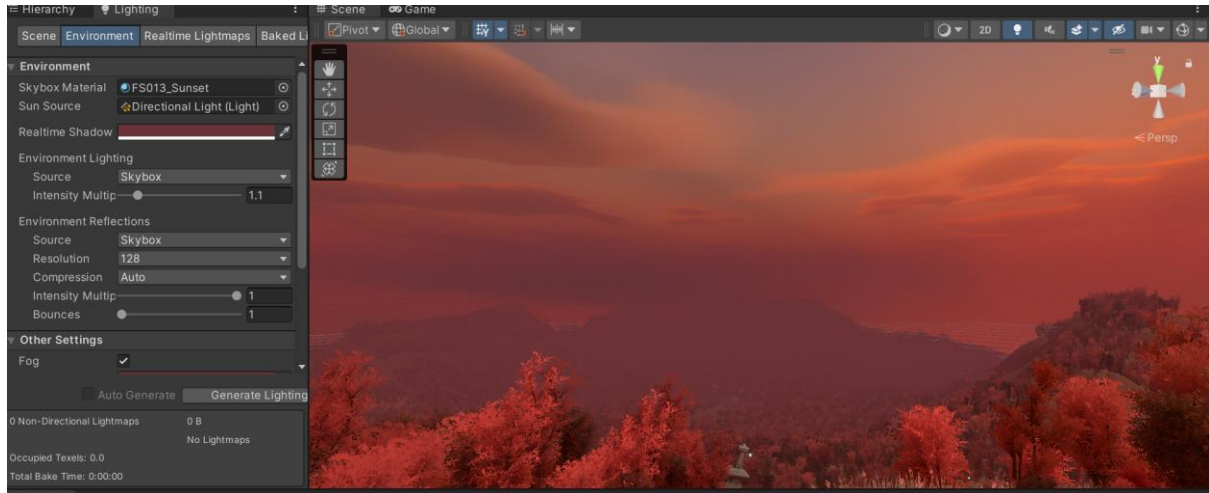
15 Forest environment

I erased some of the vegetation to create a clear path. To further enhance my scene, I edited the skybox. Originally, I had used the NatureStarterKit2 (Shapes, 2016) skybox but didn't feel the colours fit with the environment I had planned.



16 NatureStarterKit2 (Shapes, 2016) skybox

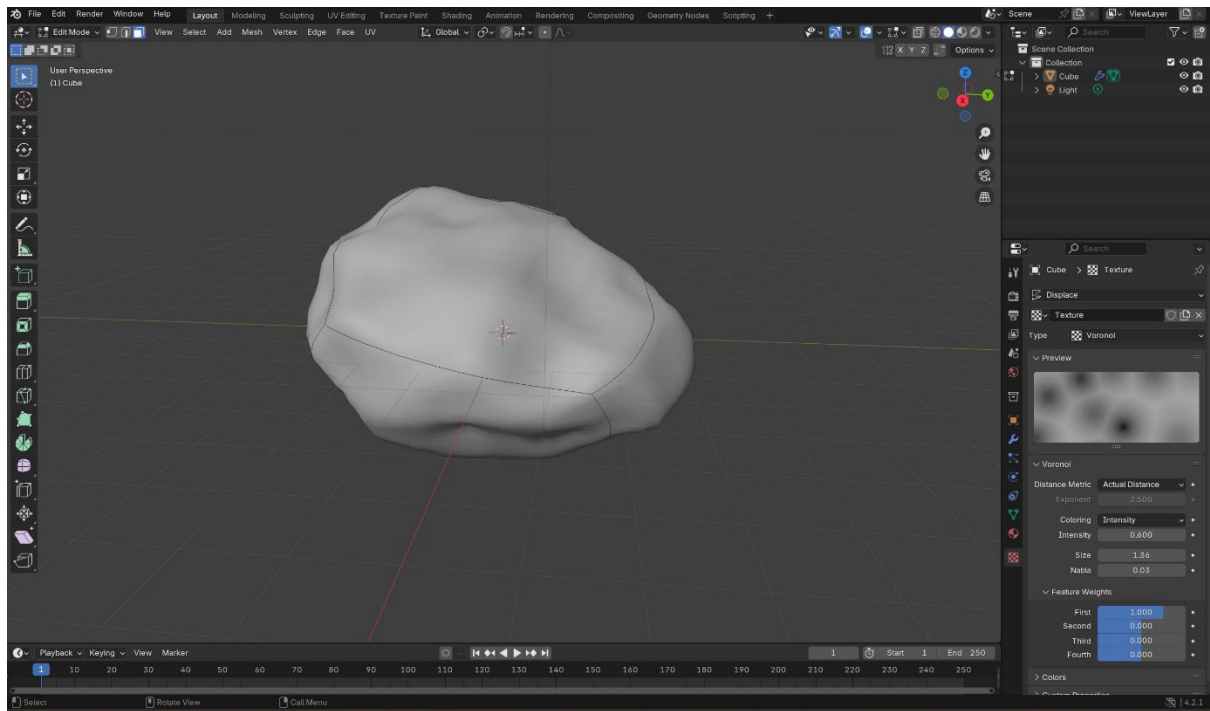
I assigned the FS013_Sunset material from Fantasy SkyBox Free (Render Knight, 2024) to my environment. I included fog in the lighting settings and matched this to the darker purple tones in the new skybox.



17 Fantasy SkyBox Free (Render Knight, 2024) skybox

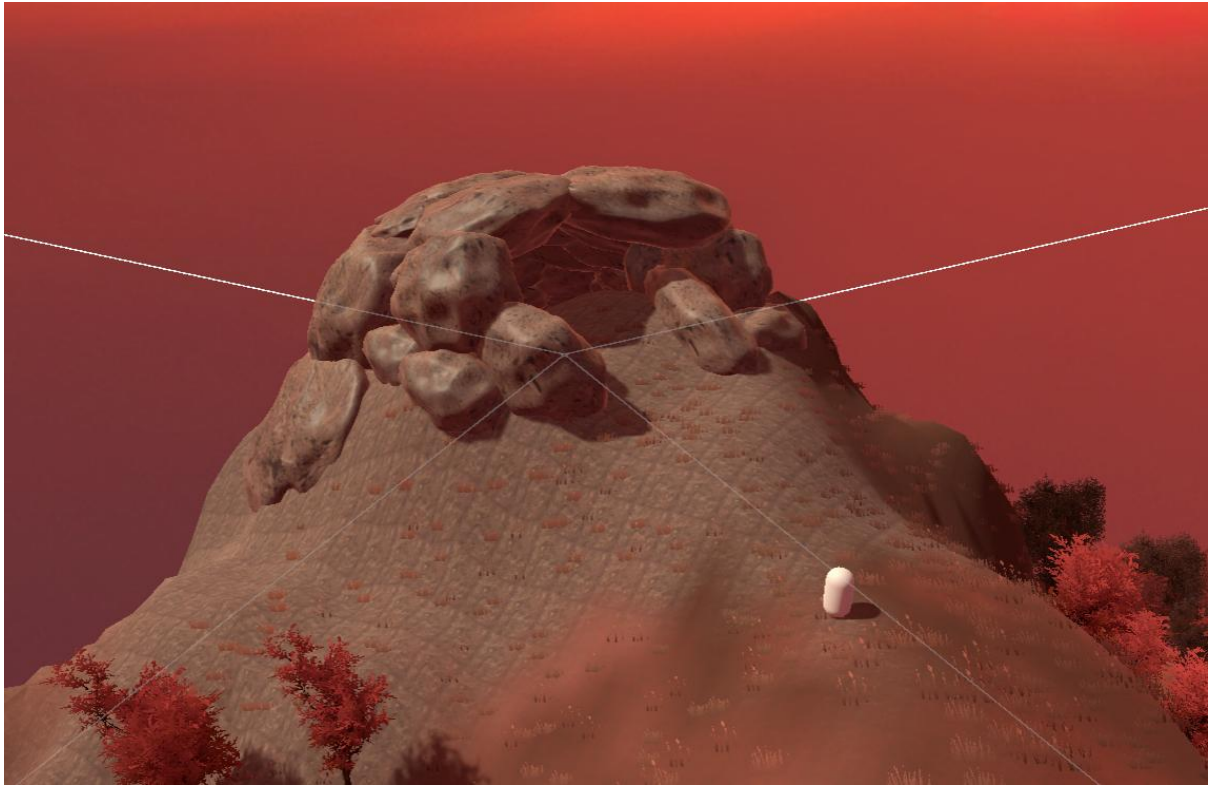
Then I added a capsule game object at the location in which I intended the player to get their first impression of the wider world space. From this perspective, I was able to edit the render distance of objects in the terrain editor menu and also edit the fog details to see clearly how well this affected my view of the environment.

To build the cave, I created a rock model in blender and imported this to my unity project.



18 Rock model

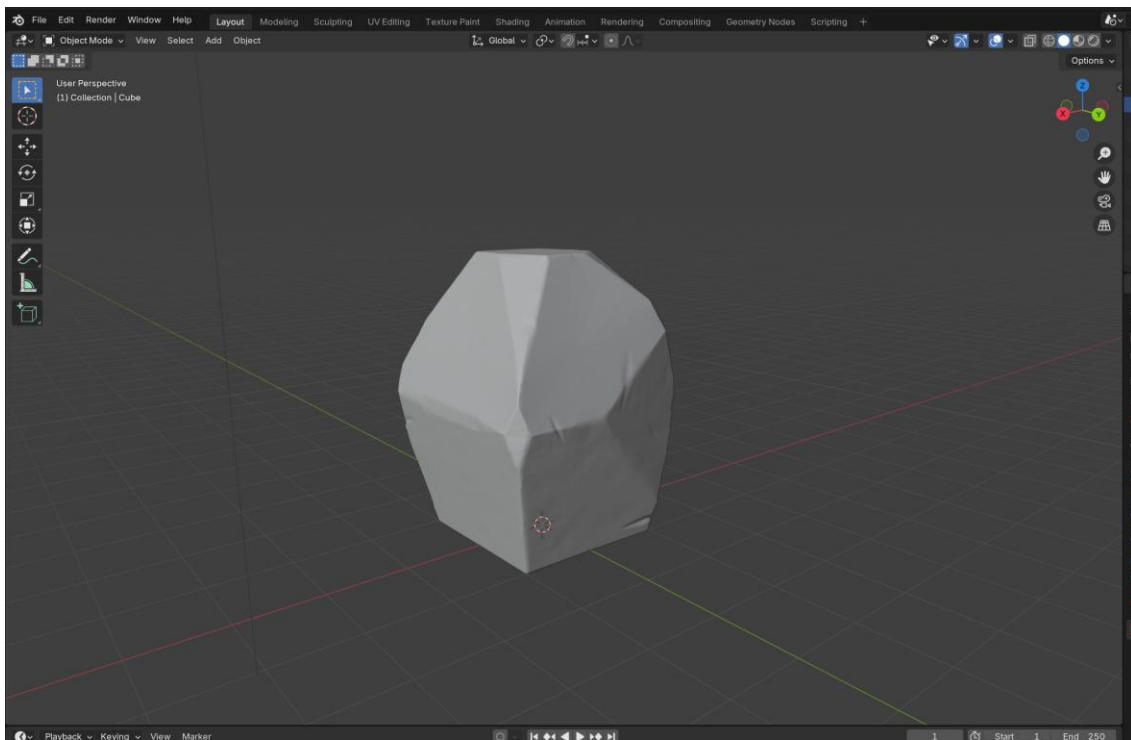
I used this rock model repeatedly to sculpt a cave environment where my player will be spawned in the beginning, manipulating the object's transforms within the inspector to achieve the best shape for my cave.



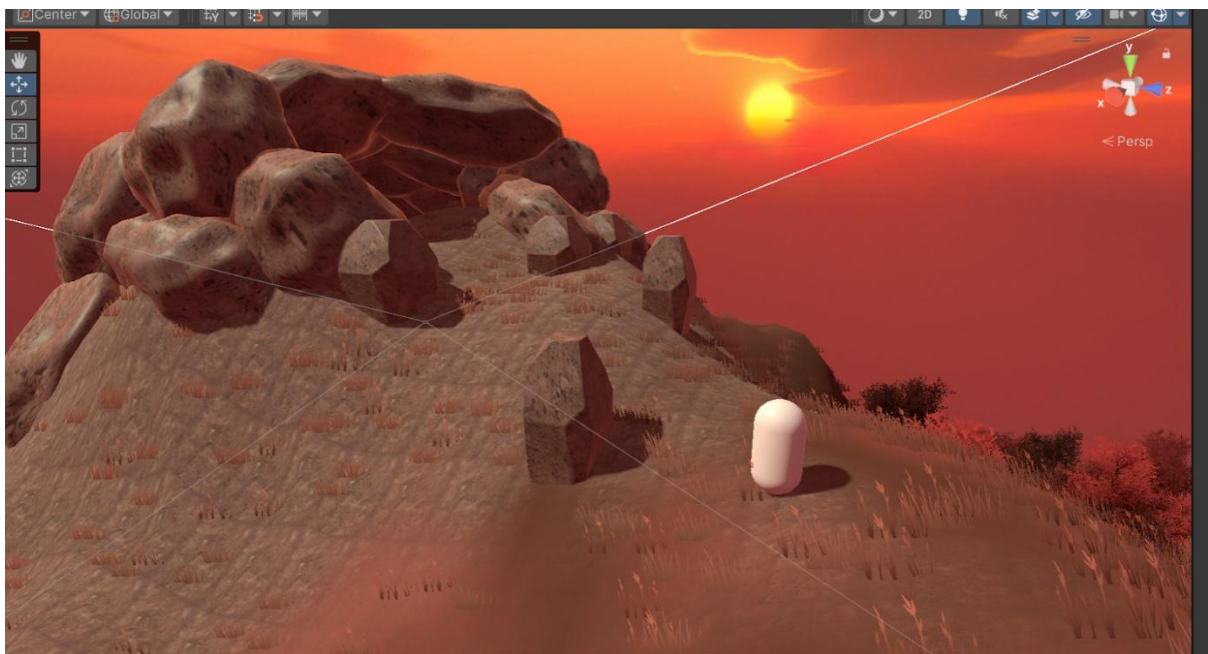
19 Finished cave

In Unity, I applied the gravel texture I had created previously to these game objects and then added box collider components to prevent the player from walking through any of the cave walls. I had originally attempted this with the rigid body component on my player capsule in the fps controller and using box colliders on the rocks, however this was unsuccessful. When I removed the rigid body from my player, this issue was solved.

The cave structure from the exit didn't look how I had hoped, To resolve this, I returned to blender to create another rock object with sharper edges, placing these in the world space around the cave exit.



20 New rock model

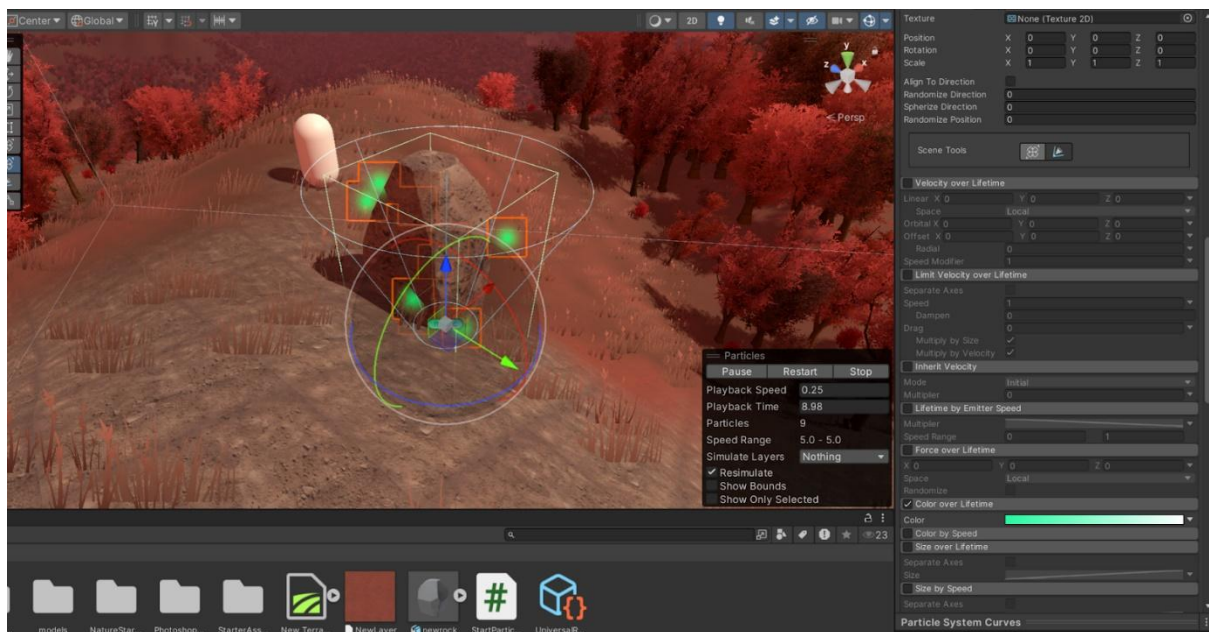


21 Improved Cave

The contrast between the smoother and sharper rock shapes helped improve the cave's overall design. For consistency purposes, the gravel texture I created was also applied to these objects. The scale and position of the completed cave made it clearer to distinguish

the path I wanted to mark out for the player and I began to consider how best to represent this with.

I planned to use the rocks from the cave edit but they didn't stand out in the environment so I decided to add fairies to specific areas. To do this, I used the particle effect system. In using this system however, I was aware it's computationally expensive and not suited to be on constant loop and repeated throughout the environment.



22 Particle Effect

I experimented with scripting a collision system with the particles where they would generate if the player reached a distance from the intended path. This used a box collider on the target object where the particles would spawn from, and wrote a script which I applied to the player game object. To manipulate the particle systems inside the script, I attempted to use the `ParticleSystem.EmissionModule.enabled` Documentation (Unity Technologies, n.d.) as guide. When this failed, I tried the `Play()` method on my particle game object variable. With little success, I then attempted to use the `SetActive()` function to reveal particles after collision.

```

private void OnTriggerEnter(Collider other) {
    //check it is the particle emission collider
    if(other.gameObject.name == "magic") {
        Console.WriteLine("particles");
        //play particle sequence

        //following unity documentation

        /**
        * code written following Unity Documentation ParticleSystem.emission
        * Available at : https://docs.unity3d.com/6000.0/Documentation/ScriptReference/ParticleSystem-emission.html
        */
        var emission = magicParticles.emission;
        /**
        * code written following Unity Documentation ParticleSystem.EmissionModule.enabled
        * Available at: https://docs.unity3d.com/6000.0/Documentation/ScriptReference/ParticleSystem.EmissionModule-enabled.html
        */
        emission.enabled = true;

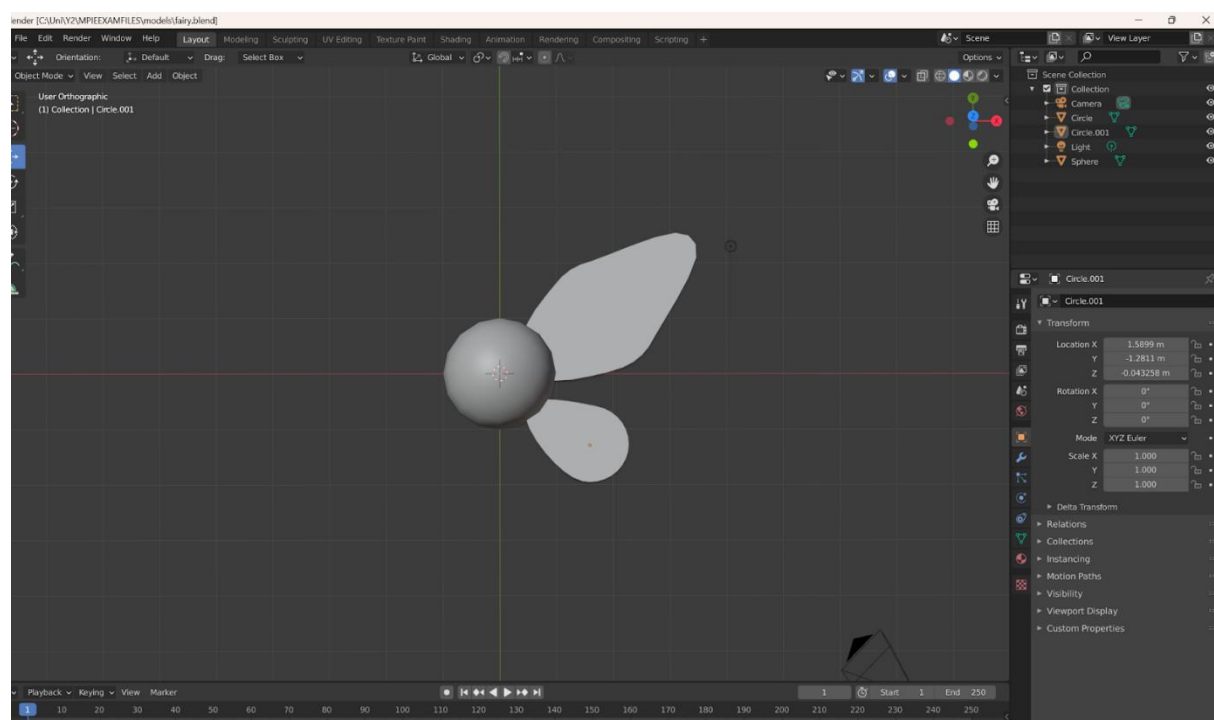
        //trying play method instead to see if it will plat particle sequence
        magicParticles.Play();
    }

    if(other.gameObject.name == "magicrock") {
        magicParticles.gameObject.SetActive(true);
        // magicParticles.Play();
        // psCollide = true;
        // Console.WriteLine("collided");
    }
}

```

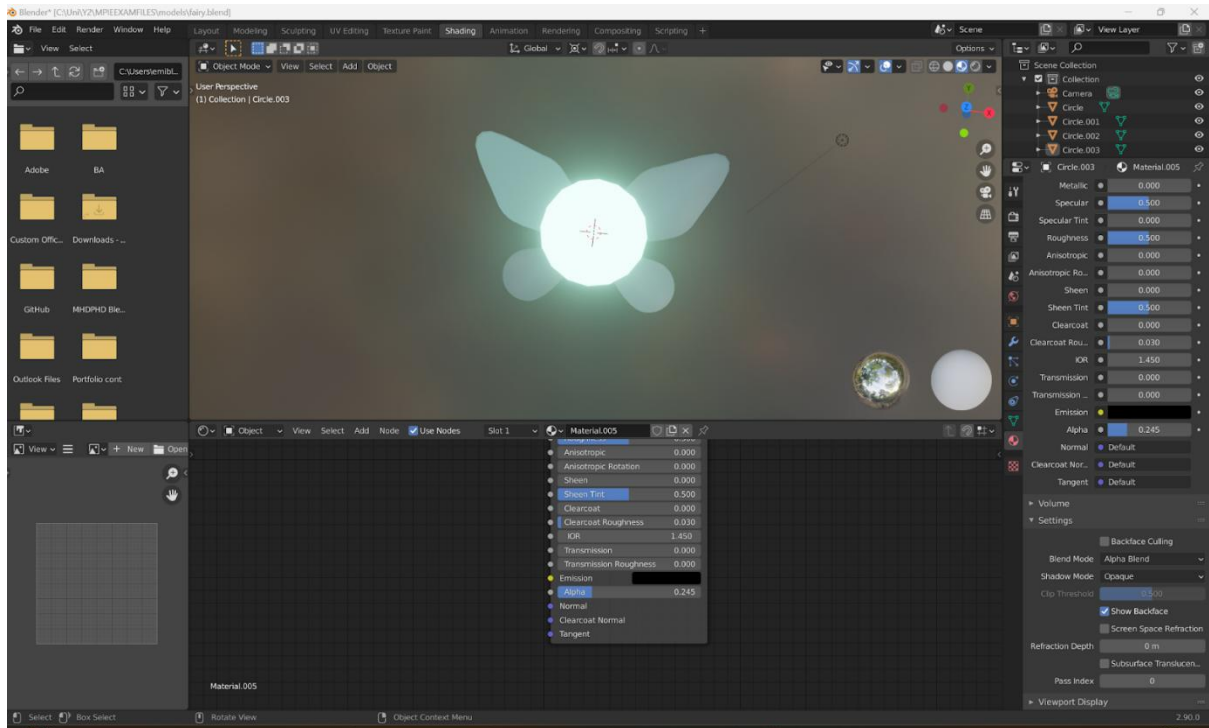
23 Particle management script

To solve this problem, I decided to use 3D models and animate them instead to represent the fairy NPCs. Using the fairies from Nintendo (1998) The Legend Of Zelda: Ocarina of Time as inspiration, I modelled the NPCs in blender.



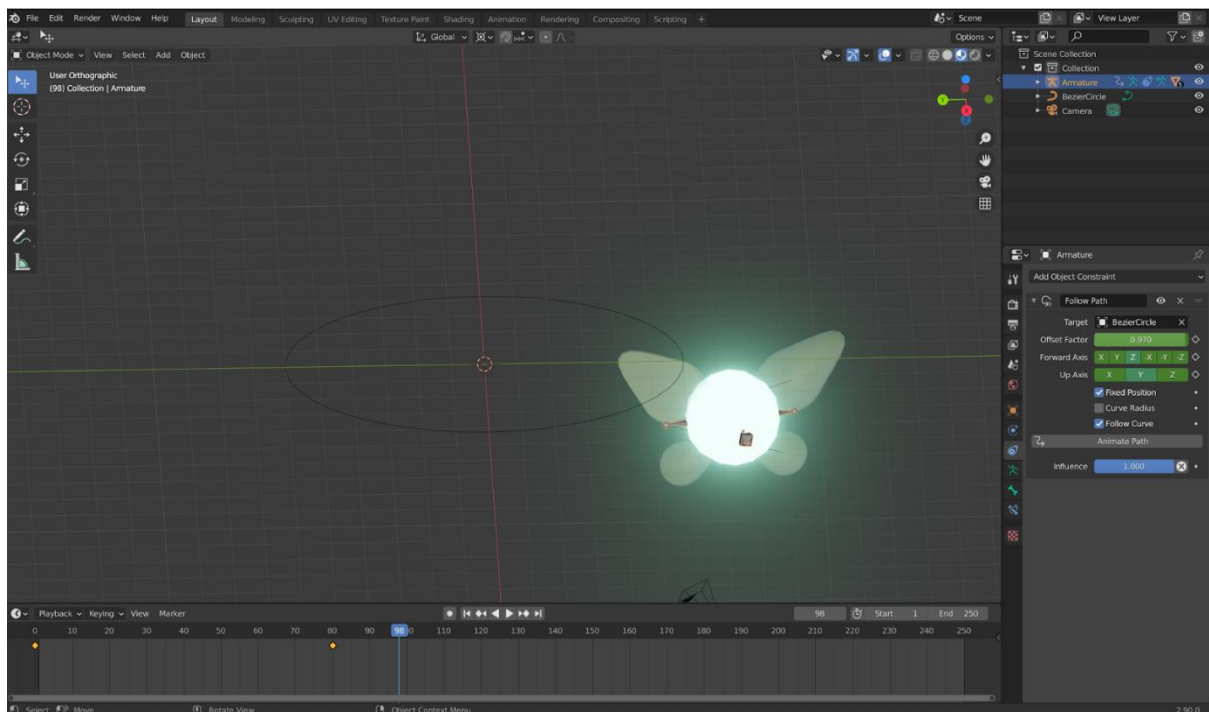
24 Start of fairy model

I used a sphere for the main body component as it was the closest shape to the particle emissions and then added additional circles, transforming them to shape the wings.



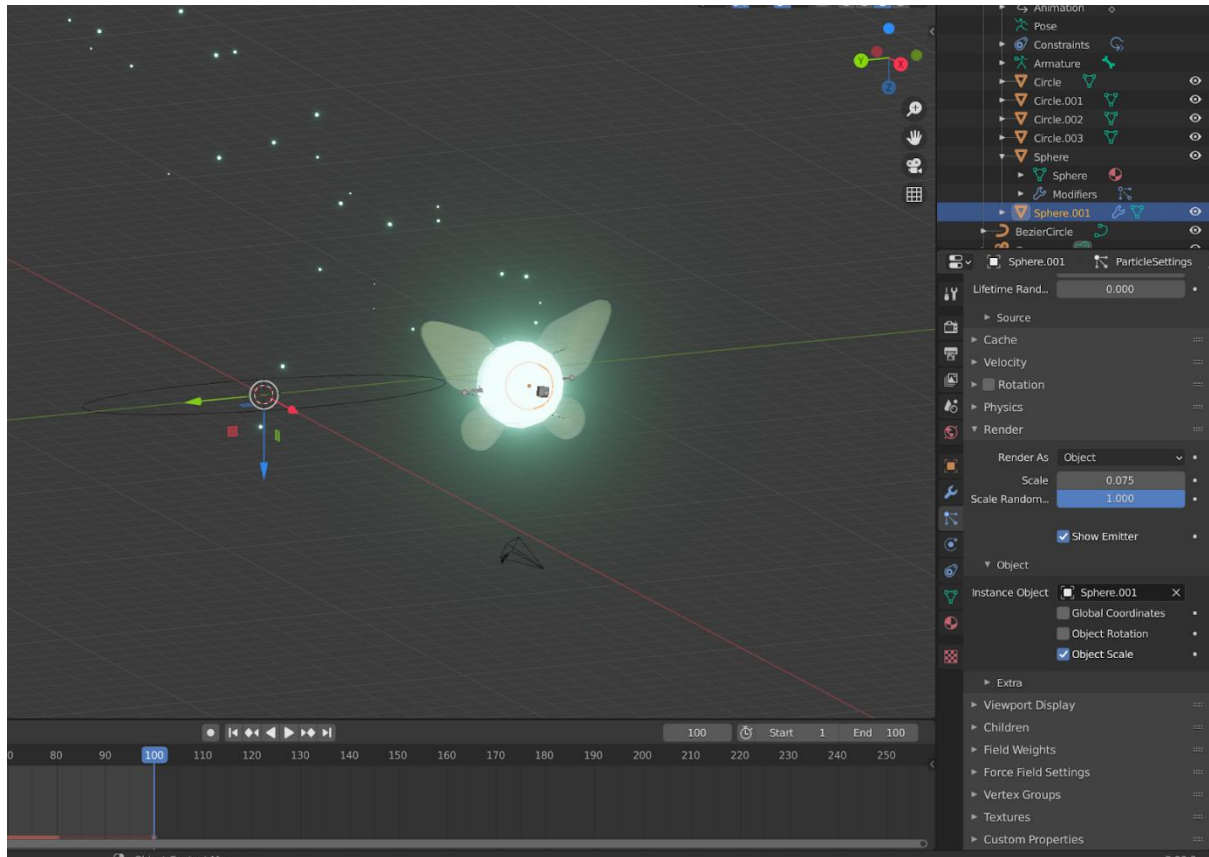
25 Adding materials

I created a material for my fairy in blender and enabled the bloom setting to create a glow effect. I added a separate material to the wings and changed the transparency.



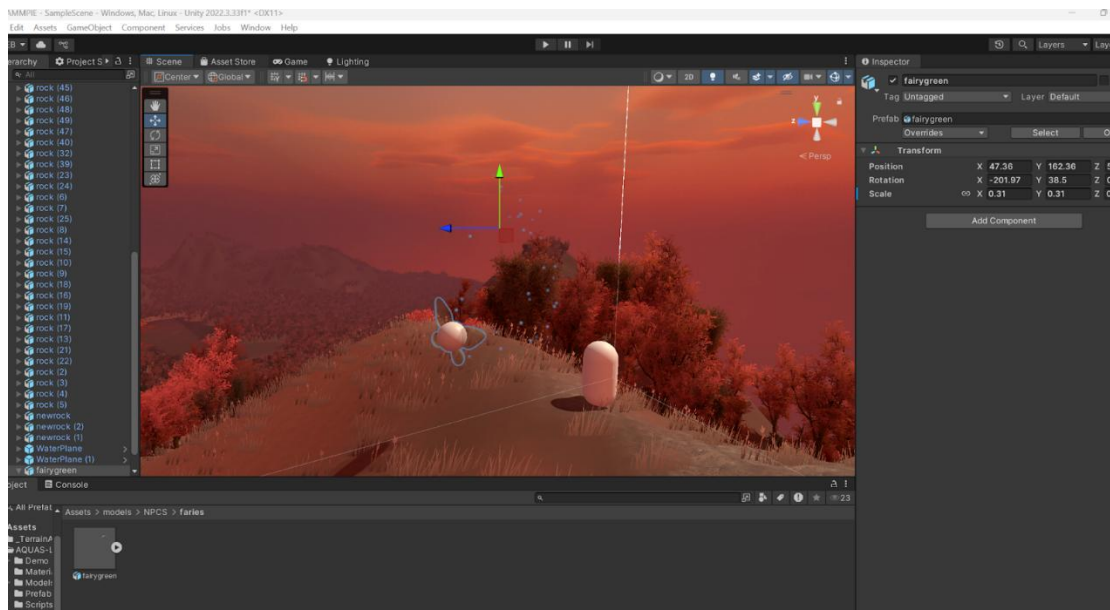
26 Animation sequence

I used single bones in creating armature rig for the fairy and I added a curve to the layout. I used a follow path component and paired it to the curve so my fairy would fly in a circle. This was determined by keyframes changing its offset factor.



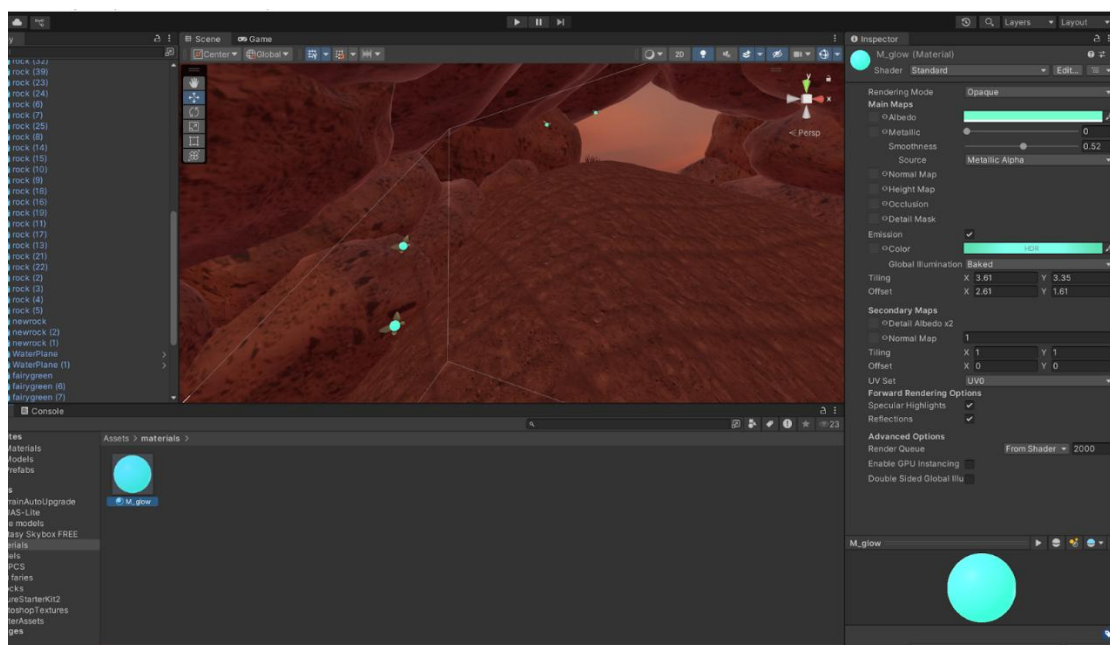
27 Particle effect

I used the particle system inside blender to leave a trail following the fairy. Once my fairy model was completed, I exported the file to unity.



28 Fairy in world

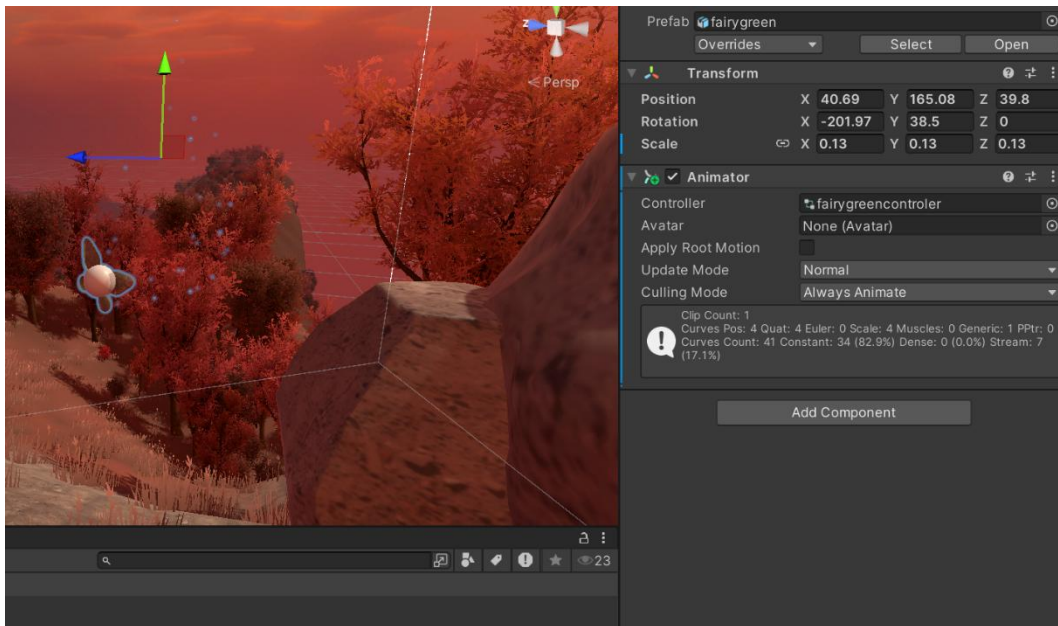
However, I had issues with exporting the material applied to the main body. I created a new material in Unity to replace this but unfortunately was unable to replicate the same effect in Unity as I did in Blender.



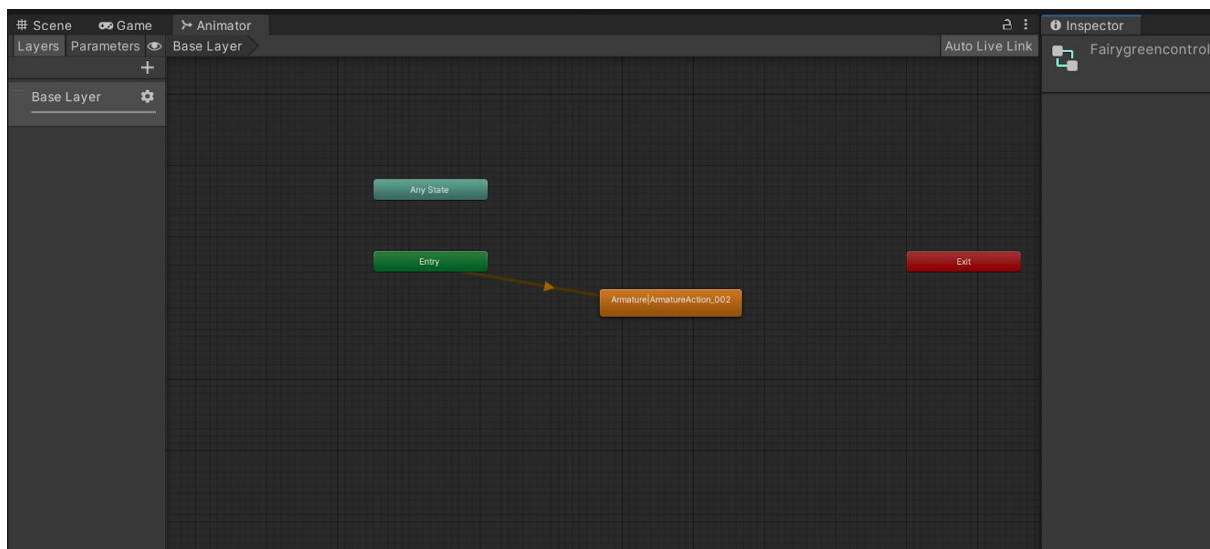
29 New material

To use the animations I created, I added an animator component to the fairy model and also created an animation controller. Inside this, I paired the animation to the entry key and edited

the animation component itself to set the animation to constant loop. When this controller was applied to the animator component, the fairy started to fly around the intended path.



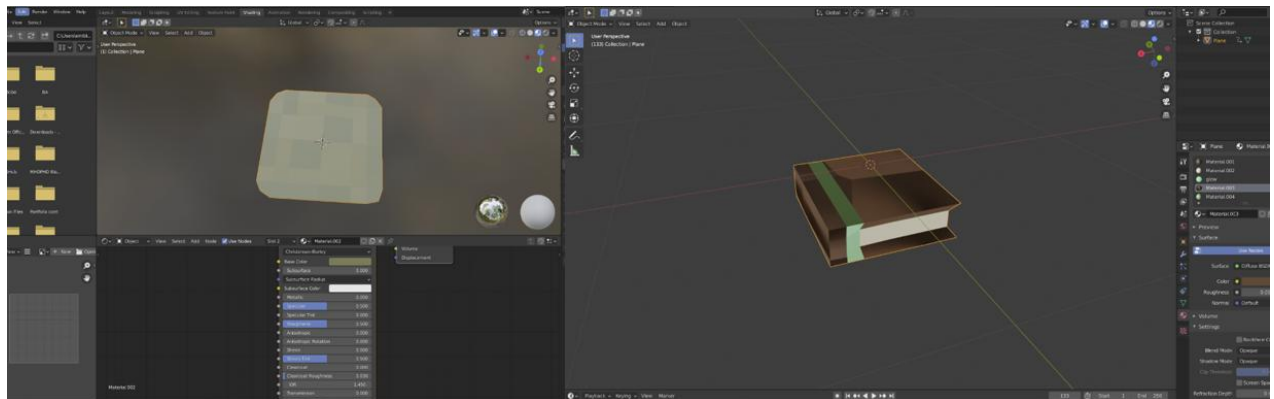
30 Animator component



31 Animation controller

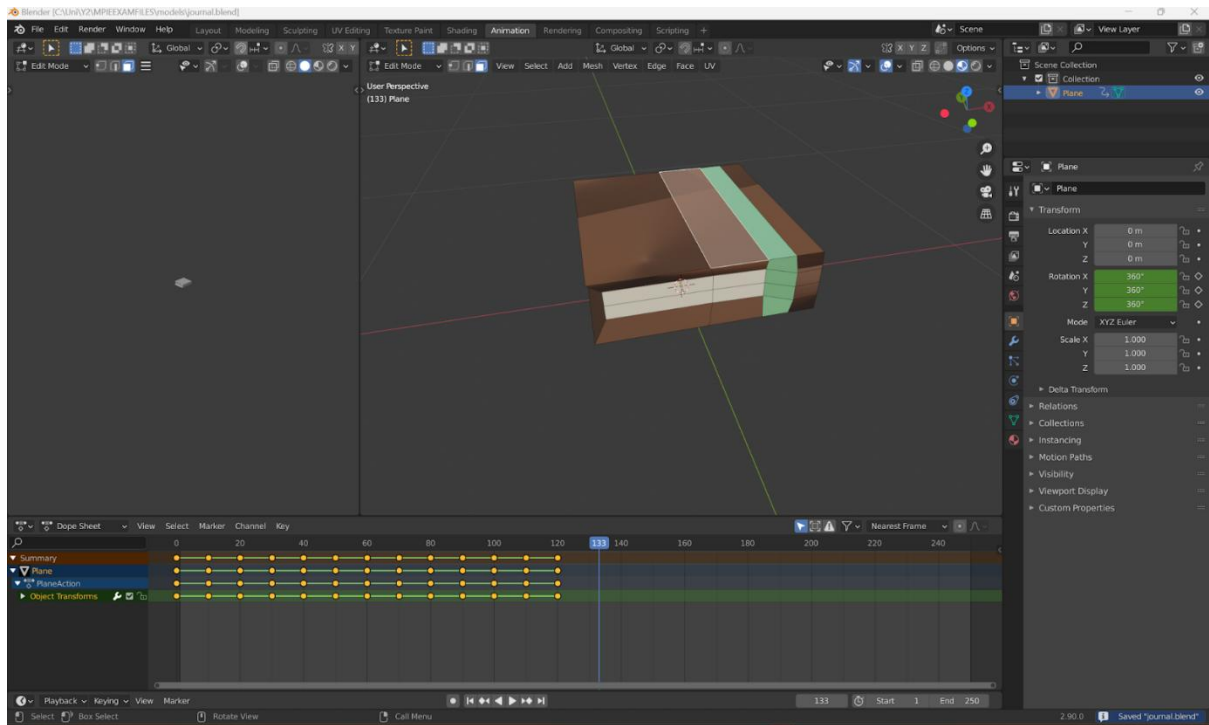
I felt using this in place of the particles added more depth and personality to the world, giving the fairy NPC characters more defined features like the small wings and small trail of glitter that follows their movements. Following the previous success with the fairy models, I moved

onto creating the collectable items within the game. These items would consist of a journal and missing pages from the journal.



32 Item models

I planned for these items to be accessible from an inventory at a later stage of development and reveal sketches and details about the different creatures and structures found in the environment. These pockets of lost information would make the key structures of the narrative in this experience. I followed the same workflow as previously with the fairy model but used the transform rotation tab in the animation panel to animate the objects spinning in a circle. I added a fixed increase on each axis's rotation value at different keyframes until the object had completed a full rotation and reached 360 degrees. This sequence on loop gave the illusion of the object constantly rotating around a fixed point when used in world space.



33 Keyframing animation

To control these animations in Unity, I followed the same structure I had previously with managing the fairy animations and added an animator component to each collectable item as well as creating another animation controller, setting the sequence to the entry key. Initially, the items seemed to completely disappear when the animation was added but I discovered this was not the case and the animation was being played at a different world space location than I had expected. To solve this issue, I created an empty game object and placed it at the intended location. I then made the collectable items a child of this to fix the location to that point.



34 Fixed item position in world view

For the player to 'collect' these objects, I needed to implement a collision system that made the items disappear when the player capsule in the FPS controller collided with them. To do this, I added a box collider to every collectable game object. I then created a C# script to add to my FPS controller which would monitor the collisions with these objects. To do this, my script uses the `OnTriggerEnter()` function and checks if the 'other' being specified is the collider that belongs to a specific object. If this is true, it uses the `SetActive()` function to set the game object to false which removes it from

view.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

0 references
public class PageCollection : MonoBehaviour
{
    // Start is called before the first frame update
    0 references
    void Start()
    {

    }

    // Update is called once per frame
    0 references
    void Update()
    {

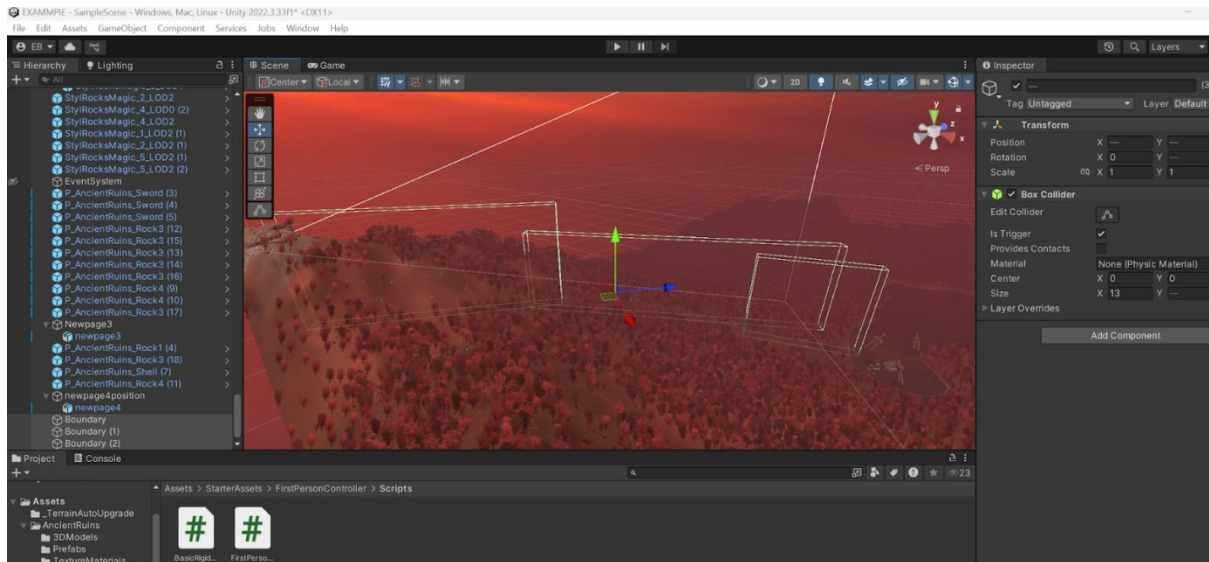
    }

    //Following structure from MPIE-Practical-W5P2-Collisions
    0 references
    private void OnTriggerEnter(Collider other)
    {
        //accesses page near ruin
        if(other.gameObject.name == "newpage") {
            other.gameObject.SetActive(false);
        }
        if(other.gameObject.name == "newpage1") {
            other.gameObject.SetActive(false);
        }
        if(other.gameObject.name == "newpage2") {
            other.gameObject.SetActive(false);
        }
        if(other.gameObject.name == "newpage3") {
            other.gameObject.SetActive(false);
        }
        if(other.gameObject.name == "newpage4") {
            other.gameObject.SetActive(false);
        }
    }
}
```

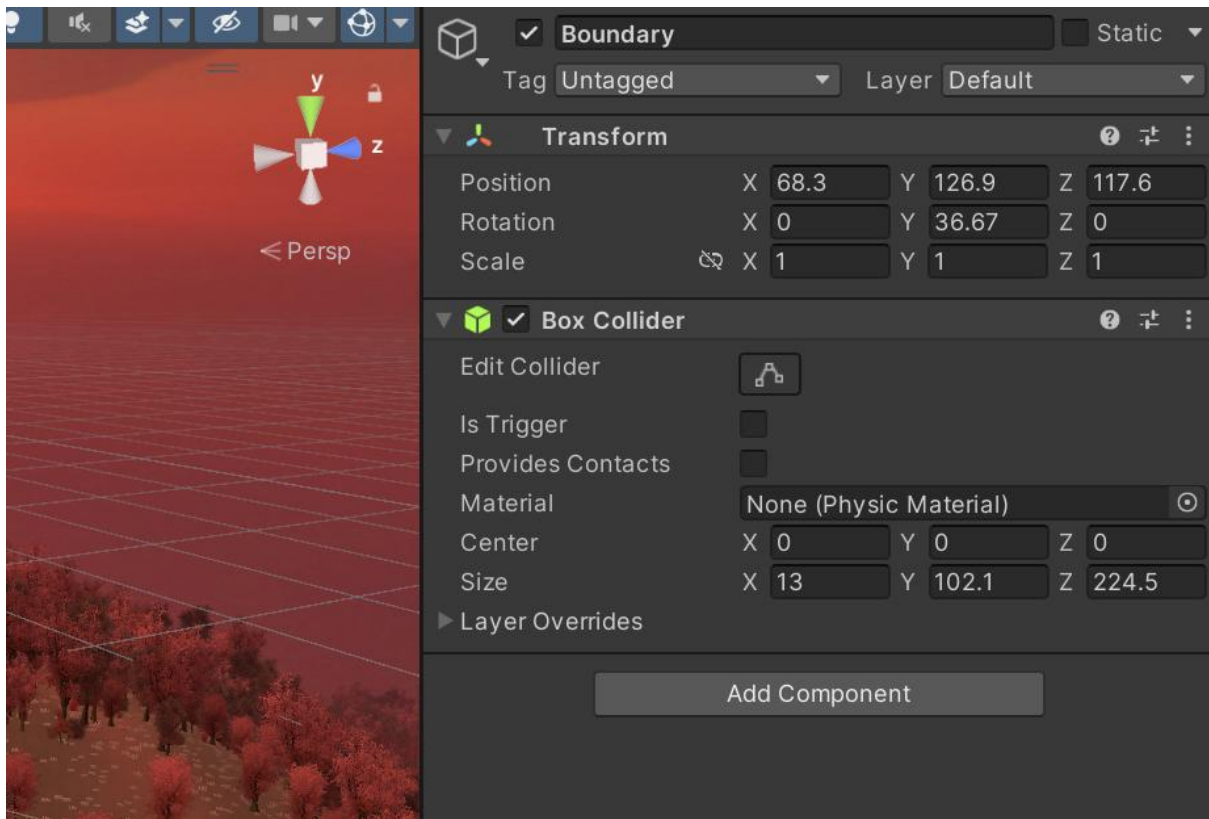
35 Collision detection on item script

Once I had tested these collisions and was confident it worked effectively, I began to set up item collection locations. I used a combination of models from Stylized Rocks with Magic Rune (Comeback, 2021) and Ancient Ruins and Plants (Comeback, 2021) with my own models to create the ruin scenes. I was satisfied with the functionality, design and outcome of this process.

I felt concerned with the player travelling too far off the intended path and decided to add empty game objects which each have box collider. I scaled these colliders to create a perimeter around the intended space.



36 Box Collider barrier



37 Box Collider settings

To further enhance the environment and create a more immersive experience, I added multiple sound features. To do this I imported different mp3 files into unity and paired them to audio source components of game objects. I placed an audio listener on the player camera.

For the movement sound, I created C# script that tracks if the WASD buttons are being pressed using the Input.GetKeyDown Documentation (Unity Technologies, n.d.). This code does not work and the sound continues to play on loop regardless of player input. This could be improved by adding a variable that accesses the audio source inside the original script that monitors player movement and calculates the expected behaviour, and adding a way to play from the audio source into the function that confirms movement.

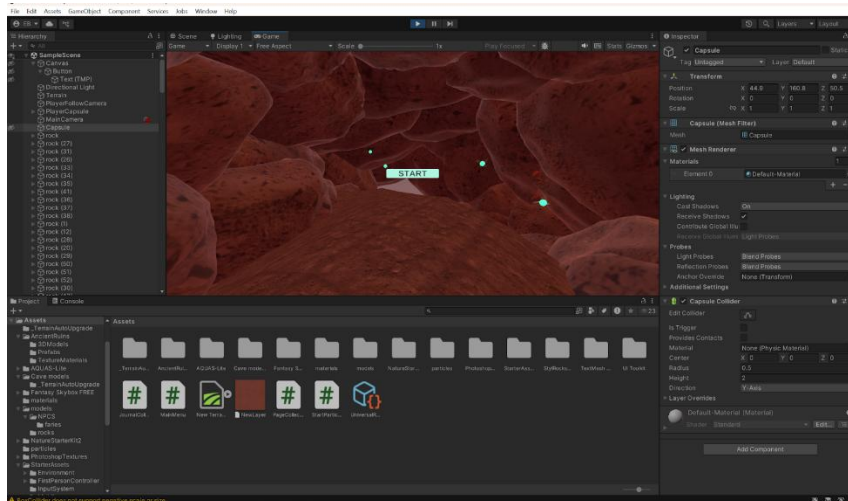
```
// Update is called once per frame
0 references
void Update()
{
    /**
     * code written folloing Unity Documentation Input.GetKeyDown
     * Available at : https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Input.GetKeyDown.html
     */
    if(Input.GetKeyDown(KeyCode.W))
    {
        AudioSource fs = footsteps.GetComponent<AudioSource>();
        fs.Play();
    }
    if(Input.GetKeyDown(KeyCode.A))
    {
        AudioSource fs = footsteps.GetComponent<AudioSource>();
        fs.Play();
    }
    if(Input.GetKeyDown(KeyCode.S))
    {
        AudioSource fs = footsteps.GetComponent<AudioSource>();
        fs.Play();
    }
    if(Input.GetKeyDown(KeyCode.D))
    {
        AudioSource fs = footsteps.GetComponent<AudioSource>();
        fs.Play();
    }
}
```

38 Play Footstep script

I added an ambient forest soundtrack that plays on loop for the duration of the game. I chose this sound because of the subtle white noise and the bird song in the file that was subtle enough to imply the sound was travelling from a distance. This was important to replicate because the world plan is very dense but also open in path spaces and I wanted the sound to match this.

The games main narrative features were dependent on a stable UI management that allowed for 2D pop up planes of the journal pages. This was unsuccessful and I was unable to create and manage various planes for the player to view, losing a major part of my original

gameplay plan. I had issues with using a button as a start game feature. I set up the button on the canvas and managed the intended outcome with the `OnClick()` function accessible in the inspector. My attempts at getting the canvas to disappear after the button was clicked were unsuccessful.



39 Intended UI Functionality

I created a C# script which would manually perform this task however, I was unable to access the button component and received compiler errors in response to my efforts.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

0 references
public class MainMenu : MonoBehaviour
{
    1 reference
    public Button startButton;
    0 references
    public GameObject canvas;

    /**
    * code written following Unity Documentation Button.onClick
    * Available at : https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Input.GetKeyDown.html
    */
    // Start is called before the first frame update
    0 references
    void Start()
    {
        Button btn = startButton.GetComponent<Button>();
        btn.onClick.AddListener(TaskOnClick);

        // startButton.onClick.AddListener(TaskOnClick);
    }

    1 reference
    void TaskOnClick()
    {
        Debug.Log("button clicked");
    }

    // Update is called once per frame
    0 references
    void Update()
    {
    }
}

MainMenu.cs 2 of 2 problems
Button' is a type, which is not valid in the given context (CS0119)
```

40 C# Script and error message

I created a system for onscreen feedback from collecting the different items. A small number displayed by a journal image icon would increase by one for every item collected. I created two separate scripts, one which managed the increase in total value which was paired to the player capsule, and another which gathered feedback from the initial script and converted it to text to display on screen. This idea failed when I was unable to access the total number feedback in the second script.

Reflection

The visual side of the projects development was successful. I achieved my design using a combination of my own 3D models and premade unity assets to build the world. Using the animation sequences added more life to the scenes and helped identify the collectable items as separate to other objects in the world space. The most successful feature was the use of

fairy models in place of the particles. This was less computationally expensive, more efficient and better customised to my needs and desired outcomes.

The final result lacked in efficient implementation of different technical features across the intended mechanics. The issues with managing the UI features in unity removed a key narrative feature that the game experience was structured around. To improve this project, I would revisit the issues with the canvas features in the project and attempt to implement a different way of managing these, potentially managing the screen displays within the original collection script and using the same `SetActive()` function, setting the display to false in start initially and revealing the text after collision. The collectables themselves became hard to spot because of the neutral colouring that blended with the surrounding objects. I considered using spot lights to improve this and draw focus to the objects. In using a more obvious indication of the object location, it would clarify which path is intended for the player to follow.

The game also lacks a clear start and end in terms of narrative features and mechanics. A huge improvement would be to provide the player with some form of feedback to clarify they have completed the task and come to the end of the experience.

Word count: 2622

References:

Nintendo, 1998 *The Legend Of Zelda: Ocarina of Time* [Nintendo 64 game] Japan: Nintendo
ParticleSystem.EmissionModule.enabled Unity Technologies Documentation (n.d.) Available at: <https://docs.unity3d.com/6000.0/Documentation/ScriptReference/ParticleSystem.EmissionModule-enabled.html> [Accessed : 18/01/2025]

Input.GetKeyDown Unity Technologies Documentation (n.d.) Available at: <https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Input.GetKeyDown.html> [Accessed: 18/01/2025]

Appendices:







A - Mood board Image Sources

B – Texture Image Sources

C – Asset Packs

D – Sound Sources

Appendix A

Image description	Image	Source
Legend of Zelda Breath of the Wild Screenshot		https://reactormag.com/i-have-decided-that-being-bad-at-breath-of-the-wild-means-i-am-good-at-life/
Elden Ring <u>Limgrave</u> Screenshot		https://www.eurogamer.net/elden-ring-limgrave-all-dungeons-bosses-map-fragments
The Legend of Zelda: Ocarina of Time, Navi		https://en.wikipedia.org/wiki/Navi_%28The_Legend_of_Zelda%29
Destiny 2, Planet Nessus		https://www.destiny2.com/Nessus
Destiny 2, Nessus		https://destiny.fandom.com/wiki/Nessus
The Elder Scrolls, Ruins of Bthalf		https://elderscrolls.fandom.com/wiki/Ruins_of_Bthalf

Appendix B

Description and usage	image	source	license
Lawn, Grass, Meadow image (Used in paint texture material in terrain editor)		https://pixabay.com/photos/lawn-grass-meadow-texture-yard-253616/	Free to use under Pixabay content license
Gravel, Free background, Desktop backgrounds image.		https://pixabay.com/photos/gravel-stones-dirt-road-fixed-3388388/	Free to use under Pixabay content license

Appendix C

Asset Pack	Image	Source	License
Stylized Rocks with Magic Rune by Comeback		https://assetstore.unity.com/packages/3d/props/stylized-rocks-with-magic-rune-192933	Standard Unity Asset Store EULA
Ancient Ruins and Plants by Comeback		https://assetstore.unity.com/packages/3d/props/exterior/ancient-ruins-and-plants-201914	Standard Unity Asset Store EULA
AQUAS Lite - Built-In-Render Pipeline by Dogmatic		https://assetstore.unity.com/packages/vfx/shaders/aquas-lite-built-in-render-pipeline-53519	Standard Unity Asset Store EULA
Fantasy Skybox Free by Render Knight		https://assetstore.unity.com/packages/2d/textures-materials/sky/fantasy-skybox-free-18353	Standard Unity Asset Store EULA
Starter Assets - FirstPerson Updates un new CharacterController package by Unity Technologies		https://assetstore.unity.com/packages/essentials/starter-assets-firstperson-update-in-new-charactercontroller-package-196525	Non standard EULA
Nature Starter Kit 2 by Shapes		https://assetstore.unity.com/packages/3d/environments/nature-starter-kit-2-52977	Standard Unity Asset Store EULA

Appendix D

Sound	Source	License	
Footsteps Dirt Gravel by freesound_community	Sound Effect by freesound_community from Pixabay	Free to use under Pixabay content license	
Avala Trail Forest Nature by dbsound	Sound Effect by Aleksandar Nikolic from Pixabay	Free to use under Pixabay content license	
Fairy FlyBy 2 by timgormly	Sound Effect by freesound_community from Pixabay	Free to use under Pixabay content license	
Water noises by Soul_Serenity_Ambience	Sound Effect by Soul_Serenity_Ambience from Pixabay	Free to use under Pixabay content license	