Wild-Ireland – Educational Wildlife application that teaches through a first-person game.

TU857

BSc in Computer Science (Infrastructure)

Jake Bolger

C18395341

Tanya Thompson

School of Computer Science

Technological University Dublin

04/01/2022

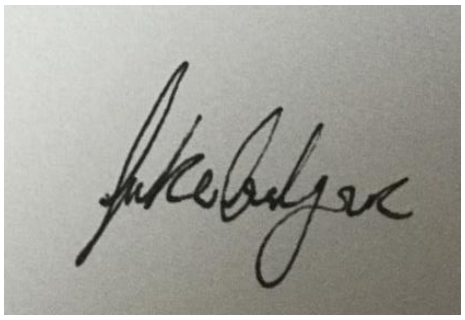
Abstract

Nowadays, it is difficult to teach students about wildlife by only using textbooks. This project attempts to create a game environment that uses procedural generation and AI to allow students to learn about Irish Wildlife and give them a new learning experience. This project uses realistic environments, animal AI, and Quizzes to learn about Irish wildlife in a new and creative way. It focuses on wildlife behaviour and giving the user the right information to help them learn.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:



_____

Jake Bolger

03/01/2022

Acknowledgements

Throughout my time at TUD, there was a lot of guidance and support received through the duration of this project. I would like to thank my parents who supported me through the year, their encouragement and support kept me on track.

I would also like to thank my supervisor Tanya for being a great mentor. She was extremely helpful and always willing to have meetings even over the holidays. Her guidance helped me complete this project.

Table of Contents

Table of Figures

1. Introduction

1.1. Project background

Teaching students in schools and colleges about certain topics can take a boring and uninstructive approach. Many people recall finding it challenging to concentrate and learn information whilst reading from textbooks and lecture slides. Although students can learn this way, reading from textbooks and lecture slides can fail to create an engaging and interactive way for the students to learn. For many students it makes it harder to concentrate for long periods of time.

This project uses 3D modelling, AI, and educational techniques to develop and create an interactive experience that keeps the user engaged in what they are learning about. Using AI to create realistic wildlife behaviours and using quizzes to test the user on the information provided, this system gives the user another option to choose from when learning about wildlife in Ireland.

The use of procedural generation allows the user to experience an immersive environment to explore and discover various wildlife.

1.2. Project Description

The main goal of this project is to be used as an educational tool to teach students about wildlife in Ireland and improve their knowledge of the various species of plants and animals. I plan to accomplish this by creating a 3D game in unity that allows people to have a more interactive and enjoyable learning experience.

The user will be able to create and start a new environment whenever they please. These environments will be randomly generated by the game. This Game will allow the user to move freely throughout the terrain which consists of real Irish wildlife such as plants, trees, and common species of animals. The player will be able to move with a first-person view of the environment and they will be able to explore the terrain as they please. Upon finding different wildlife they will be able to examine the wildlife. By inspecting the wildlife, the user will get access to information about the desired object they are looking at such as videos, images, and text.

Models and assets will be created using Blender and Unity. The animals in this game will use AI Pathfinding to manoeuvre and interact with the player and their surrounding environment, they will also be placed procedurally throughout the environment. Procedural animation will also be used to animate these animals.

Once the player has gathered info about a certain life form which will consist of videos, pictures, and text. They will be able to store this information in an inventory which they will be able to view throughout the game. Quizzes will be available to the user who wish to test their knowledge on what they have learnt.

Sometimes learning through textbooks can be boring and hard to attract the attention of students. The idea is that the students will be able to access this game by downloading from a link or web app. I plan to launch this project and hopefully it will be used in primary schools and educational institutions all throughout Ireland.

1.3. Project Aims and Objectives

The main aim of this project is to provide a game that helps students learn in an engaging way that doesn't just focus on learning from textbooks and PowerPoint slides.

To reach the objectives of this system, goals were set and are as follows:

- Conduct thorough research and review literature relevant to this project.
- Compared similar systems to this project.
- Choose suitable design methodologies and create prototypes.
- Create a procedural environment using procedural coding techniques.
- Design and create wildlife with realistic behaviours.
- Use educational techniques to create quizzes for a learning experience.
- Create and design interfaces that are simple to use.
- Test and evaluate the system by using suitable testing methods.
- Once the process is complete, review and reflect on the system using user feedback.

1.4. Project Scope

This project will have AI that that can interact with the user and environment. It will have a procedurally generated terrain and terrain objects. The animals will be procedurally created and animated. The user will be able to inspect objects and add them to an inventory system.

This AI in this project does not aim to use machine learning for the animals to interact with the environment. It will instead allow the animals to use AI pathfinding to navigate through specific areas of the terrain and use the rules set in the scripts attached to the animals to react to the environment and the player in specific ways for each animal.

The game will be designed to only have one type of terrain which will be a forest, it won't have other terrain found in Ireland such as beaches, mountains, urban areas.

The animations for the animals will be procedural so they will not have unique animations for each one.

The inspect system will not work on every object in the game, only on specific wildlife game objects. The inventory system will not hold unlimited objects, it will hold the number of objects that are available for inspection.

1.5. Thesis Roadmap

1.5.1. Literature Review

This chapter discusses all the relevant background information and research that was completed for this project. Firstly, it looks at the similar systems to this project that have already been created. It then shows the research done into the different technologies that could be used such as games engines, programming languages, database tools, script editors and 3D modelling software. It then goes on to discuss other research such as AI, animation, audio, and assets. Lastly, it goes on to discuss final year projects and academic papers. All this research is described, compared, and decided upon.

1.5.2. Design

For this section, the design approach of the system is discussed which will be used to help develop the system. It discusses version control, software methodology and an overview of the system. It then goes on to talk about requirements, UI design techniques, prototypes, use cases, database, and the overall system design.

1.5.3. Development

This section discusses the development process that has been undertaken to create this system using the architecture presented in the design section. It then goes on to talk about issues in development.

1.5.4. Testing and Evaluation

This section outlines the testing and evaluating done on the project. It talks about the different testing methods used throughout the game such as white and black box testing and discusses in detail the feedback acquired from third-party users.

1.5.5. Conclusions and Future Work

This section will discuss and reflect on the system that was created. It will discuss the conclusions in detail, the issues and risks that came along with it and the plans for future work in this system. A GANTT chart will be used to complete this part of the project.

2. Literature Review

2.1. Introduction

This section looks at, researches, and considers all the different technologies associated to this project. It looks at alternative solutions, programming languages, script editors, render pipelines, modelling software, database tools and various applications.

It also looks at other research such as AI, whilst also focusing on other techniques to be used such as animation, audio, save systems, inventory systems and Unity Timeline for cutscenes.

Finally, it discusses existing final year projects based on this topic.

2.2. Alternative Existing Solutions to Your Problem

On a variety of platforms such as Steam, PS Plus and Xbox games – Microsoft Store, there are existing wildlife games that provide educational experiences to their users.

The following applications are games that provide these experiences.

2.2.1. Beyond Blue

Beyond Blue is an educational narrative underwater diving game that allows the user to explore the ocean and published by an American studio called E-Line Media(1). This game was researched because many of its features share similar aspects to this project, such as interactive animal AI and educational information about wildlife.

This game's features are as follows:

- Interactive ocean allowing tracking of sea creatures.
- Narrative Experience for educational purposes.
- Unique soundtrack.
- Educational mini documentaries.
- Photo mode to allow for pictures.
- Interactive AI animals.
- Animal Collection.

Beyond Blue as a game has many **advantages,** they are as follows:

- It's a good representation of how a game can make learning very interesting for a user.
- It creates an interactive experience that supplies a lot of valuable information about wildlife.
- Allows the user to gather this information by interacting with animal AI throughout the game.
- This game is available on a variety of platforms for example, Steam, Apple Arcade, PlayStation 4 and Xbox One and comes at a price of €16,79.

Disadvantages:

- Although this game has many rich features it does not explore the aspect of quizzes.
- Its main purpose isn't for the user to learn the information being presented.

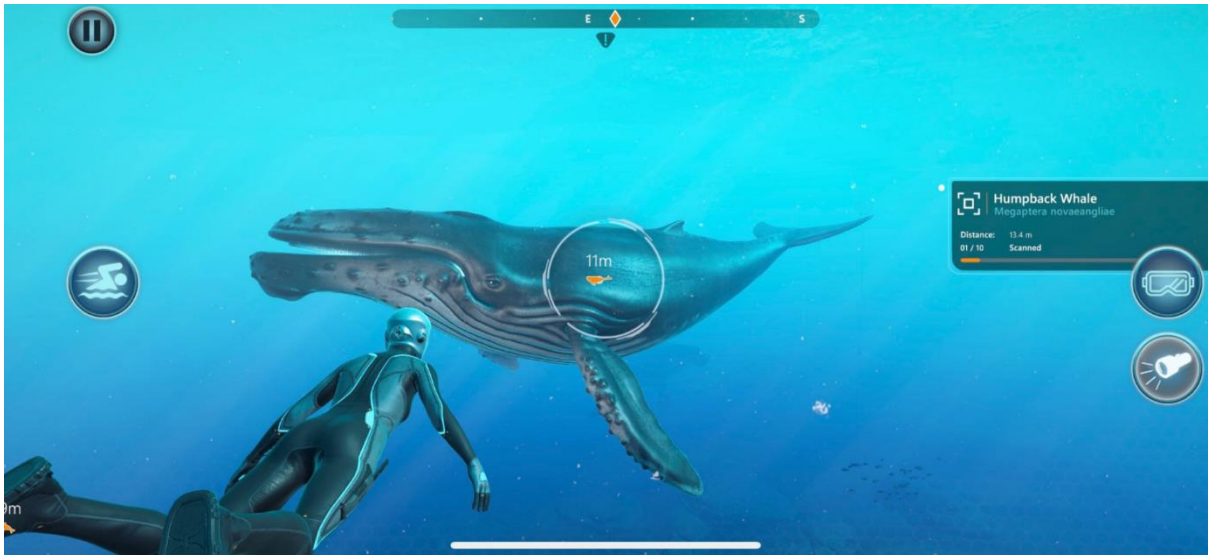


FIGURE 1-BEYOND BLUE

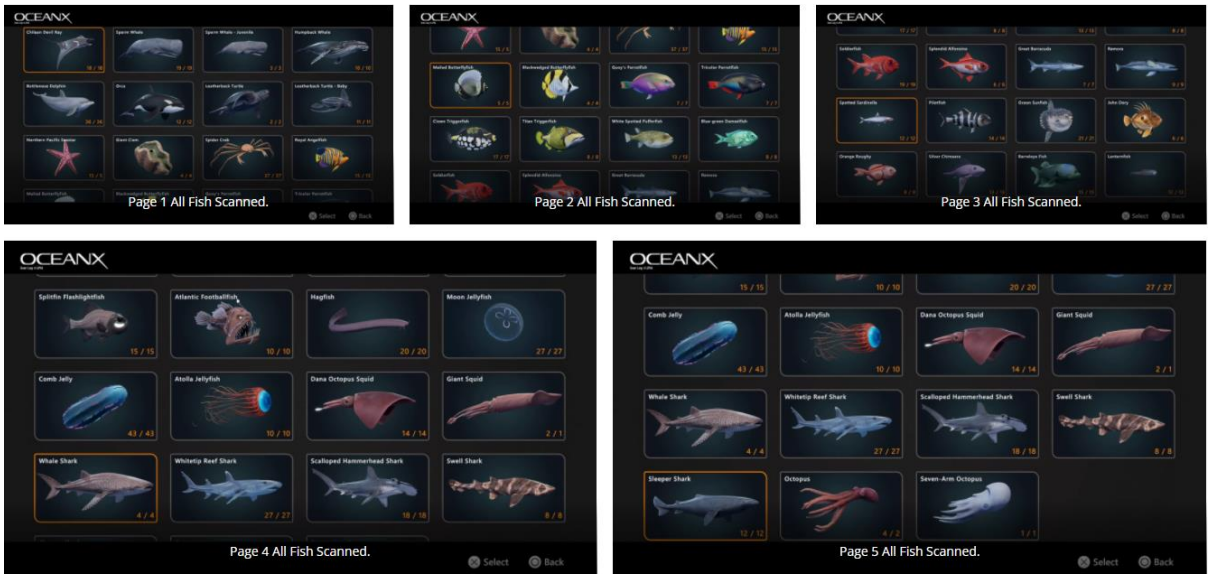


FIGURE 2-BEYOND BLUE

2.2.2. The Hunter: Call of the Wild

This game is an immersive hunting game published by Expansive Worlds and Avalanche Studios(2). The main purpose of this game is hunting. Although the main purpose of this game is not educational it does teach the user how to hunt and it shares a lot of the main features and concepts of my project such as World exploration, animal AI and learning.

This game's features are as follows:

- World exploration / Open world.
- Animal AI.
- Complex animal behaviour.
- Weather events.
- Day and night cycles.
- Use of guns
- Learn how to hunt.

Call of the Wild is a high-quality game, and its **advantages** are listed below:

- Good example of how an open world game can work.
- Good example of how one could implement animals using AI into a game.
- The hunting feature demonstrates how the user can be taught information on different animals.

Disadvantages:

- However, this game isn't very educational.
- Have no methods of letting the user practice the knowledge they've learnt?

FIGURE 3-THE HUNTER: CALL OF THE WILD

FIGURE 4-THE HUNTER: CALL OF THE WILD

2.2.3. Zoo Tycoon

Zoo Tycoon is an educational game that simulates a zoo. The main purpose of this game is to build a zoo and learn about the various animals inside. This game is efficient at teaching children about different species of animals(3). This game is good in relation to the educational aspect of this project as it allows the user to collect species of animals and learn about them via a journal.

Published by Microsoft Corporation, this game's features are as follows:

- 200 animals.
- Create custom zoo.
- Collect species.
- Educational information ("zoopedia")
- Animal AI.

Advantages:

- Zoo Tycoon is a good example of an educational game for children.
- It creates an interactive gameplay environment which allows users to build and learn about animals whilst having an enjoyable experience.
- A feature that was very relevant to my project was the "zoopedia". This allowed the user to press a button and view their "zoopedia" which gave them information about the animals.

Disadvantages:

- This game had no testing feature such as quizzes so the user could test their knowledge.
- This seems to be a reoccurring theme with educational games.

FIGURE 5-ZOO TYCOON

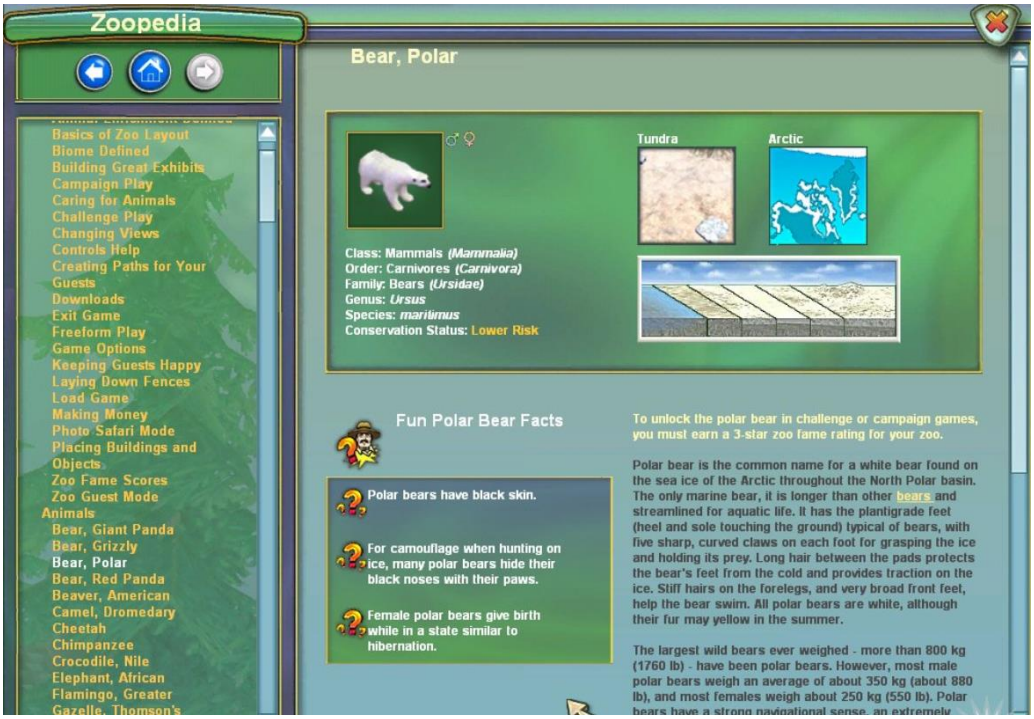


FIGURE 6-ZOO TYCOON

2.3. Technologies you've researched

In this section, multiple technologies that may be used in this project will be discussed. The most viable options that were considered will be discussed and the main advantages and disadvantages will be researched. Finally, an explanation of what technologies were chosen will be provided along with the technologies that weren't chosen.

2.3.1 Games Engines

A game engine is a software tool that is used to design, develop, and create video games. There are lots of different games engine available but only one will be chosen to use on this project.

Below are the games engines that were researched at and considered to use:

- **Unreal Engine**
- **CryEngine**
- **Armory**
- **Unity**

Unreal Engine:

Unreal engine is one of the most advanced and open 3D games engine tools. This engine was developed by epic games and first released in 1988. Unreal engine 4 can be downloaded free and has been used to create a variety of big games such as “Gears of War”, “borderlands” and the “Batman” games.(4)

Advantages:

- This engine provides stunning and workable graphics.
- Its user interface is constantly being updated with the latest tools
- It can enable a user to create games without writing scripts.
- It uses the programming language C++ which is generally the best option for most game developers.

Disadvantages:

- When trying to make simple games, Unreal Engine is not the best choice.
- Unreal engine is only suitable for larger teams of people to work on and projects that are long-term.
- This engine is hard to learn so for developers just starting out it us not suitable.
- The engine also requires the user to pay a 5% tax once the game is profitable.

In relation to this project, the learning curve for this engine might not be suitable for a project of its size and it is not suitable for small projects.

FIGURE 7-UNREAL ENGINE 4

CryEngine:

CryEngine is a game engine designed by a company called Crytek. This engine was used to create all the “Far Cry” games. This engine is free to use and was initially released in 2002.(5)

Advantages:

- Support for VR
- Very little programming required
- Easy to learn for someone to learn.
- Has good support and its graphics capabilities are quite good.

Disadvantages:

- CryEngine’s userbase is quite small.
- There are fewer ways to find help when using it.
- It is not suitable for small development teams or projects.

In relation to this project, although it’s easy to use, it is not suited to a small project like this.

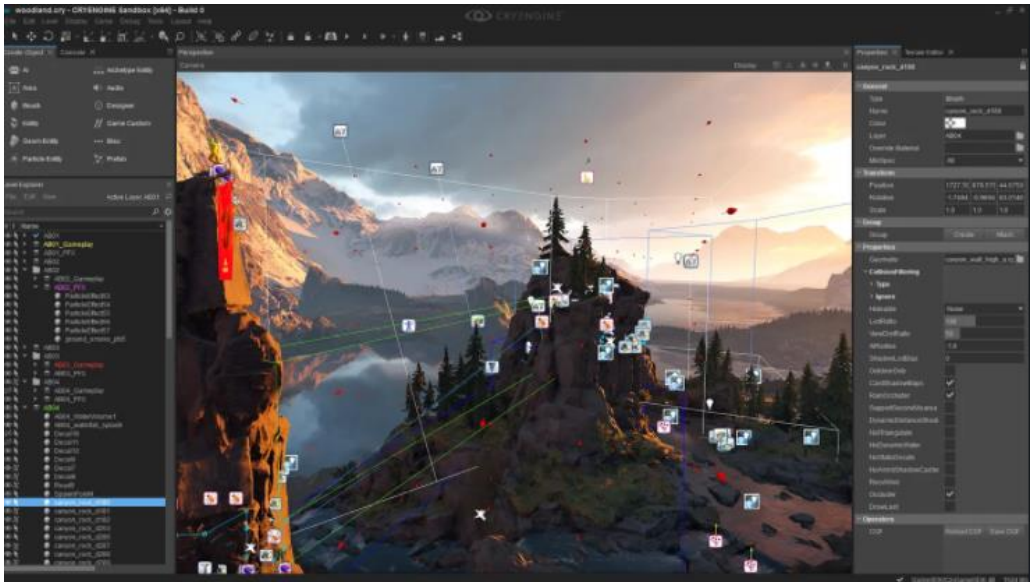


FIGURE 8-CRYENGINE

Armory:

Armory is a 3D Engine that is open source and provides a full Blender add-on. It's written in the HAXE programming language. Games created in this engine are "Offroad Mania", "Ghost Rush" and "Xpedition". (6)

Advantages of Armory are:

- Armory is an Open-Source engine.
- This engine provides good Flexibility.
- Logic Nodes make a visual way of creating interactive behaviour.
- The coding for this engine it quite easy compared to other engines.
- This engine provides a Blender add-on.

Disadvantages are:

- This engine is not even fully developed.
- The user base for this engine is very small providing low support.

In relation to this project the blender add on would be a good advantage, but the engine is not finished.

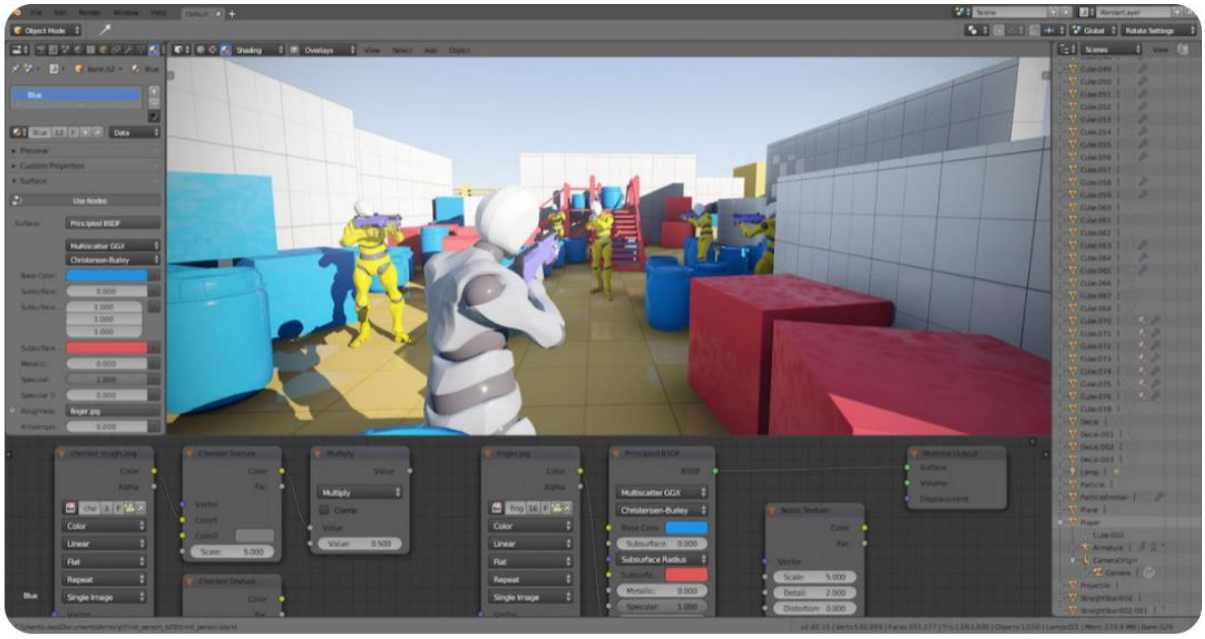


FIGURE 9-ARMORY

Unity:

Unity is a game engine developed by Unity Technologies. It was released in 2005. It is cross-platform and is free to download and use. This engine is very popular and has lots of big games that have been created using it. “Cuphead”, “Escape from Tarkov” and “The Forest” were all created in Unity.(7)

Unity Advantages are:

- Its Free to use.
- There is a huge Community support for Unity.
- Unity is well known for being suitable for small teams.
- Interface is easy to learn.
- Uses C#. This language has tons of support and is easy to learn.
- Asset store. This provides lots of free assets to be used in games, such as models, objects, code.

Disadvantages are:

- Graphics aren’t as good as other engines
- Not suitable for AAA games.

In relation to this project its perfect for small teams like this, it’s got a lot of support so if there are issues, they can be resolved easier. Some risks that might be associated with Unity is that using too much memory or space in the project that slows the project or game down. Assets like materials and textures take up lots of memory. Optimizing the memory space in Unity is important to keep the project efficient.

FIGURE 10-UNITY

Decision

The games engine “Unity” was chosen. The main reason for this is that my project is a one-man game which Unity is very suitable for. Unity has great community support with lots of information online about it which will be helpful when creating this game. It also uses C# which I already have experience in.

The reason for not choosing any of the other engines listed is because they all have bigger learning curves and are not suitable for small teams such as one person. There is very little support for some of them which I will need in the future.

2.3.2. Programming Languages

In Unity, to use many of this engine’s features, scripts need to be written using a programming language and used to give it instructions to create games. there are a few different languages that I have researched and considered. (8)

The languages are as follows: C#, JavaScript / UnityScript, Boo, IronPython, MoonSharp (Lua), C/C++ and Rust.

C#:

C# is a general-purpose programming language that supports a variety of programming disciplines. Commonly known as the best option for Unity.

Advantages:

- C# is the right language for anyone starting or of little experience in Unity.
- All Unity’s libraries are built in using C#.
- C# is easy to learn and far more accessible than other languages for Unity providing greater support.

Disadvantages:

- In terms of disadvantages, C# doesn’t have any when it comes to coding in Unity.

UnityScript:

This language is modelled after JavaScript and is designed specifically for Unity.

Advantages:

- This language is good for developers who come from a JavaScript background.
- Similar to JavaScript.

Disadvantages:

- Even though UnityScript is like JavaScript there are also a lot of differences such as classes and multiple variable declaration.
- A huge disadvantage is that Unity announced that they will withdraw support for this language. The use of this language would be unwise.
- Unity withdrew support for it.

IronPython:

Iron python is a variation of the python programming language. IronPython isn't a good language to develop games and to use this language you need to download the IronPython libraries form GitHub. Overall, this language isn't very good as Unity relies heavily on C#.

Lua or MoonSharp:

MoonSharp is more commonly used to act as a bridge to C#.

Advantages:

- it would be useful for users to create game mods.
- MoonSharp is worth considering if you want to interface with your C# code.
- It is also free to download from the Unity asset store.

Disadvantages:

- Low amount of support.
- Not very popular.

C++ :

This language would be mainly used for plugin creation. However, if you can use C++, you might as well just use C# instead.

Advantages:

- Good for plugin creation.

- Similar to C#.

Disadvantages:

- Very low popularity for Unity.
- Doesn't serve too much of a purpose.

Rust:

This is a relatively new language. it was created by Mozilla.

Advantages:

- This language is better to use than C++.
- Its function is to develop high-performance software in a shorter time limit.

Disadvantages:

- Not much point in using it for this project.
- Doesn't serve a purpose for this project.

Decision

The programming language **C#** was chosen because it is far superior compared to the other languages in Unity. There is a huge amount of support for it. It's extremely popular and all Unity's libraries are in C#. It's clear that C# works well with Unity and the statistics from the research I have done previous show that the decision to use C# is the correct choice.

FIGURE 11-C#

2.3.3. Unity Script Editors

When using scripts and writing code in unity, a script editing tool is needed so that the developer can input the code and use it in Unity.

Below are the tools I have considered:

- Visual Studio Code
- MonoDevelop

Visual Studio Code

This tool is owned by Microsoft and has been integrated for Unity. It runs on all operating systems and is a very popular editor for Unity.

Advantages:

- Is good for editing and debugging.
- The C# features in Unity are well supported in visual studio.
- It's the most popular editor for unity. Giving it a lot of support.

Disadvantages:

- Visual studio takes longer to start up than other editors.

Monodevelop

Monodevelop is an integrated development tool that works for Unity that runs on all operating systems.

Advantages:

- Good for small Projects.
- It is an open-source environment.

Disadvantages:

- when using Monodevelop with unity it is less stable than Visual Studio.
- Even though it is good when dealing with small projects, Visual studio is better.

Decision

The decision was made to use Visual studio. It is far more popular than Monodevelop and is more stable with better autocomplete features. It works just as well if not better than Monodevelop with projects of any size which is perfect in relation to this project.

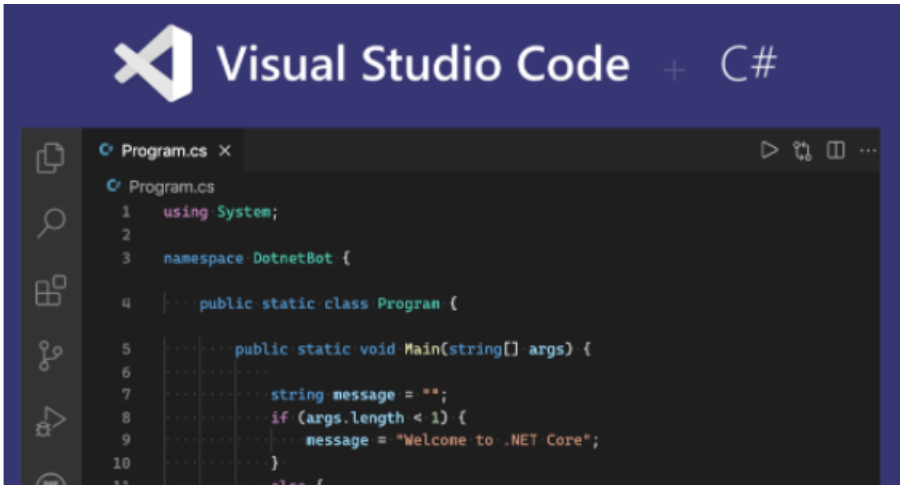


FIGURE 12-VISUAL STUDIO

2.3.5. Unity 3D Render Pipeline

When using unity, a render pipeline needs to be picked. A render pipeline allows Unity to perform the steps needed so that it can render a 3D or 2D scene. This influences the performance of your Game.

When researching render pipeline I narrowed my decision down to two pipelines, **HDRP** and **URP**.

The Universal render Pipeline

This pipeline allows you to boost performance without having to use code to do it. It is compatible with all unity platforms and is the best pipeline**.**

The high-Definition render Pipeline

On terms of performance this render pipeline focuses more on the graphics and is suitable for high quality graphics projects.

Decision

From my research the URP has shown that it's the best option putting performance over everything else whilst still giving beautiful visuals. This pipeline will also be used in creating the inspect feature for my project by creating multi-layer rendering.

FIGURE 13BEFORE AND AFTER USING URP

2.3.6. 3D Modelling

For this project, models of various animals will be needed. Instead of buying or downloading premade assets from the asset store. It was decided that I would be modelling the animals myself. This is because you often must pay for good quality assets. Modelling these animals, myself shows a higher level of skill and increased complexity of my project.

When doing 3D modelling you can either use Unity itself or you can choose an external modelling tool outside of Unity.

The 3D Modelling tools that were **researched** and **considered** are as follows:

- ZBrush
- SketchUp
- Blender
- 3Ds Max
- Maya

ZBrush

ZBrush is one of the most advanced 3D sculpting tools allowing the user to mimic traditional sculpting techniques. It was published in 2009 by Pixologic, Inc.

If you want to do detailed sculpting work with high-poly work, ZBrush is ideal. However, the learning curve is quite steep and it's not very good for renders. The cost of a ZBrush license is $795.

SketchUp

SketchUp can be used for a variety of 3D modelling. It has a large community which can be very useful and the capabilities of SketchUp are good. However, the rendering is limited, and coding is needed to be able to use it which will make it less efficient.

Blender

Blender is free and open source, and its modelling capabilities are very broad. Blender has tons of support and a large community making it easy to learn and use without much experience making it good for small developers. In terms of disadvantages, it doesn't have many except for the effects on AMD hardware making them slower. Some issues that might be experienced is exporting the models into unity. The textures and materials might not be exported correctly. Another issue is blender models can be made in detail and when using the models in unity they might cause unity to run slower or handling the models might cause problems for example if you created a tree model in Blender, there could be thousands of leaves on that tree. Unity could have a hard to processing the model.(9)

3Ds Max and Maya

3Ds Max and Maya are both made by the same company Autodesk. 3Ds Max has probably the broadest range of features out of all the modelling tools. It is mainly known for its modelling capabilities whilst Maya is more geared towards animation. Both software's take a long time to learn which would make them unsuitable to a developer like myself who is short on time. The two software's also cost money to use which is not ideal.

Decision

The 3D modelling tool **Blender** was chosen to use as it has lots of support online and its learning curve isn't as steep as the other. It is also free and more suited to the scale of my project. I have a small amount of experience in blender as well, blender was the obvious choice.

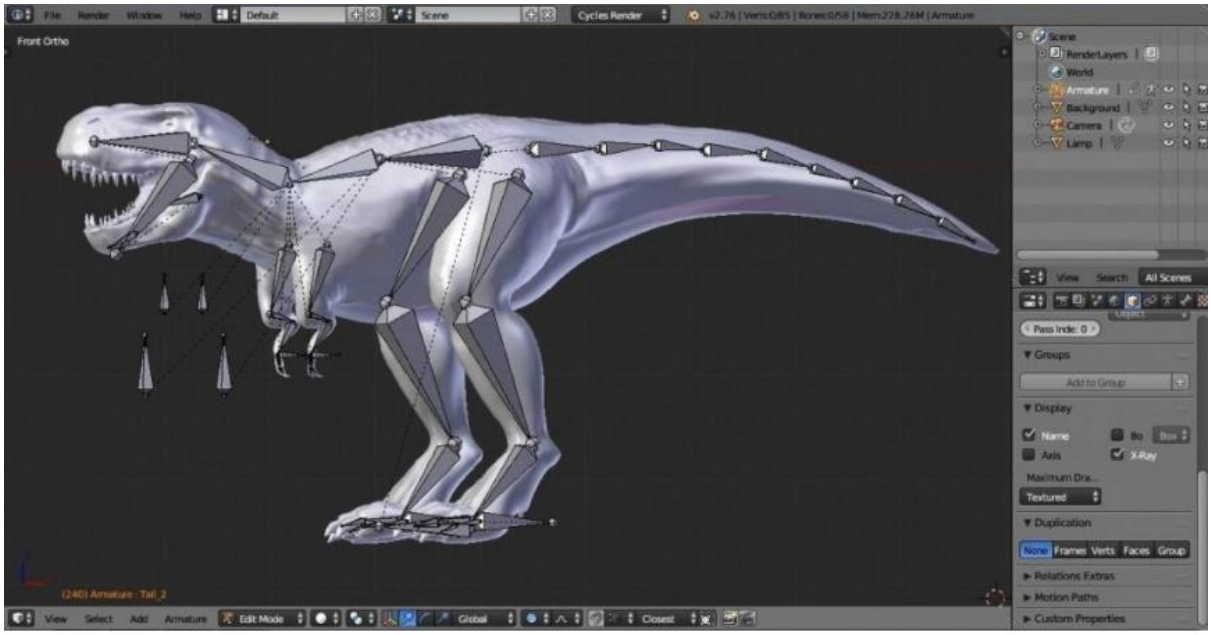


FIGURE 14-BLENDER

2.4. Other Research you've done

2.4.1. Cutscene Research

For this project, I will need to create and use cutscenes in Unity. In my research, I have found the best way accomplish this will be by using the Unity Timeline Editor and Cinemachine.(10)

Timeline Editor: In Unity, the Timeline Editor is built in tool which enables the user to create cinematic content such as cutscenes and audio sequences.

Cinemachine: works wells with Unity timeline allowing the user to have access to dynamic cameras that improves cinematic experiences.

2.4.2. Inventory System Research

In this project an inventory system will be used to store wildlife that the player has found. In this section the deciding on what method of how this inventory system will be created is being researched.

When researching how to create an inventory system in Unity, there were two methods looked at. The first one was to download a **premade inventory** asset from the unity asset store. The second method was creating an **inventory**

system in Unity using C# to write scripts and classes to build a system that handles items and has a User Interface to display the inventory. (11)

When using a premade inventory system, the advantages were:

- Less workload involved in process of creation.
- Quality of inventory systems were high.
- Easy to use.
- Saves times on learning how to make an inventory system.

Disadvantages:

- All the premade systems cost money.
- Takes away from the complexity of the project.
- Harder to customize or change the inventory system.
- Might not suit the specifications of this project.

When creating an inventory system in Unity the advantages were:

- Self-made inventory system is more customizable.
- Can be tapered to the exact specifications of this project.
- Adds more complexity to the project.
- Is free to create the system in Unity.

Disadvantages:

- The quality of this inventory system might not meet the standard of the project.
- More workload involved in this process.
- Takes more time to learn how to create this system.

Decision

The method chosen to do was creating a self-made Inventory system in Unity. This was chosen was because when writing the code and creating the user interface yourself, it allows for better customization to fit the specifications of this project, and this games system. It adds more complexity to the project instead of just using a premade inventory system making the self-made system the better choice.

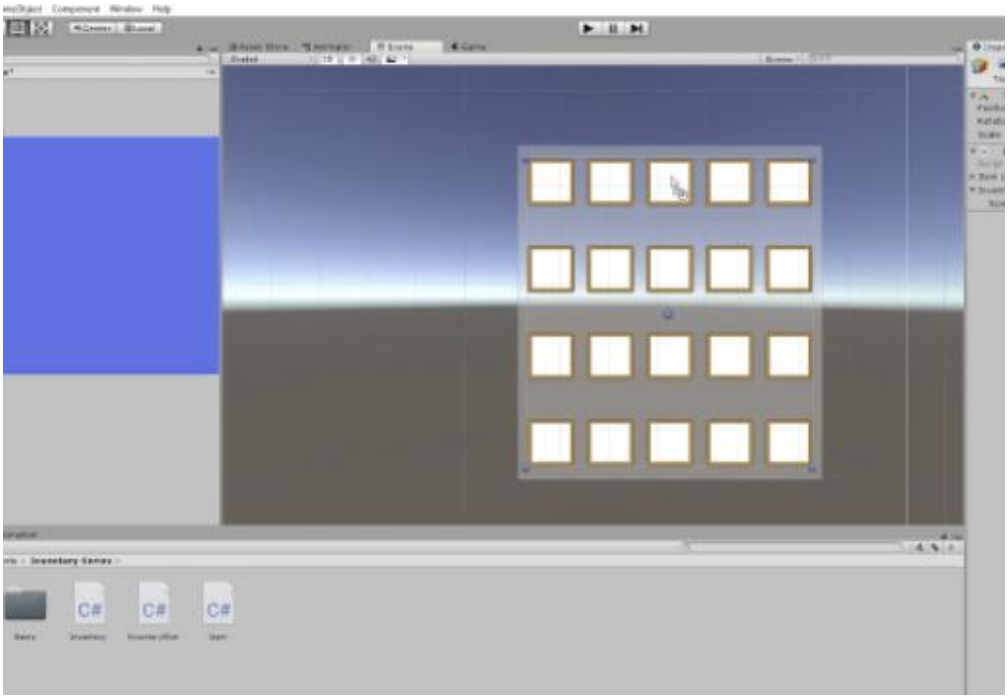


FIGURE 15-SAMPLE INVENTORY SYSTEM

2.4.3. Inspection system Research

In this project an inspection system to inspect various types of wildlife will be needed. Research on what method was going to be used was researched in this section.

When researching how to create an inventory system. There were two methods researched. The first method was creating an inspection system in Unity and the second was using a premade asset from the asset store.

The first method consists of creating various C# scripts and classes in unity, and using a technique called Raycasting to create this system. Raycasting is when an invisible ray goes from one point to another and detects whether an object is in the path of the ray.(12)

The advantages of creating the system in Unity are:

- Easier to taper the specifications to the game.
- Doesn't cost any money.
- Adds complexity to the project.
- Ray casting is a reliable and efficient method.

Disadvantages:

- Higher workload and difficult to complete.
- Raycasting can be difficult to learn.
- Quality of system might not be as good as the premade asset.

Advantages of downloading a premade inspection system:

- The quality of the system could be very high.
- Easy to use the systems.
- Saves a lot of time creating one.

Disadvantages:

- All the premade systems cost money to buy.
- The project would be less complex if used.
- Harder to customize to suit specifications of the game.

Decision

It was decided to create the inspection system in Unity and not download a premade one. Whilst the premade ones are of high quality and quicker to implement, the custom inspection system will be more suited to the specifications of this project, and it doesn't cost any money to create.

FIGURE 16-SAMPLE PROMPT SYSTEM

2.4.4. Save system Research

In this project, the user will need to save their data and their progress in game. For this a method of saving the data will be used. Normally, for small amounts of data you can just use the built in PlayerPrefs (class that stores preferences) in Unity, but for this project other tools will be needed.

The different ways for saving data that were researched and considered are as follows:

- XML and JSON files
- Binary Files
- Easy Save – Asset
- MySQL with PHP

XML and JSON: This way is quite simple to do but it's not very efficient, and the files are easily modified by other users which is a security risk. These files are also used outside of Unity.

Binary Files: Binary files are a lot harder to modify making the security a lot better and are used inside Unity. These files are perfect for games that want a more complex save system than just using the PlayerPrefs in Unity. However, it can become slow to work with binary files as your project gets bigger. to use binary files in Unity you must write C# scripts and create classes to implement this system.

Easy Save: An easier way and a long-term solution instead of binary files would be a free asset like "Easy save". It saves a lot of time and allows you to save unity specific data, encrypt the data, handle a lot of data at once and save and load data from the internet.

MySQL with PHP: When using MySQL with unity there is a steep learning curve compared to other options. For projects like this MySQL is probably not needed for what is needed to be accomplished.

Decision

The method of using binary files to create a save system was chosen because it is more efficient than the other methods listed. It is well suited to the size and type of game that is being created for this project and it will allow us to create a system using C# scripts and classes so the user can save and load their game.

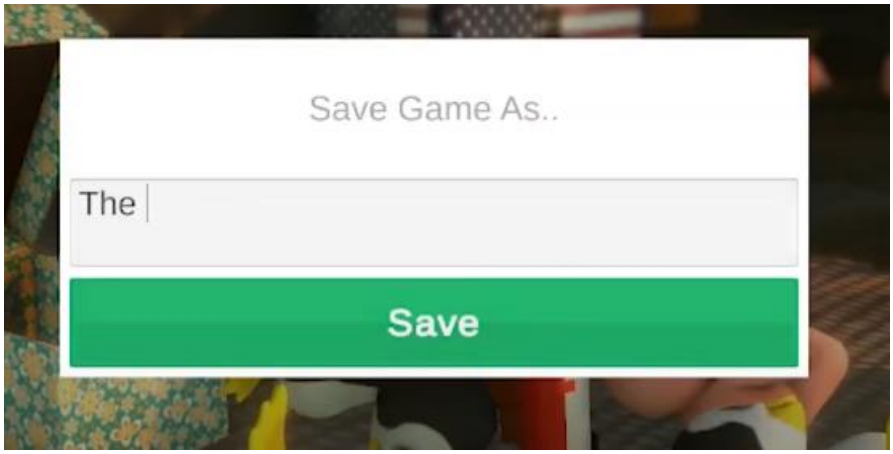


FIGURE 17-SAMPLE SAVE SYSTEM

2.4.5. Procedural Generation

Procedural generation is a programming technique that allows the developer to create data according to the programs instructions that were made by the programmer. During the development process, lots of different games generate parts of the environment or non-player characters procedurally to save time, instead of creating the assets individually. An example of this is if the developer wanted to create a forest in Unity, instead of creating each tree and bit of grass individually, they would write a program to generate the forest all at once when the game starts.

Terrain

In Unity to be able to procedurally generate terrain or levels, you must use a noise function. noise functions mean that they generate pseudorandom values.

The noise that will be used in this project will be **Perlin noise**. This allows you to write code to generate pseudo random patterns that consists of waves that increase and decrease across the terrain or level. An example of this would be in Minecraft where each time a world is created, it is pseudo randomly generated by code using Perlin noise. This will be used to generate the map terrain and the forest wildlife in the game.(13)

Advantages:

- This method can create lots of content for a game.
- Can create large games more efficiently.

- Can be used in low budget games but make them seem like they are higher budget.
- Better gameplay variety.

Disadvantages:

- Uses more hardware resources.
- Creating scripts for this method is harder.
- terrain can be repetitive.

Reason for choosing:

The main reason for choosing this method is because it is suited to this low budget project. It is efficient at creating large worlds whilst adding an element of complexity to the project when coding. It also allows the game to generate a new world each time they start a new game.

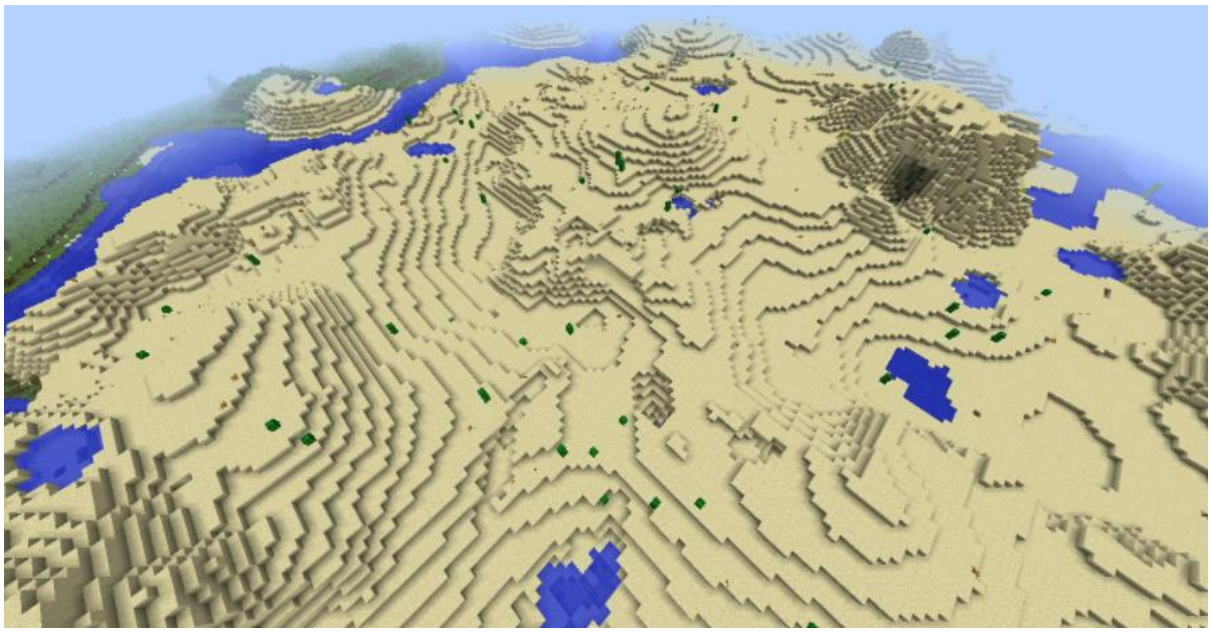


FIGURE 18-MINECRAFT PERLIN NOISE

2.4.6. AI Research

In this project a form of AI will be needed to allow the animals in the game to interact with their surroundings, the user, and behave in a certain way. To achieve this, various forms of AI were researched and decided upon to be used in this project.

Finite State Machines

The main form of AI and behavioural management that was looked at was Finite State Machines. The idea of an FSM is that a machine can be in a certain state from a range of different given states. For example, in a game if an AI player is given the state of walking around and it detects another nearby player it will go into an attack state, however, if the player moves out of sight the AI will go back to wandering.

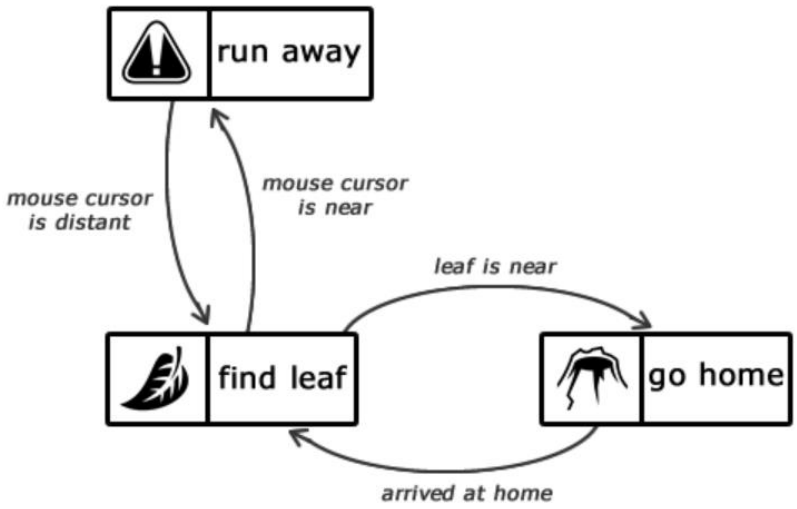


FIGURE 19-FSM

Bolt

When researching different techniques of AI, I came across a visual scripting tool for unity called Bolt. This technique allows you to work on unity projects and aspects such as AI without using any code. This could be a solution for tasks that are too difficult and take up too much time to complete within the given time constraints.

ML-Agents

In Unity, there is a plugin called ML-Agents. This plugin enables the developer to create environments that train the AI agents in their game. This means that the agents will be able to learn by themselves. This method is used in AAA games and allows you to teach intelligent behaviours to AI agents. This method tends to have a higher level of difficulty, but it also has lots of support from Unity.

FIGURE 20-ML-AGENTS

AI Pathfinding using Unity NavMesh AI

In games, characters often must navigate around obstacles in the level. This is a common scenario in games, so Unity provides a built-in pathfinding system, called NavMesh. This NavMesh system allows you to write C# scripts to set instructions for how the non-Playable character, in our case an animal should act with the environment.

An example of this, if the animal wanted to navigate around a specific area but if it came within a certain distance of our player, the animal would run away or maybe chases after the player according to the instructions in the C# script that were set by the programmer using the NavMesh AI system.

From my research I have found that NavMesh is very helpful and allows you to do hundreds of pathfinding calls per frame which is very efficient. This is a component that will certainly have a part to play in this project. It will allow me to create C# scripts and write programs that will be able to set specific behaviours for each animal in this game.(14)

An issue that might arise when using NavMesh is when the NPC's need to navigate around obstacles and terrain. Whilst researching Unity's pathfinding system there seems to be lots of minor error and bugs with the NavMesh system, when navigating around complicated obstacles. For example, if

characters are moving through narrow pathways, they seem to get stuck and are unable to move away from the other characters.

Decision

The method that was chosen for the animal AI in this project was Unity's AI pathfinding system NavMesh. The reason for this is because it will allow me to write C# scripts for each animal in the game to set what behaviours they have and how they will react to the environment and the player. This method of AI is perfect for this project because it isn't too complex making it suitable for a small development team such as mine, but it also has a coding element to it which still adds a certain level of complexity to the AI.

2.4.7. Animation Techniques

Keyframe Animation

Keyframing is the most popular and most simple animation technique. This technique allows you to draw a shot and move different parts of your object to then measure the position and time in frames. For example, the animator moves a limb of a character's body whose movement is saved each keyframe. When this is completed, the movement is turned into an animation. An advantage of this is that the movements creating can be made to look super realistic. However, it is quite hard to make these movements as realistic as they can be.

Motion Capture Animation

Motion capture is a technique that uses reality and animation combined. This method means you must use equipment such as bodysuits for a person to act out the movements and then are recorded as animations. This method is used in AAA games as it allows the animators to capture realistic facial and body expressions, but it generally cost a lot more to do compared to other animation techniques.

For this project the two best options for animation tools are the Animation Controller in Unity or the animation editors and tools in Blender.

Procedural animation

Procedural animation is a type of animation that will generate the animation automatically rather than creating the actions in advance. This is done by

writing code, for this project the code would be written in C# scripts in Unity and attached to the Player.

This technique doesn't look as good as other animation techniques but is far more efficient. It allows the developer to re-use all the animations on the different characters as it can use similar code to drive these animations.(15)

An example of this is if you have multiple animals that are similar in the way they move. You would only need to change the code slightly for each animal to animate them. When using other forms of animation this is not the case.

Decision

The decision was made to use procedural animation. This technique was chosen as there is limited time for this project meaning procedural animation will be more efficient when dealing with different animals to animate. Code will be able to be re-used saving more time and adding more complexity to the programming aspect of this project.

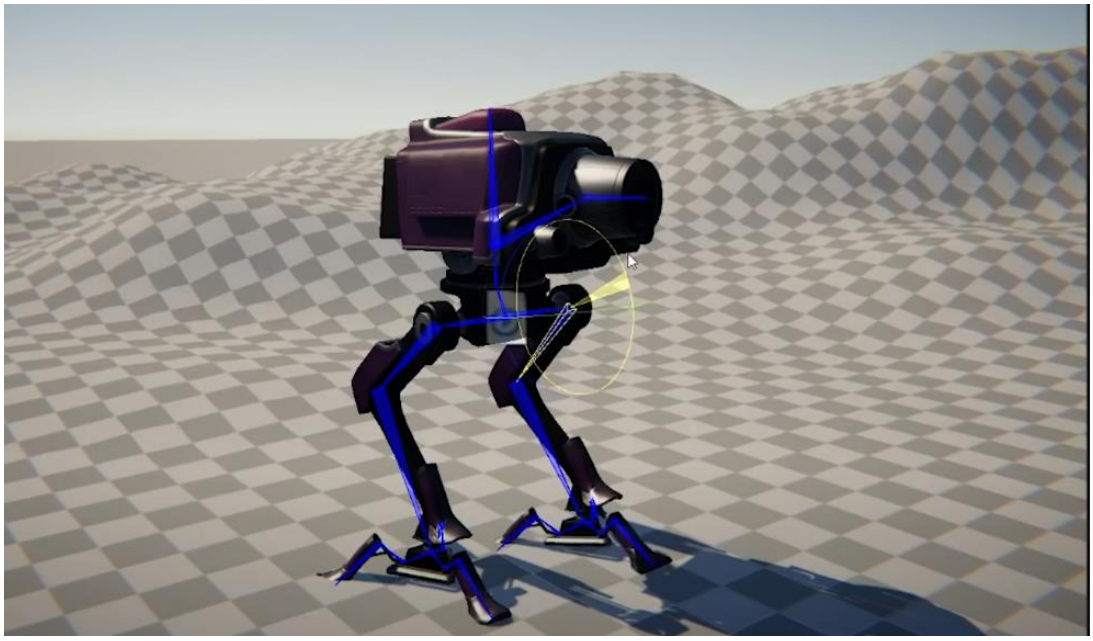


FIGURE 21-PROCEDURAL ANIMATION UNITY

2.4.8. Audio in Unity

When researching audio in unity, the research has shown unity's build in components are best for audio in unity. Components such as audio player,

audio listener and audio manager combined, along with the use of scripts. They are simple to use and will suffice for this project.

When searching for downloadable music and sounds for this project. A website called mixkit.com was found. This site offers free downloadable audio sources that could be put in this game.

The decision was made to use this website and Unity's built-in audio features as they are easy to use and don't need to be complex.

2.4.9. Unity Assets

Even though many assets in this project will be modelled either in blender or unity, there are still some that will be downloaded from the Unity Asset store.

These assets include:

- Grass.
- The Sky Shader.
- Sticks.
- Rocks.
- Grass material.

2.5. Existing Final Year Projects

Two final year projects from previous years were researched. These projects were both in some way relevant to this project to aid in the research and design.

2.5.1. Project No.1

Title: EvolVR – Investigating Procedural Ecosystems and Evolution
Student: Ryan Byrne

The goal of this project was to create a virtually simulated ecosystem of animals from multiple animal groups to better demonstrate the evolutions animals make over multiple generations. The environment is procedurally generated so that animals can be viewed adapting to a multitude of different habitats of varying hostility and seeing how traits of the animal might develop differently from one placed in a different bionetwork. The user will be able to traverse through the world in a safari-like experience where they can get close to animals and see their behaviours in a much more personal and realistic manner as if they were a real-life animal zoologist. (16)

This previous year project influenced this project as it had aspects of wildlife and procedural generation in it. The approach to procedurally generate the terrain added a level of complexity to the project that inspired the method that will be used in this project to create the terrain.

2.5.2. Project No. 2

Title: How Games Can be Augmented for Inclusion
Student: Max Blennerhassett

This project attempts to address the problem of games being inaccessible to sensory impaired individuals, particularly those with eyesight impairments. The game created for this project is named “Pentris”. Pentris is a game focused on accessibility, it can allow far more focus for accessibility rather than actual gameplay or narrative. Depending on the game’s design, particularly the genre it classes itself as, a game may be far more limited in the level of accessibility functionality that can be realistically implemented without changing the core design of the game. (17)

This previous year project influenced this project as it tries to address an issue with games being inaccessible to a particular user. In the case of this project the aim is to address the issue of normal educational techniques having low interaction , particularly with students.

2.6. Conclusions

This Section of this report explores technologies surrounding this project. The relevant research was considered, described, compared, and decided on whether it was suitable for the system being created. Alternative solutions to this project were explored. Games engines, programming languages and modelling tools were also looked at. Other research that was done such as animation techniques, AI and assets were explored. Finally, previous year projects were then considered.

3. Design

3.1. Introduction

In this section, all aspects concerning the design of this project will be discussed. The Project setup, software methodology, and system architecture is discussed. The requirements are gathered and then the front end is created. Two medium fidelity prototypes are shown along with multipole use cases. The system design is then discussed shown the backend of the feature for this system. Finally, the models and the audio are presented. (29)

3.2. Version Control and Setup

The first step in starting this project is setting up a GitHub Repository for Version Control. This means the work that is done on the project will be backed up and saved online. This will prevent loss of data. in terms of compatibility, GitHub is the perfect application to use with Unity as it is easy to push and pull data.

TeamGantt and Milanote were used for project management tools. These were used to document any ideas for the project and to plan out each step of the process. This made it easy to track the progress of the project and keep up with the deadlines.

3.3. Software Methodology

When considering the different design methodologies for this project, there are several choices. Three main options that were considered were the ADDIE model, Agile Design and Design Thinking. (18)

Agile Design

This method is used for breaking up tasks and decreasing the amount of planning. These tasks generally have short time frames that will only last a few weeks. This methodology lets the designers work with the other team members and gets feedback by showing the customers the progress. The customer interaction is a big part of this model.

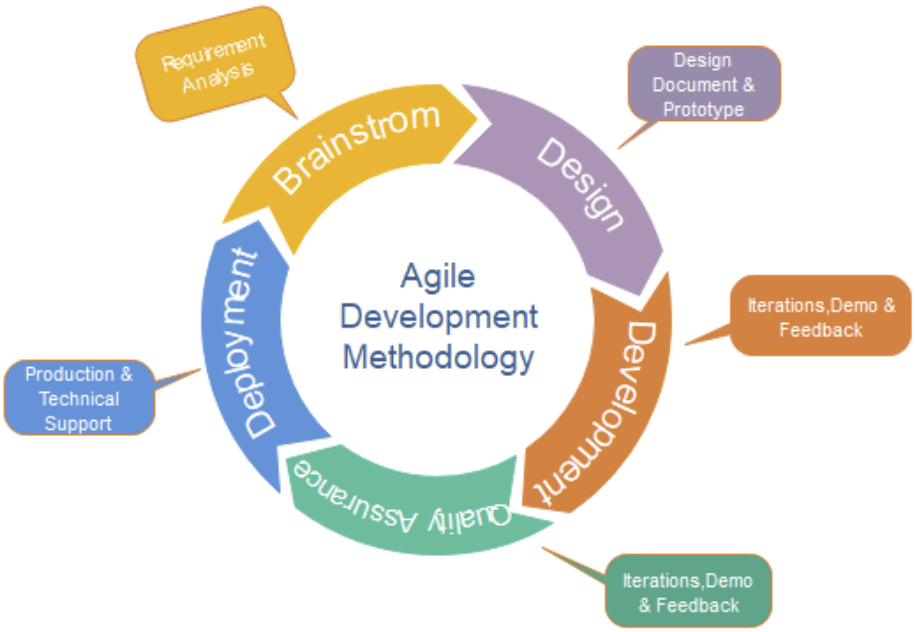


FIGURE 22-AGILE MODEL

Design thinking

This model is used to solve complex problems and develop ways to solve these problems. The focus of this model is on the problem and product. However, this model also uses customer input to solve these problems as well.

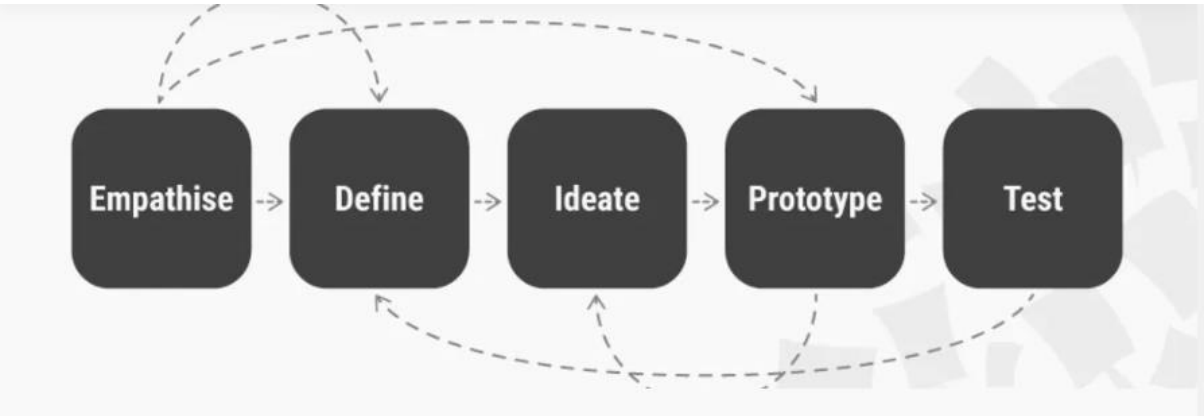


FIGURE 23-DESIGN THINKING MODEL

ADDIE Model

The Final methodology and the Chosen model that will be used in this project is the ADDIE model. This model is mainly used in educational applications for instruction and learning. It is the most common model for instructional design, and it is quite basic but very effective.(19)

There are 5 Main Stages to the ADDIE Model:

- Analysis
- Design
- Development
- Implementation
- Evaluation

This model requires each stage to be done in order and then focuses on reflection and iteration.

The reasons behind selecting this methodology were because the ADDIE is perfect for instructional design which this game focuses heavily upon. The model is the most common model and has been used for a long time proving its reliability. It creates clear learning objectives and well-structured content whilst also putting emphasis on assessment which helps with leaning outcomes.

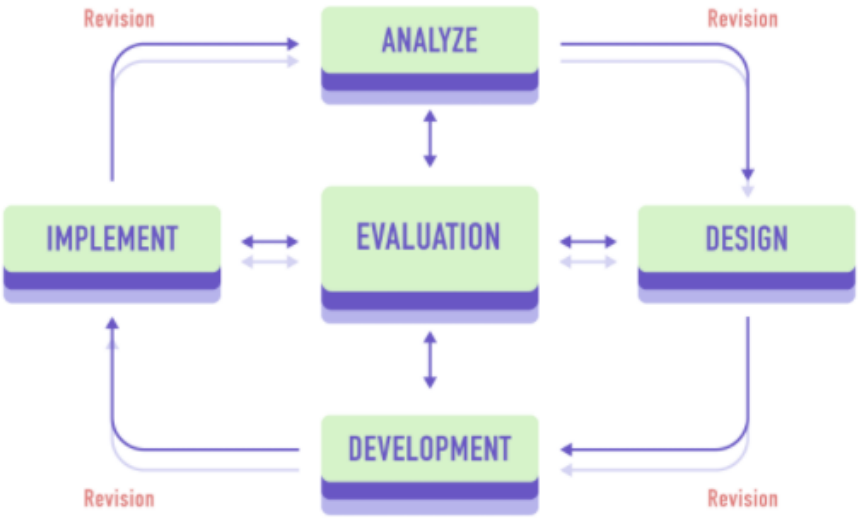


FIGURE 24-ADDIE MODEL

3.4. Overview of System

3.4.1. Architecture

The architecture of the system will organise the components into layers. All the components will be on the user's computer so the architecture will be single tiered. The Unity System architecture was used. This shows how all the different components and different software of the system will communicate.(20)

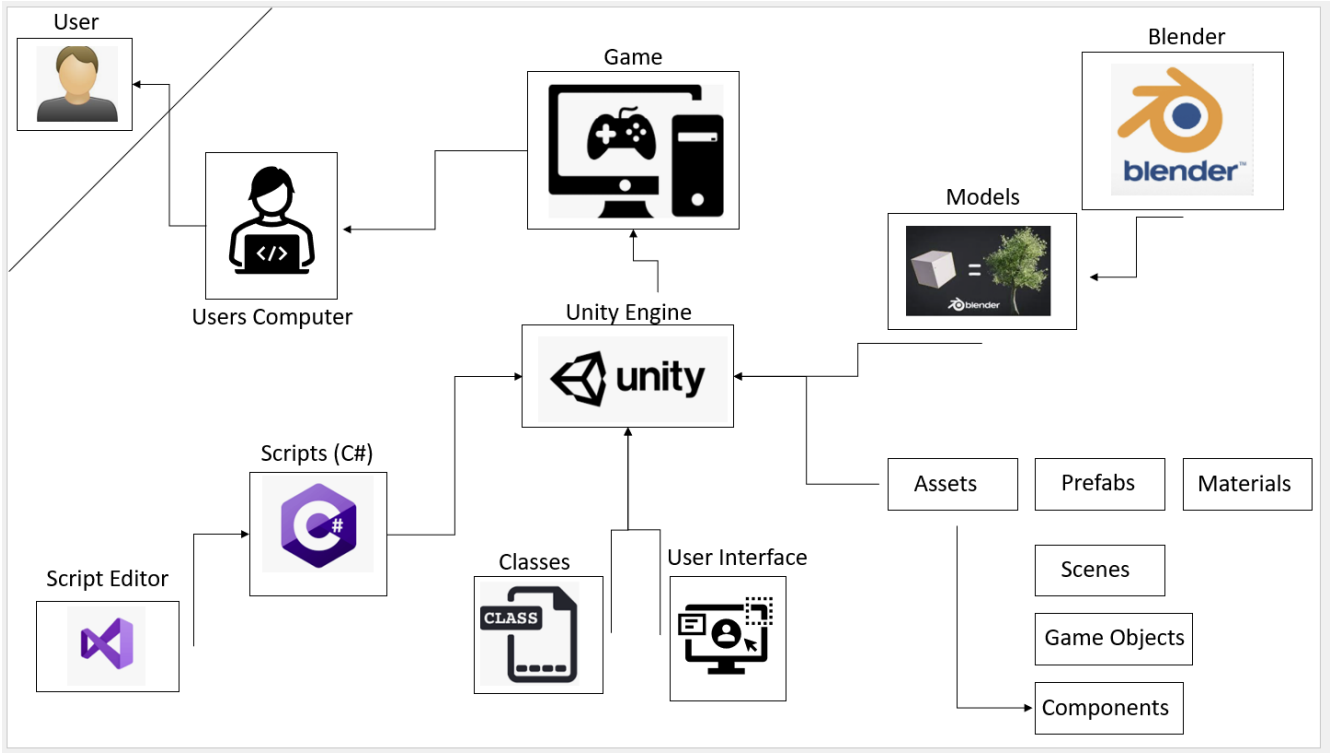


FIGURE 25-SYSTEM ARCHITECTURE

3.4.3. Requirements

When designing and developing this system, requirements that satisfy the systems physical and functional needs will be needed. There are many requirements for this system. The main requirements for the system will be listed below which will be broken down into specific tasks using a requirements table.

3.4.3.1. Requirements Table

The table below requirements for this system. The table contains the task ID, the task, the priority of that task, and a brief description of the requirement.

The three different levels of priority are as follows:

Low : Not essential for project completion, low priority task should only be completed once all medium tasks have been fulfilled.

Medium: Hold some importance for completion of the system but should only be completed after the High priority tasks have been successfully accomplished.

High: These requirements must be completed for this system to work. They hold the most priority.

ID	Task	Priority	Description
1	Loading Screen	Medium	When launching the game, loading into the terrain, or accessing anything in the game that takes more than a few seconds to render or load, a loading screen will be displayed. This will show the user that it is loading indicating that it will take a few more seconds or minutes.
2	Start Menu	High	Start menu will be the first interactable screen displayed in the game. This screen will be needed to access all the functions of the game. Start game, options, quit game.
3	Pause Menu	Medium	Pause menu used whilst moving throughout the terrain. Gives the user the option to quit, access options, save game, and resume.
4	Cut Scene	Medium	A cut scene will play when the user starts the game, this will explain what the game is about and how to play it

5	Narrator	Low	A narrator will talk during the cutscene. Using a pre-recorded sample or Microsoft edge.
6	Terrain Generation	High	The environment in which the user will be exploring will be procedurally generated using code. Trees, rocks bushes, etc.
7	Character Controller	High	A character controller will be used to the user can move throughout the terrain and play the game.
8	Character Model	Medium	Create a character model.
9	Animal Modelling	High	Model animals inside of blender. Export them to unity.
10	Animal Rigging	High	Rig animals so they can be animated.
11	Animal AI	High	Use code to create realistic animal behaviours.
12	Plant Modelling	Medium	Model plants and trees.
13	Object Scanning	Medium	Scan objects using 3d scanner to be used as assets.
14	Database Creation	High	Use database to store information for animals and notebook feature, be able to save the game.
15	Music	Low	Add background music.
16	Audio	Low	Add audio such as animal sounds, terrain sounds.
17	Inspect Feature	High	Inspect feature allows user to inspect wildlife so they can view information on it in the notebook.
18	Notebook Feature	High	Educational feature with information on wildlife. User can access it at any time.
19	Information feature	Medium	View relevant information.
20	HUD	Low	HUD displayed on screen when user is moving throughout the terrain.
21	Quizzes	Medium	Quiz used to test user on information learned.

22	Controls	Low	Control option display in menu
23	Instructions	Low	Instruction displays in menu.
24	Download Game	Medium	Allow user to download game form the internet.
25	Tutorial	Low	Tutorial at start of the game.
26	Videos Info	Low	Videos displayed on notebook for source of information.
27	Adding to database	Medium	Adding information and wildlife to notebook/database.
28	Animal Movement	High	Animating animal movement so they can imitate realistic animal behaviour.
29	Animation	Medium	Animation.
30	Exporting Phase	High	Export and render game.
31	Testing	High	Test all system requirements.

3.5. Front-End

The front end of this project consists of menus to navigate through the game, the rest of the game is set in a 3D environment which will be demonstrated through the users view.

3.5.1. Prototypes

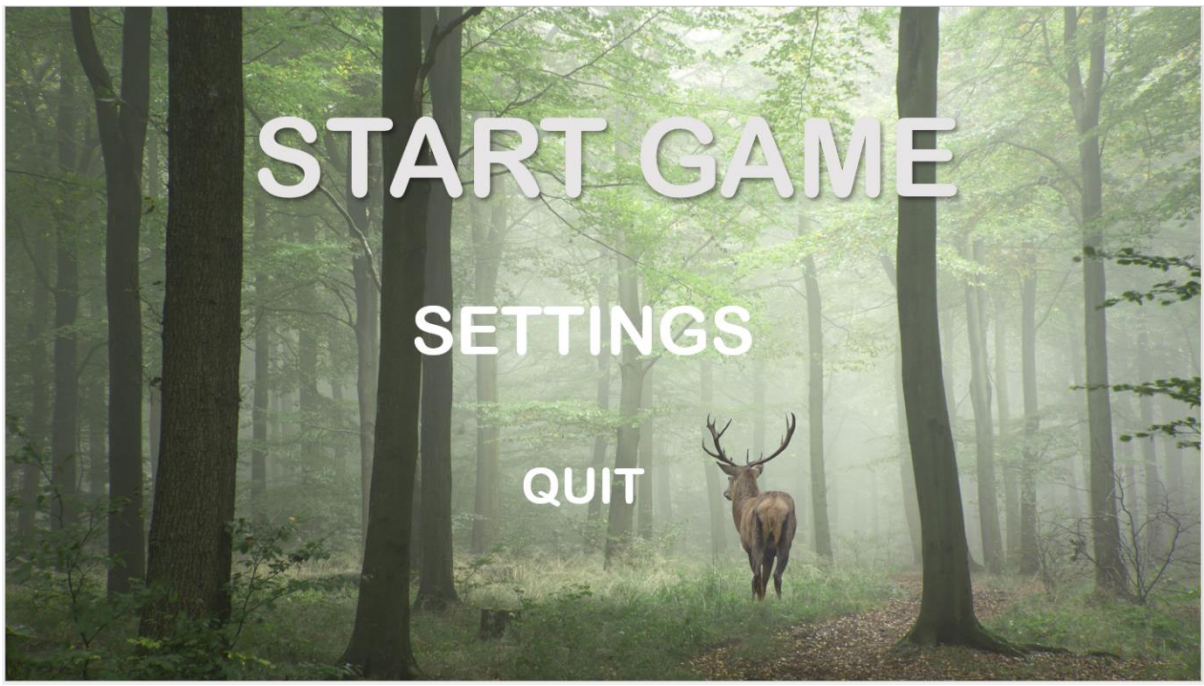
When designing the UI for the game. Two prototypes were made.

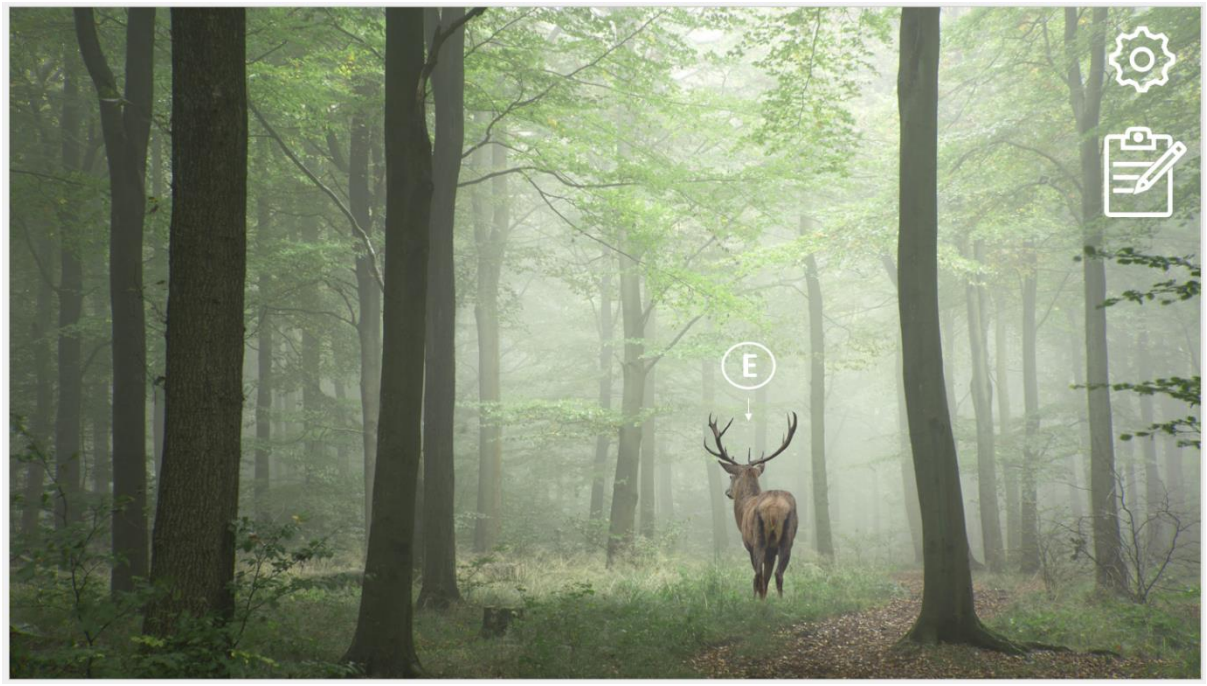
For the first iteration of prototypes, Screen layouts were made on PowerPoint. These prototypes were made to plan out the design process of the game and show what it would look like before making it in Unity.

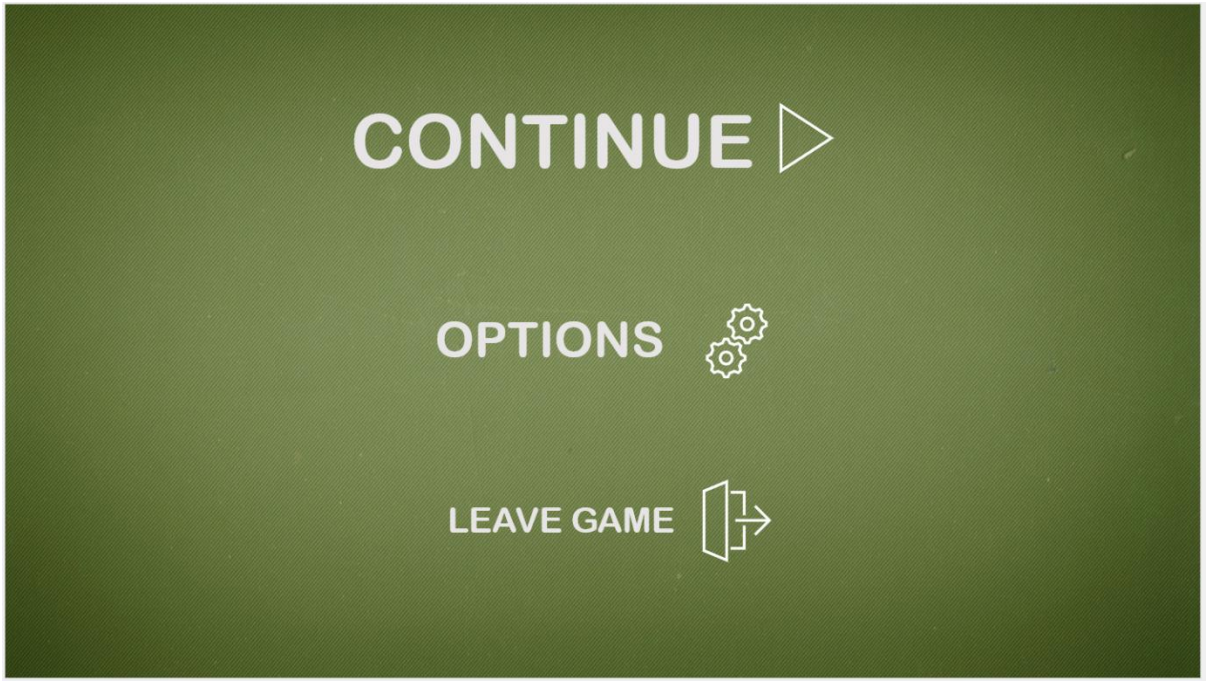
The second prototype was made after more research was done on exactly how the interfaces and game was going to work. This prototype gives descriptions and a more accurate and detailed representation of what the final product might look like.

First Prototype iteration

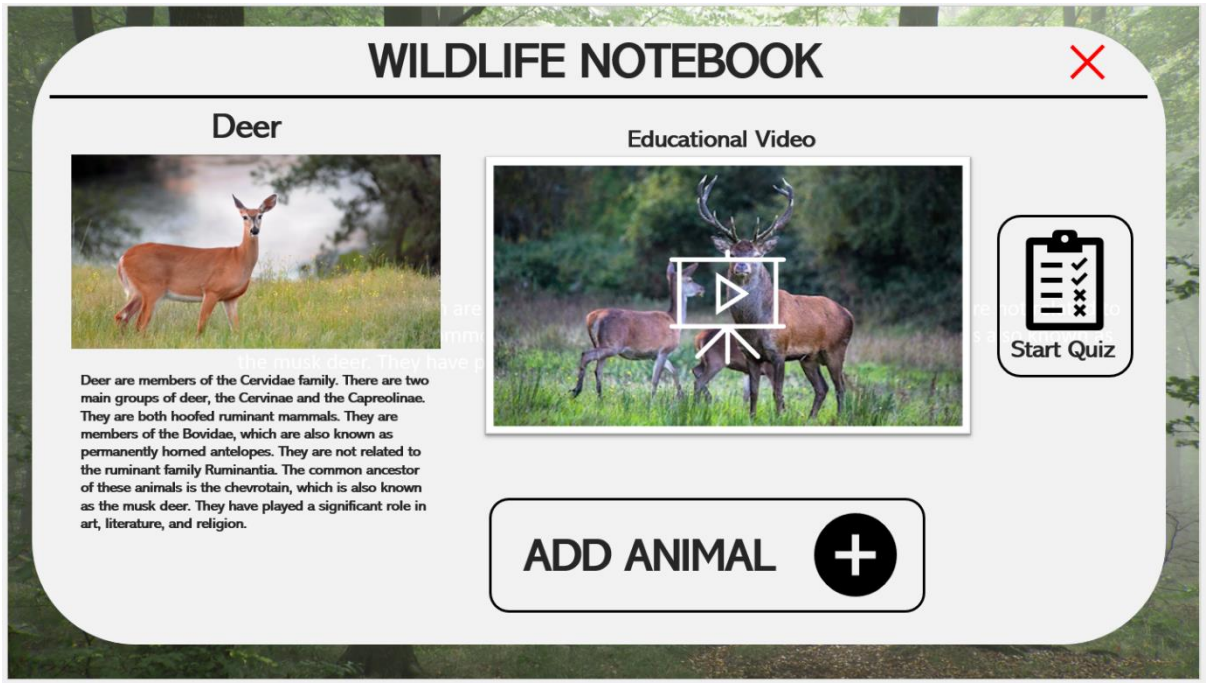
FIGURE 26-FIRST ITERATION PROTOTYPES

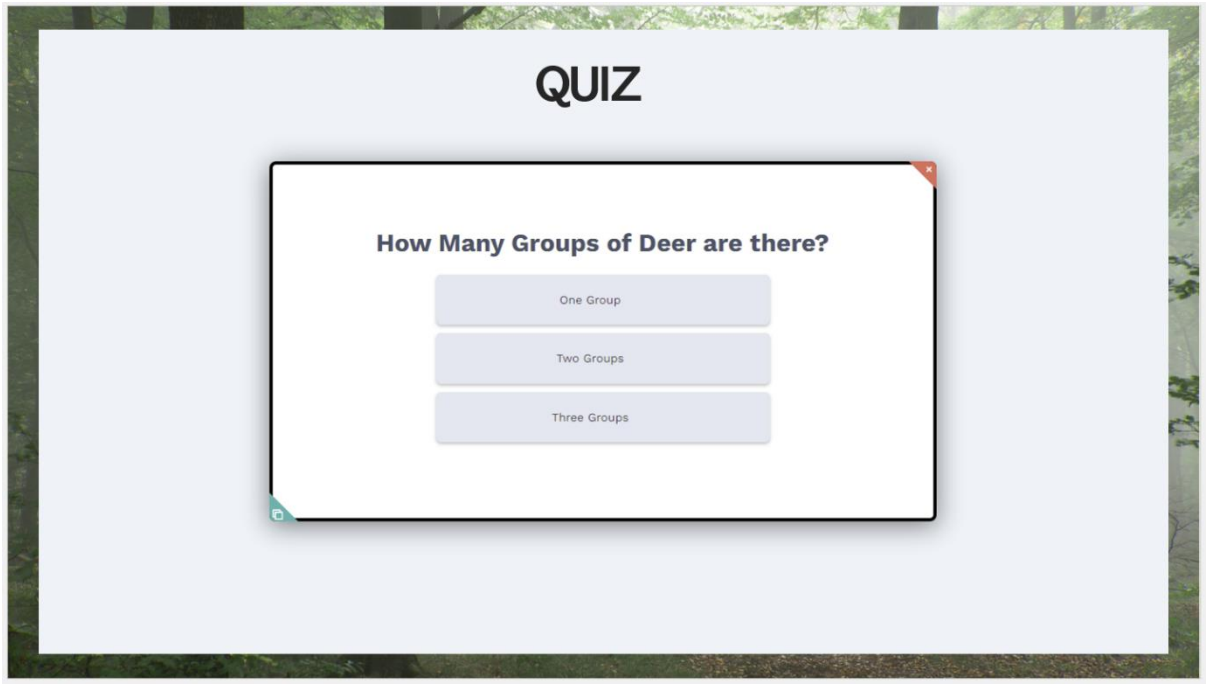


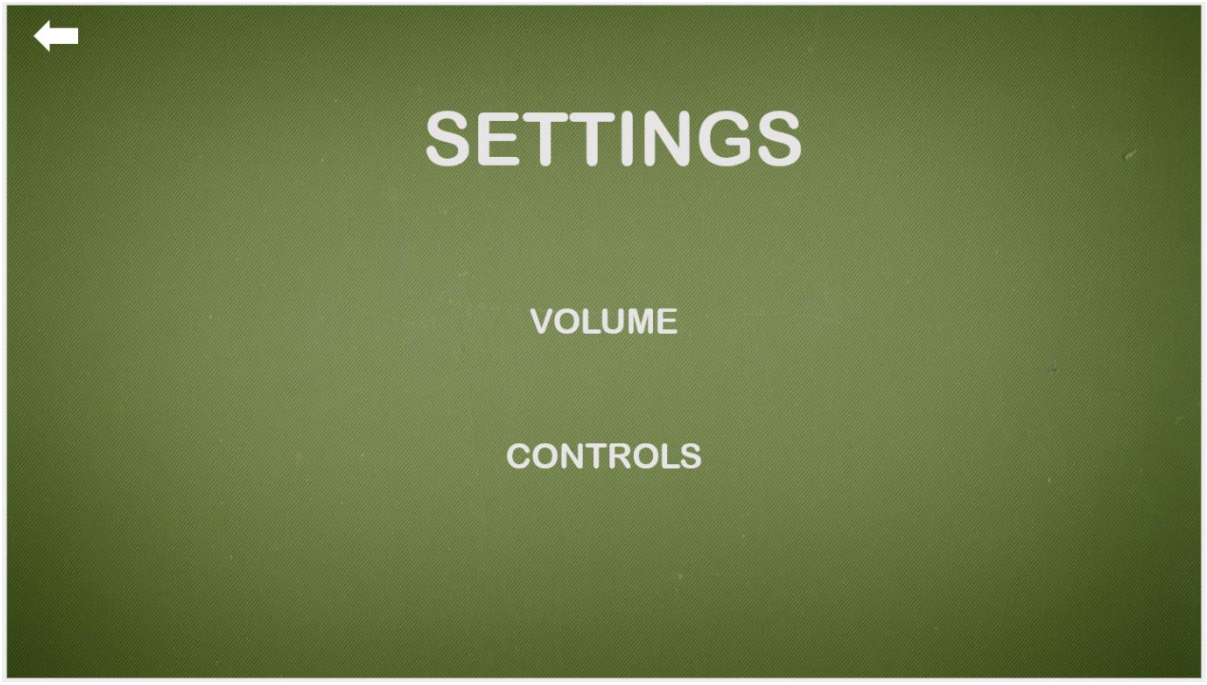

E


CONTINUE
OPTIONS
LEAVE GAME

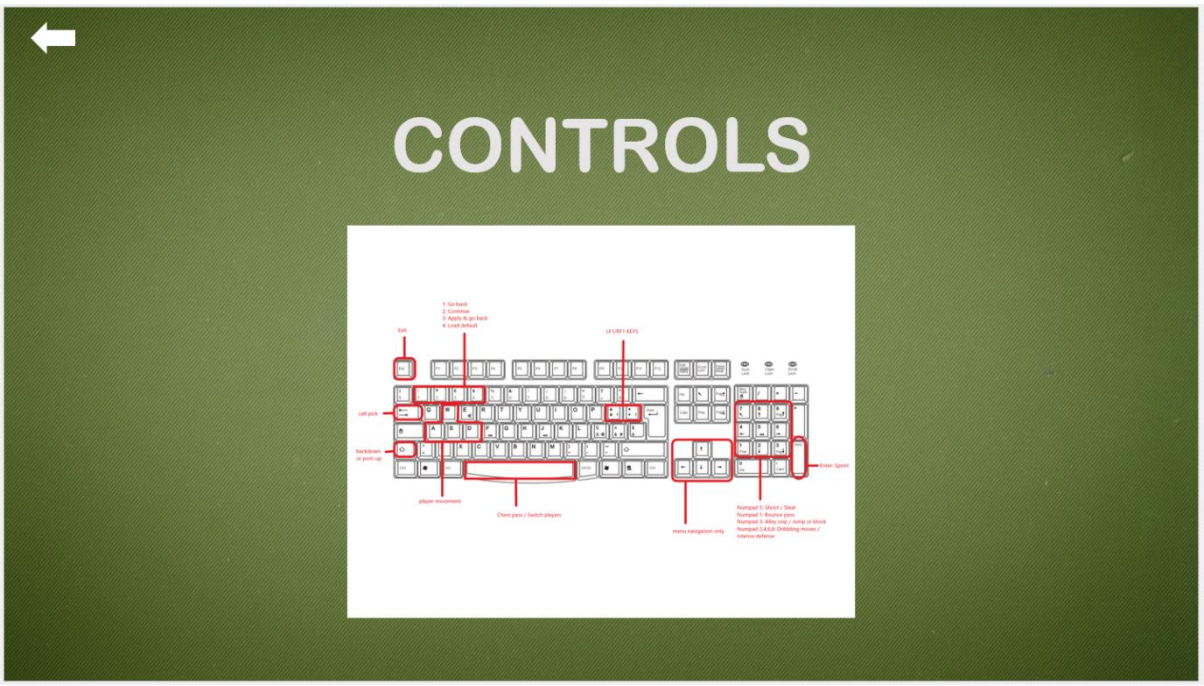

CONTINUE
OPTIONS
LEAVE GAME


WILDLIFE NOTEBOOK
Deer
Deer are members of the Cervidae family. There are two main groups of deer, the Cervinae and the Capreolinae. They are both hoofed ruminant mammals. They are members of the Bovidae, which are also known as permanently horned antelopes. They are not related to the ruminant family Ruminantia. The common ancestor of these animals is the chevrotain, which is also known as the musk deer. They have played a significant role in art, literature, and religion.
Educational Video
Start Quiz
ADD ANIMAL


QUIZ
How Many Groups of Deer are there?
One Group
Two Groups
Three Groups


SETTINGS
VOLUME
CONTROLS


LOADING...


CONTROLS

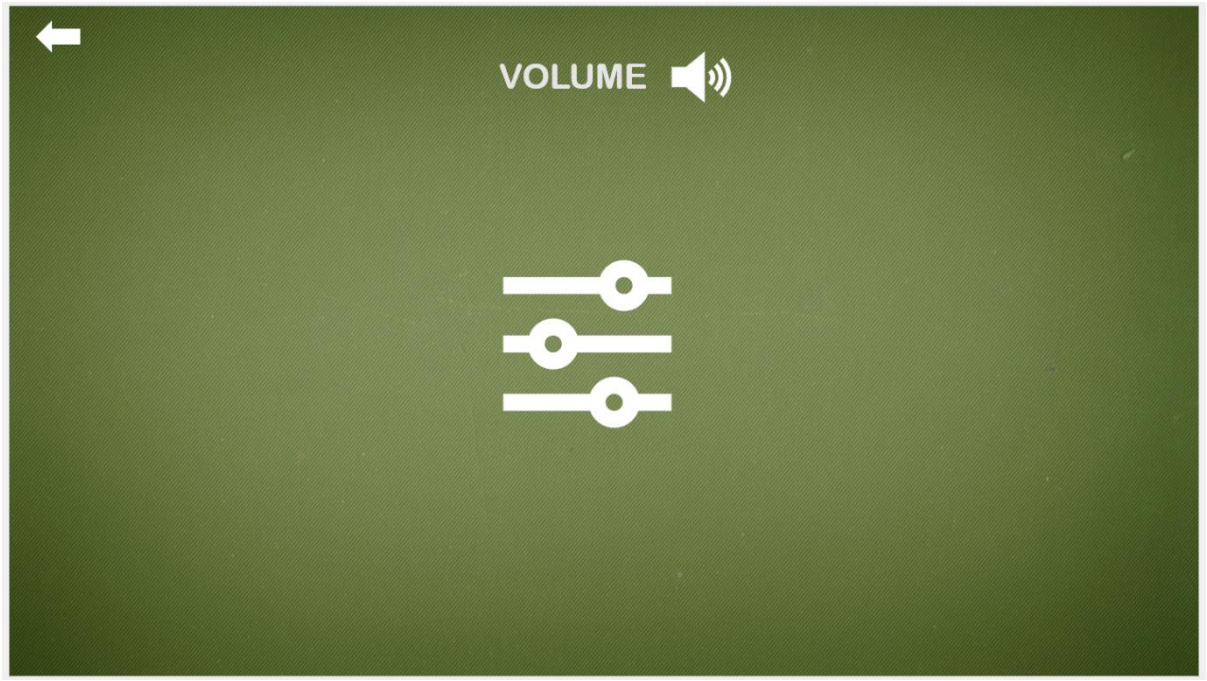


FIGURE 27-END ITERATION

Second Prototype Iteration

The second iteration of the prototype design discussed and shown below:

This prototype provides a lot more detail of how the system will work as it considers the research done for the design.

The screens will be shown as if the user is using the game in order. The first screen that the user will see is the loading screen. This screen will appear once the user has launched the game, this screen will also appear whenever the game is loading something or initializing something.

The loading screen will have a background picture from the game of a nature scene. It will then have a progress bar on the screen to show when the next scene will be loaded along with text saying, “LOADING GAME” or “LOADING ENVIRONMENTS”.

Loading Screen:

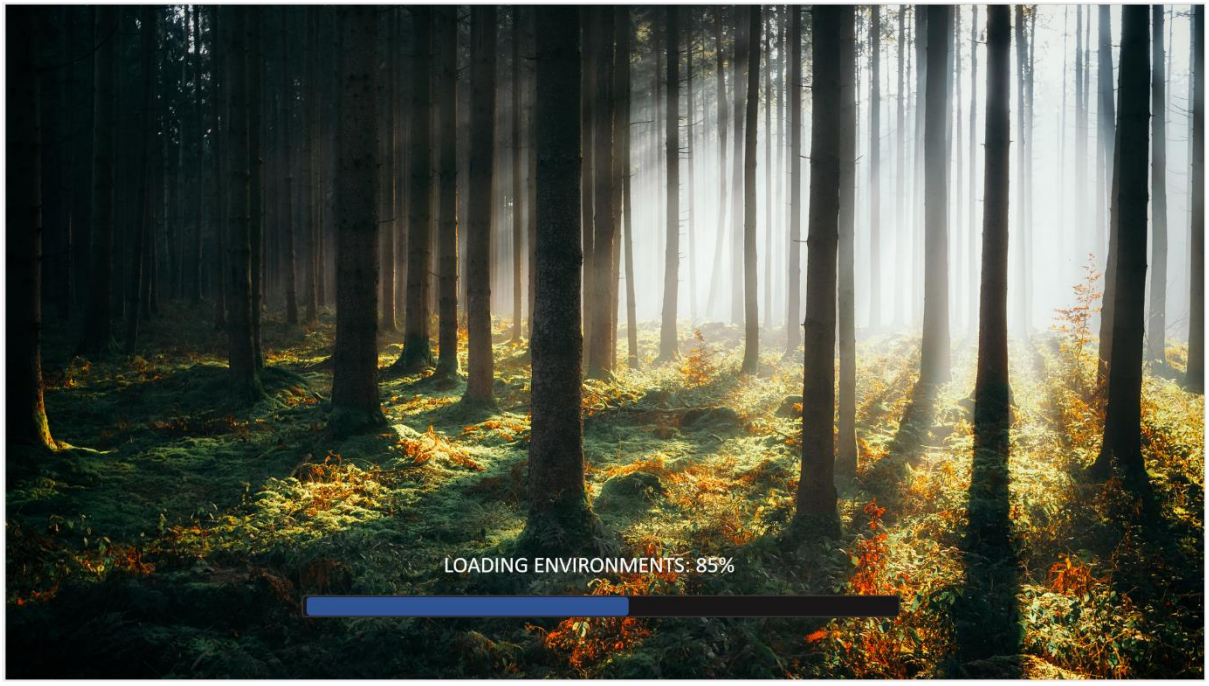


FIGURE 28-LOADING SCREEN

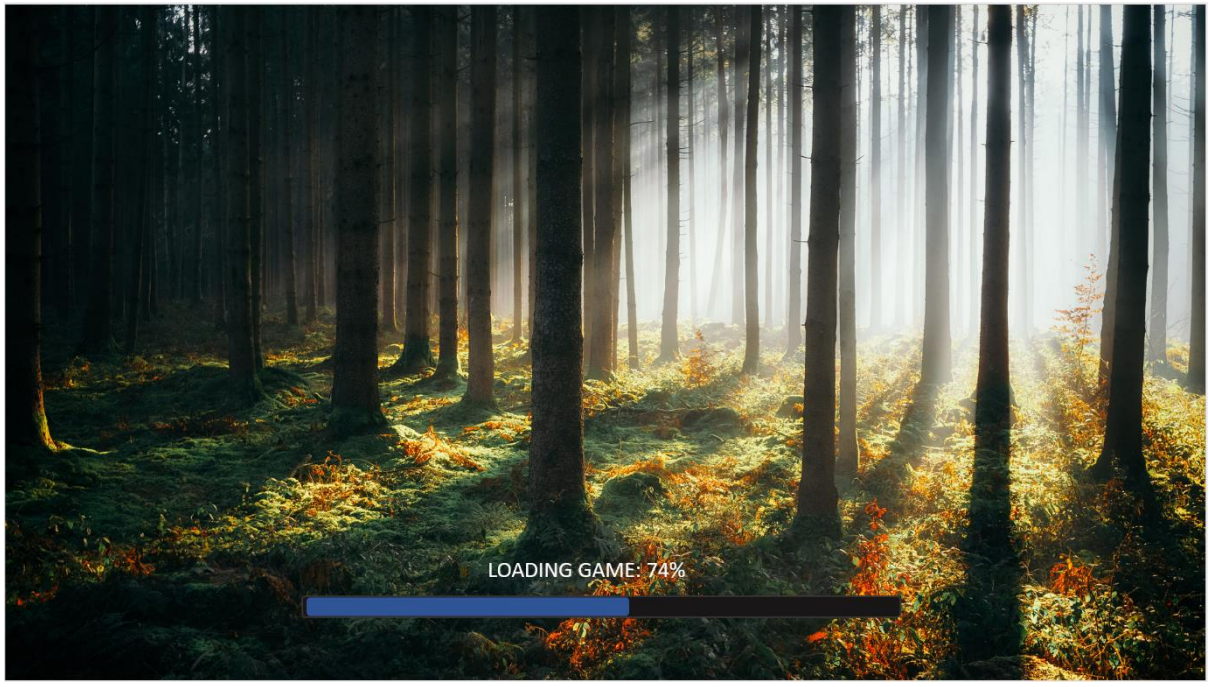


FIGURE 29-LOADING SCREEN 2

The next screen the user will see is the Start Menu screen. This screen will be the main navigation screen for the game in which the user will have three options. A “PLAY” button and load into the environment, a button to access the “OPTIONS”, and a “LEAVE GAME” button where the user can exit the application.

The start Menu will have a live background. This will consist of a scene created from the game assets with trees and plants along with a red squirrel character model from the game in the background. The squirrel will be animated.

Start Menu:

FIGURE 30-START MENU

The next screen shows the “OPTIONS” screen when the button is clicked. This will use the same background and layout style as the start menu. The “OPTIONS” menu consists of a volume adjuster and an image of the controls

The volume adjuster will allow the user to turn down the volume and the controls will show what buttons to press to use the game.

Options Screen:

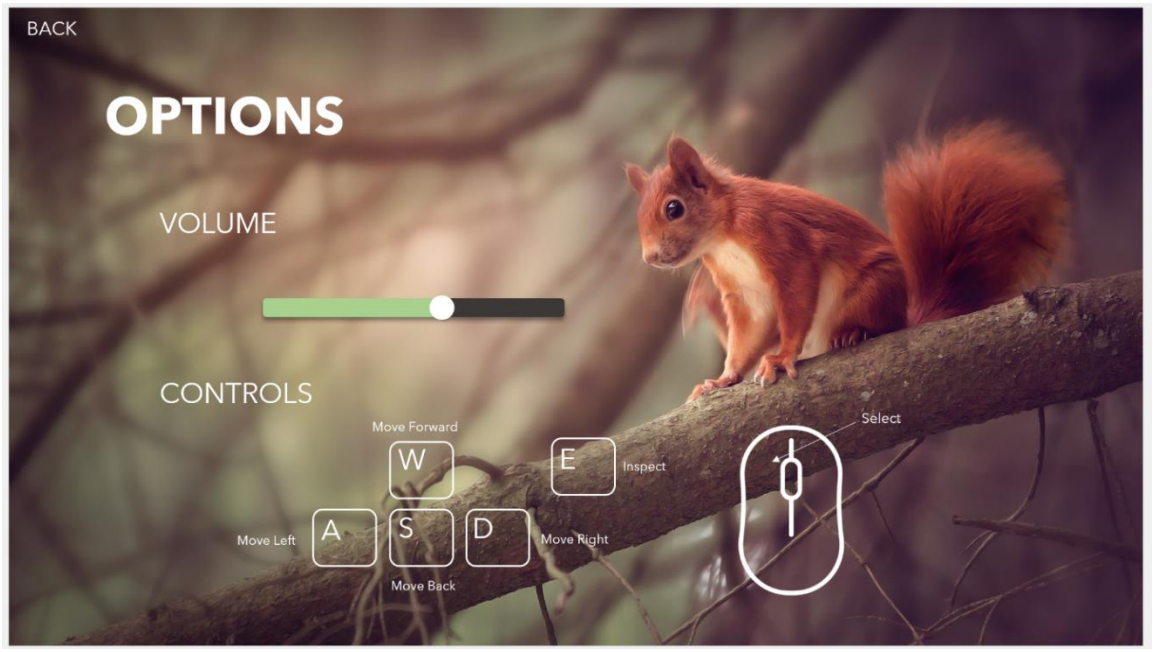


FIGURE 31-START MENU

To exit this screen the user will be able to select the back button.

When the user selects the Play button, they will load into the forest environment, where they can explore and look for wildlife. Below is a sample picture of what they will see along with the HUD. The HUD has a **pause menu** icon, an **inventory** icon, and an **inspect** icon will also appear over wildlife that the user can inspect.

Explore Screen:

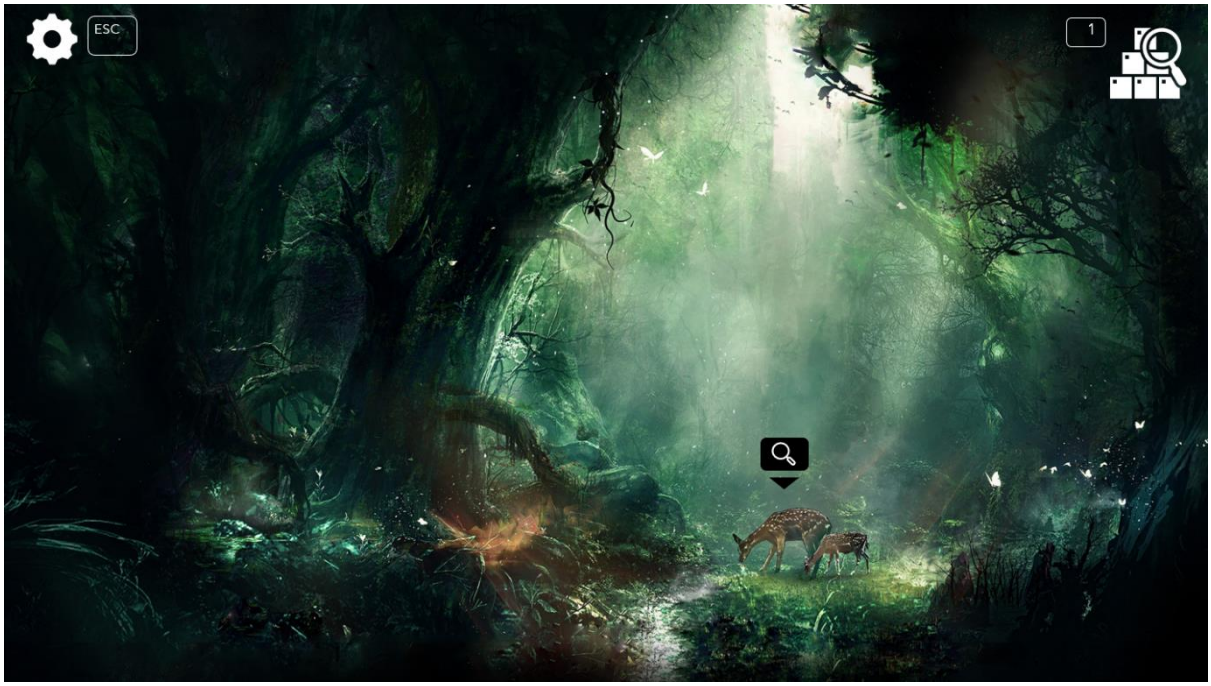


FIGURE 32-PROMPT

When the user selects the inspect option ‘E’, it will bring the user to the inspect screen which will zoom in on the object and give the user the option to rotate it and add it to their inventory or click the ‘EXIT’ button. This inspects screen will have a blurred background blurred.

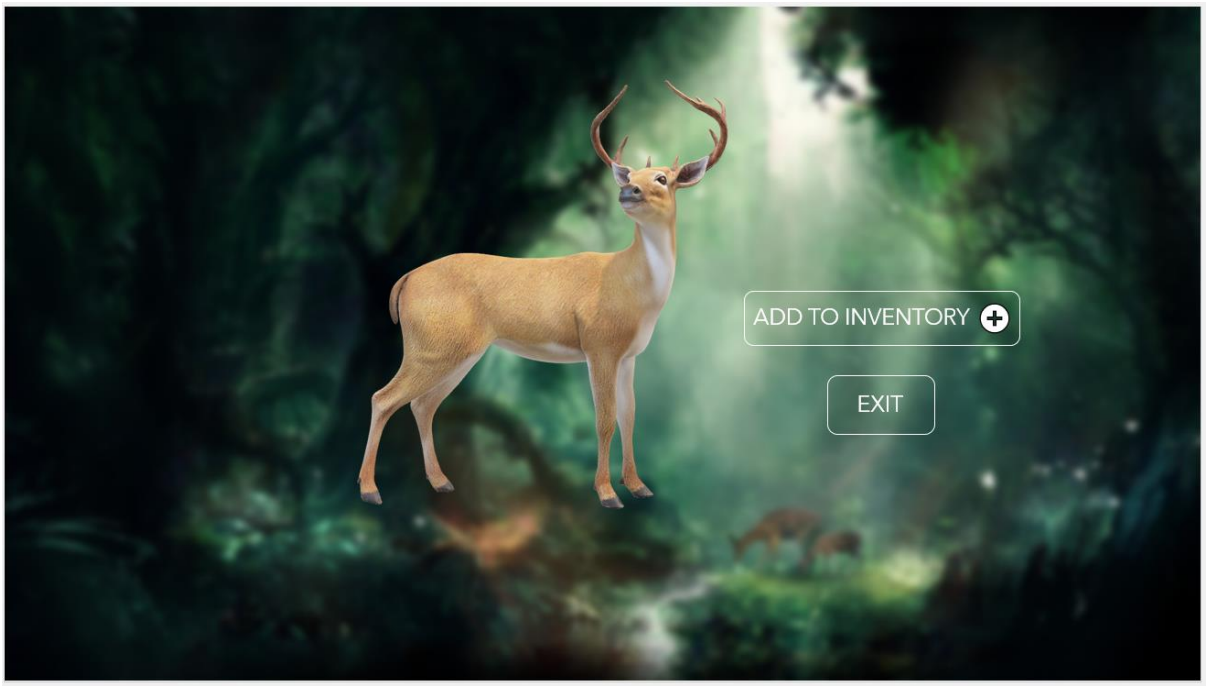


FIGURE 33-INSPECT SCREEN

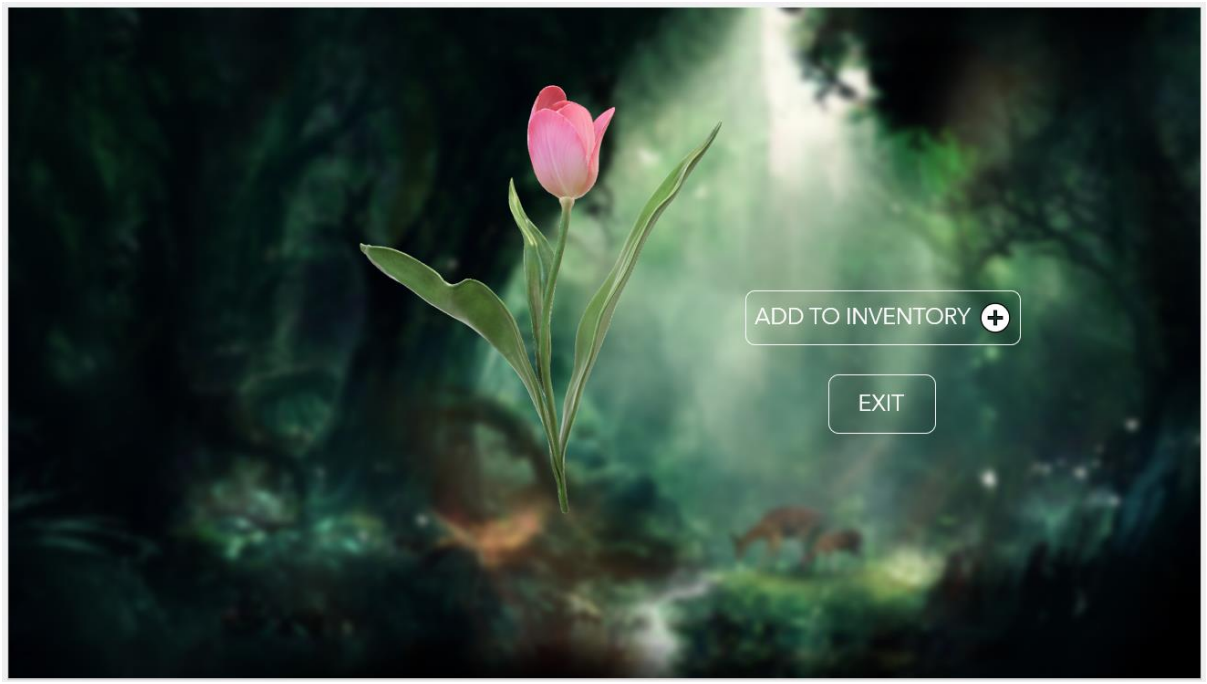


FIGURE 34-INSPECT SCREEN 2

Once you have added the animal or plant to your inventory, the user will be able to view the animal whenever the user wants to by pressing the number 1 on the keyboard to go into the inventory which will show the inventory screen with a grid of all the wildlife the user has collected, along with an 'EXIT' button.

Inventory Screen:

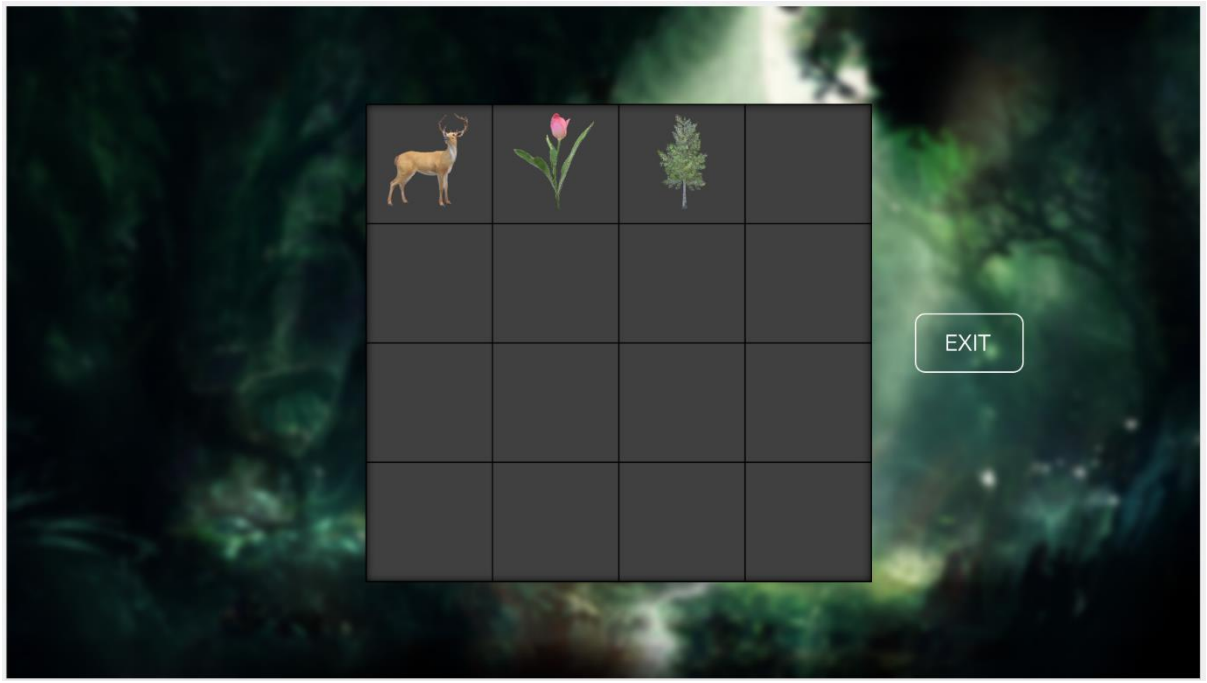


FIGURE 35-INVENTORY

Once the user is on the inventory screen, they will have the option to click in the wildlife objects they added to the inventory and display information about the wildlife. This screen will have a back button and a quiz option button.

information Screen:

FIGURE 36-INFORMATION SCREEN

When the user selects start quiz the user will be able to do a quiz based on the information of the wildlife they are looking at. The questions will be multiple choice and a score of how many the user got correct will be displayed at the end. The user will then be given the choice to retake the quiz or exit the quiz.

Quiz Screen:

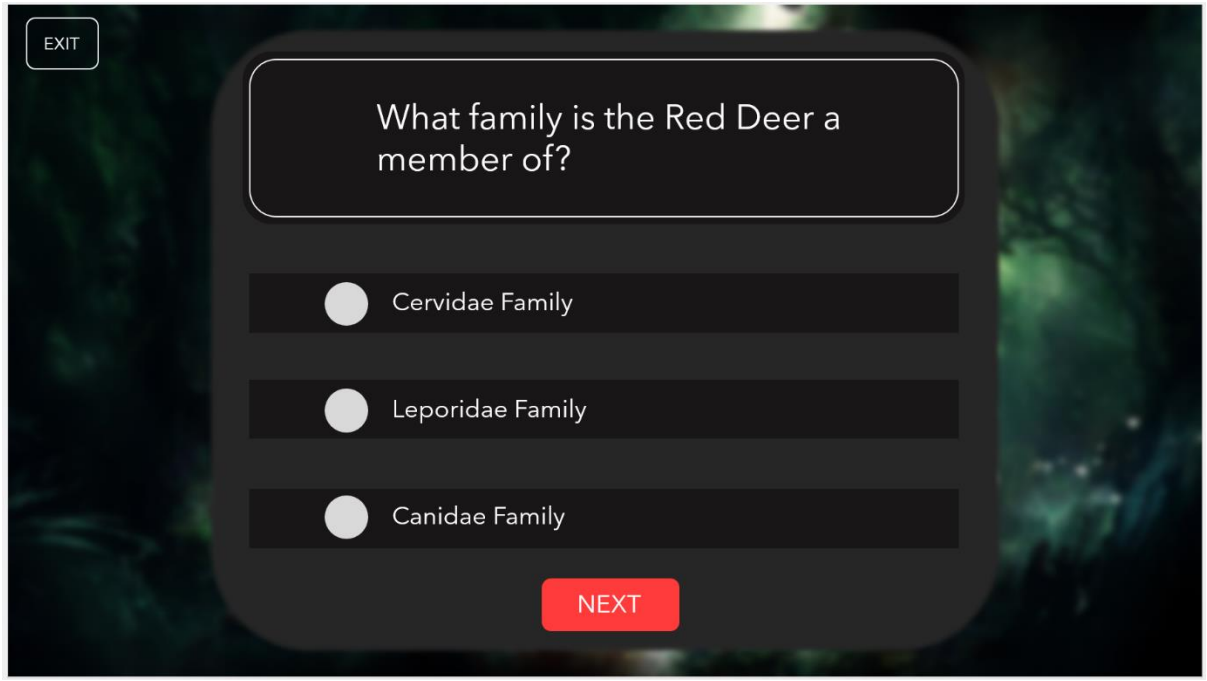


FIGURE 37-QUIZ

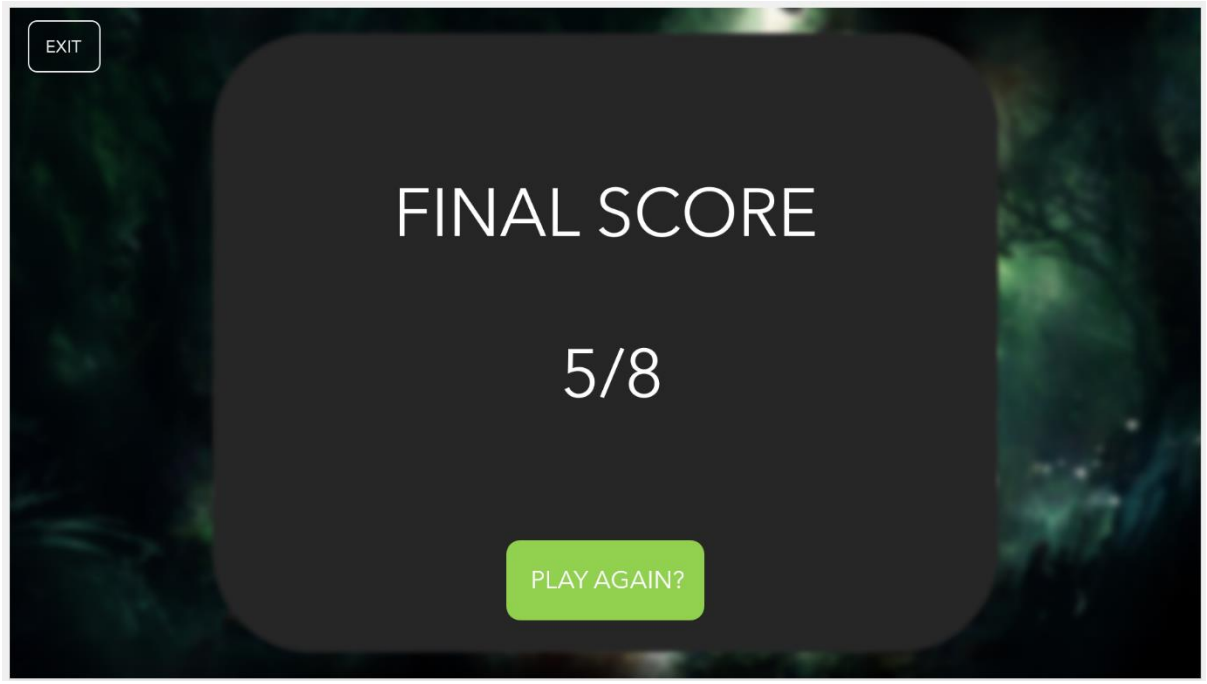


FIGURE 38-FINAL SCORE

When the user is in the game, they can access the pause menu by pressing the escape button on the keyboard or clicking the settings icon in the top left-hand corner. The pause menu will have the following options, ‘RESUME’, ‘SAVE GAME’, ‘OPTIONS’ and ‘EXIT TO MAIN MENU’.

Pause Screen:

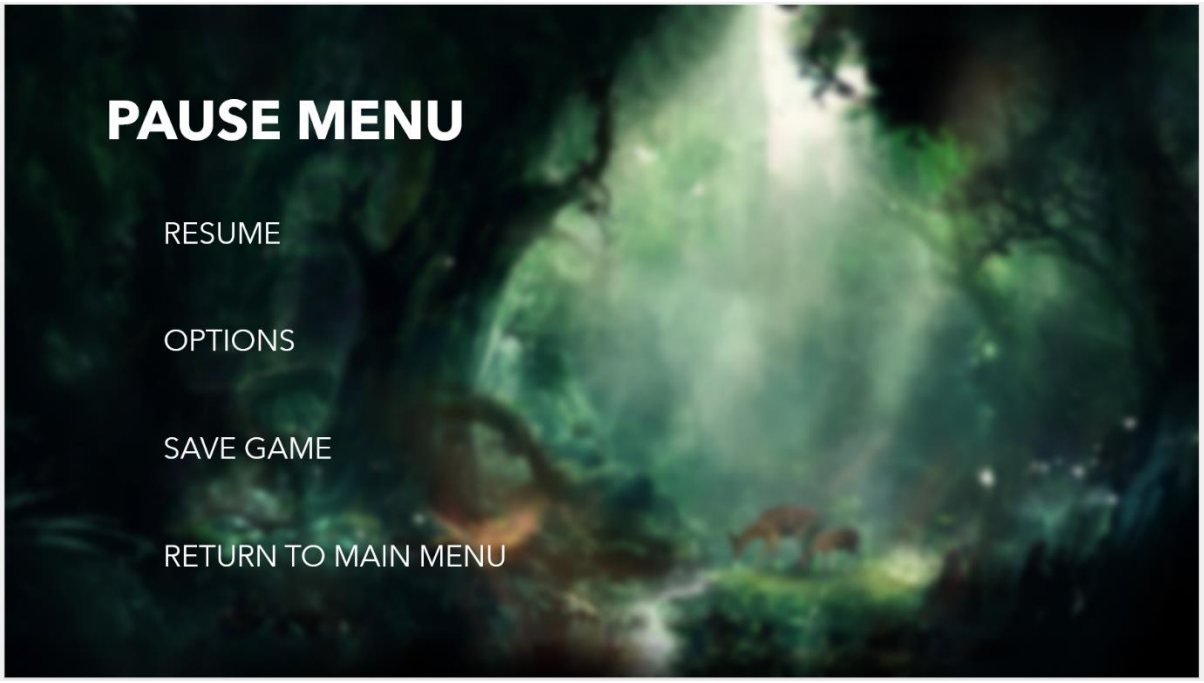


FIGURE 39-PAUSE MENU

When the user selects the options button, they will be brought to a screen that follows the same format as the main menu’s options screen, where you will be able to adjust the volume and view the controls.

The ‘RESUME’ button will bring you back to the game and the ‘RETURN TO MAIN MENU’ button will bring you back to the main menu.

Options Screen:

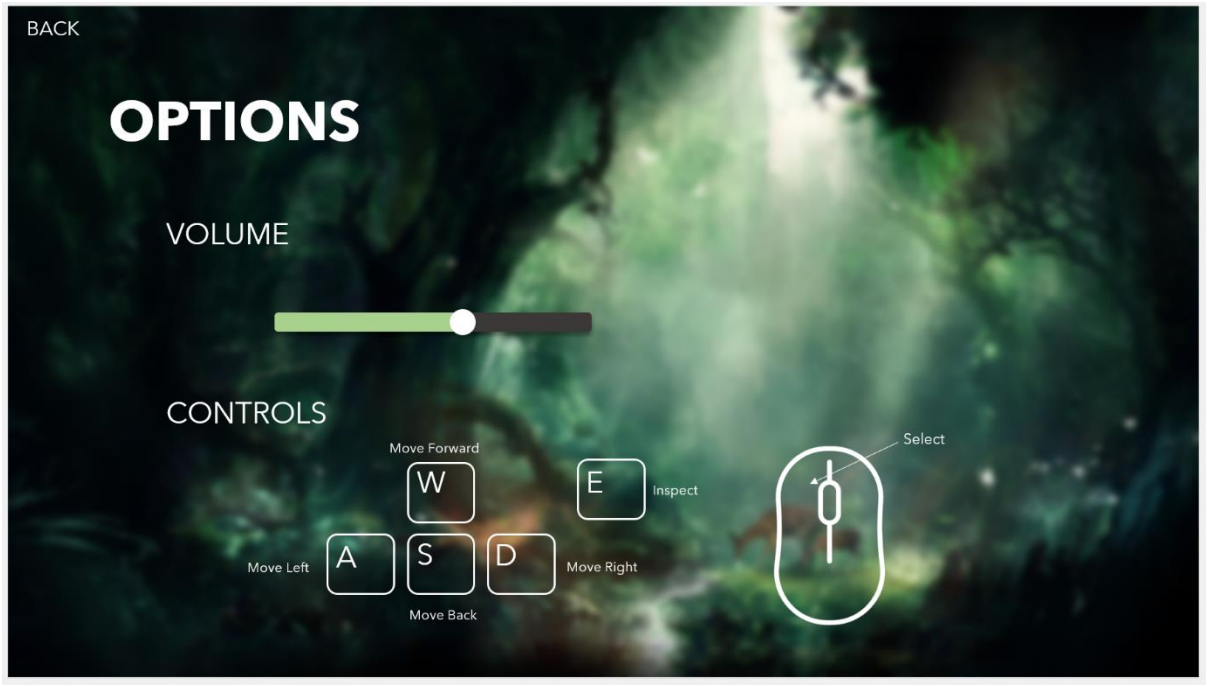


FIGURE 40-OPTIONS

If the user selects the save game option, the user will be able to save their progress and placement in the game.

Save Screen:

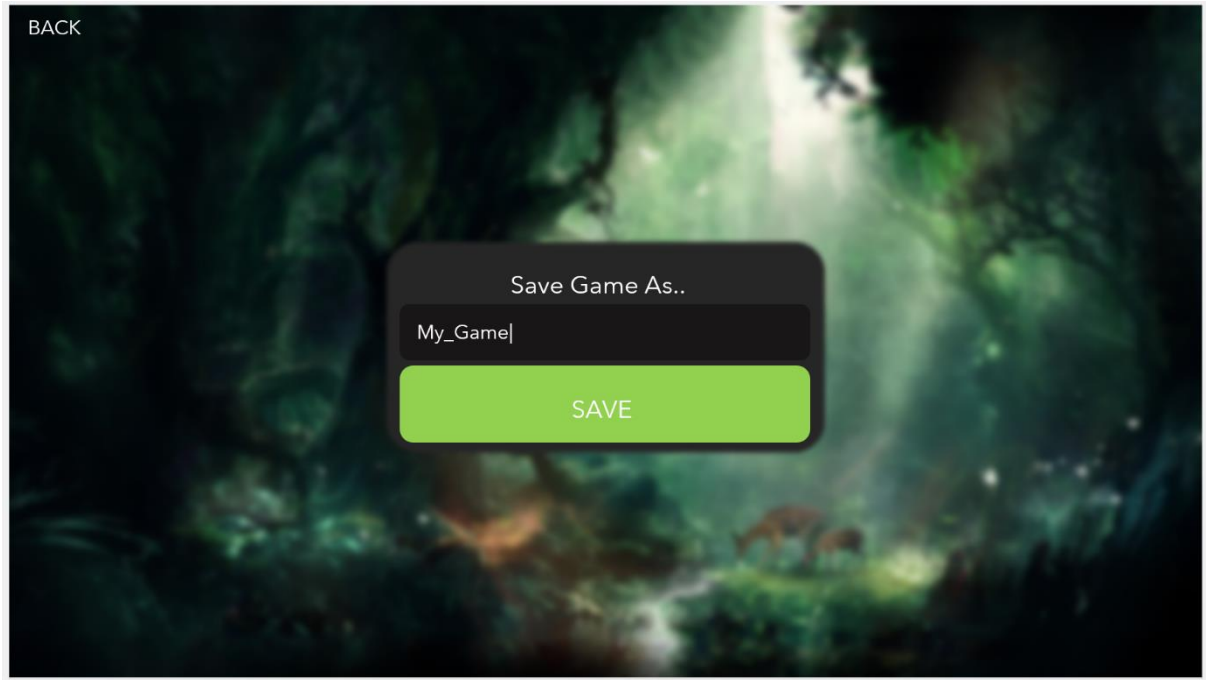


FIGURE 41-SAVE SCREEN

Now, next time the user plays the game they will be able to select the saved game they want to load.

Load Screen:

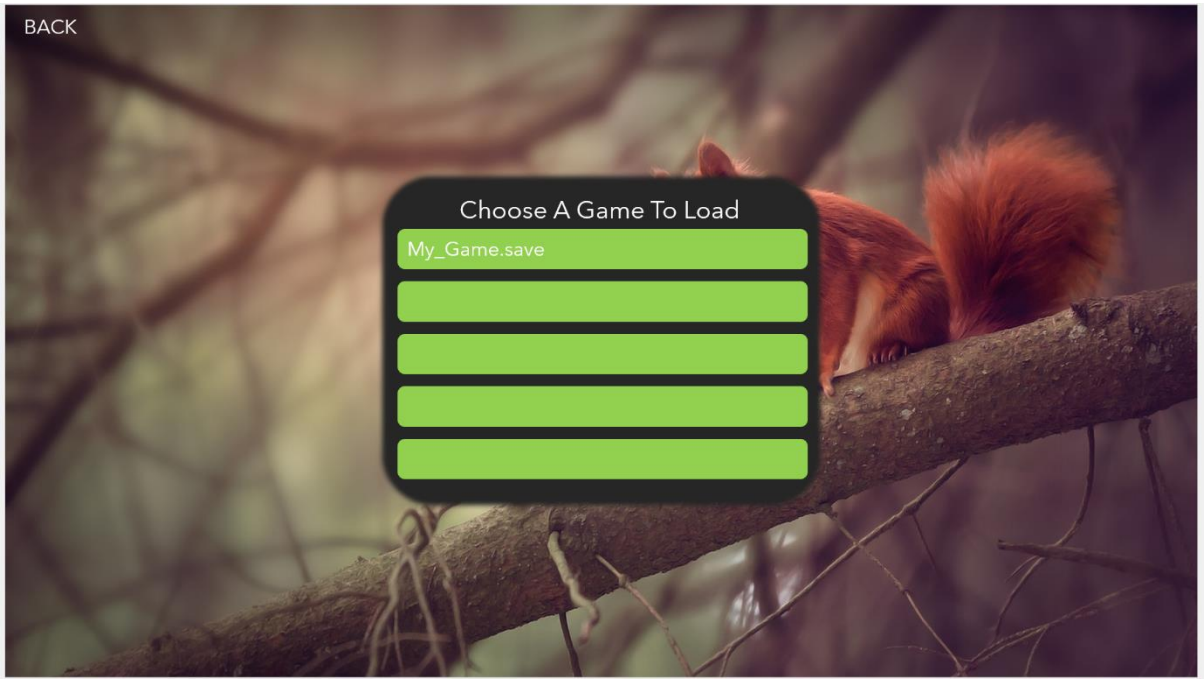


FIGURE 42-LOAD SCREEN

3.5.2. Use Cases

There are many use cases for this project. Shown below are the use cases for each main feature of the game, the first three iterations of the use case diagram will be shown that explore all the main features briefly. Each main feature of the use case will then be described in detail using use case scenarios. These scenarios will have descriptions, a goal, a main flow with steps and an error flow each.

Use Case Diagram 1st iteration

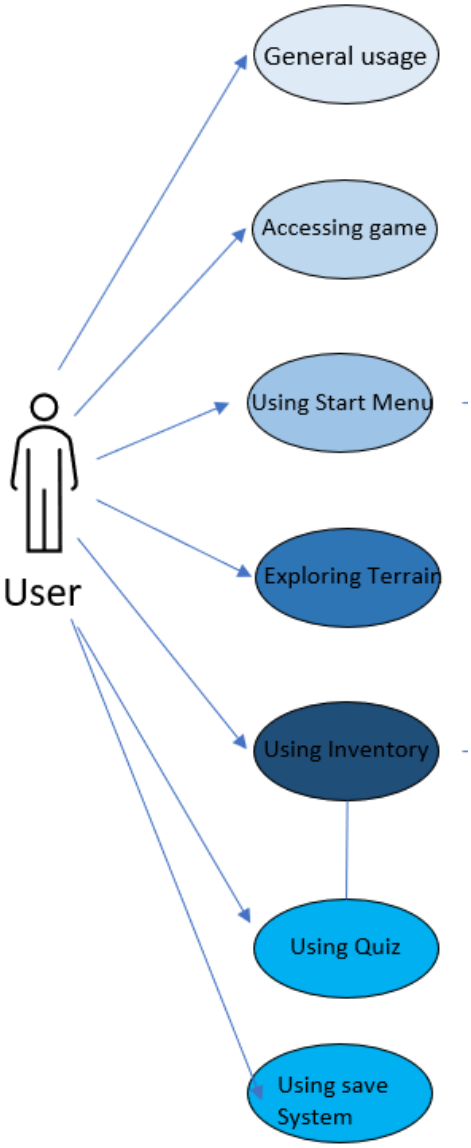


FIGURE 43-1ST ITERATION

Use Case Diagram 2nd iteration

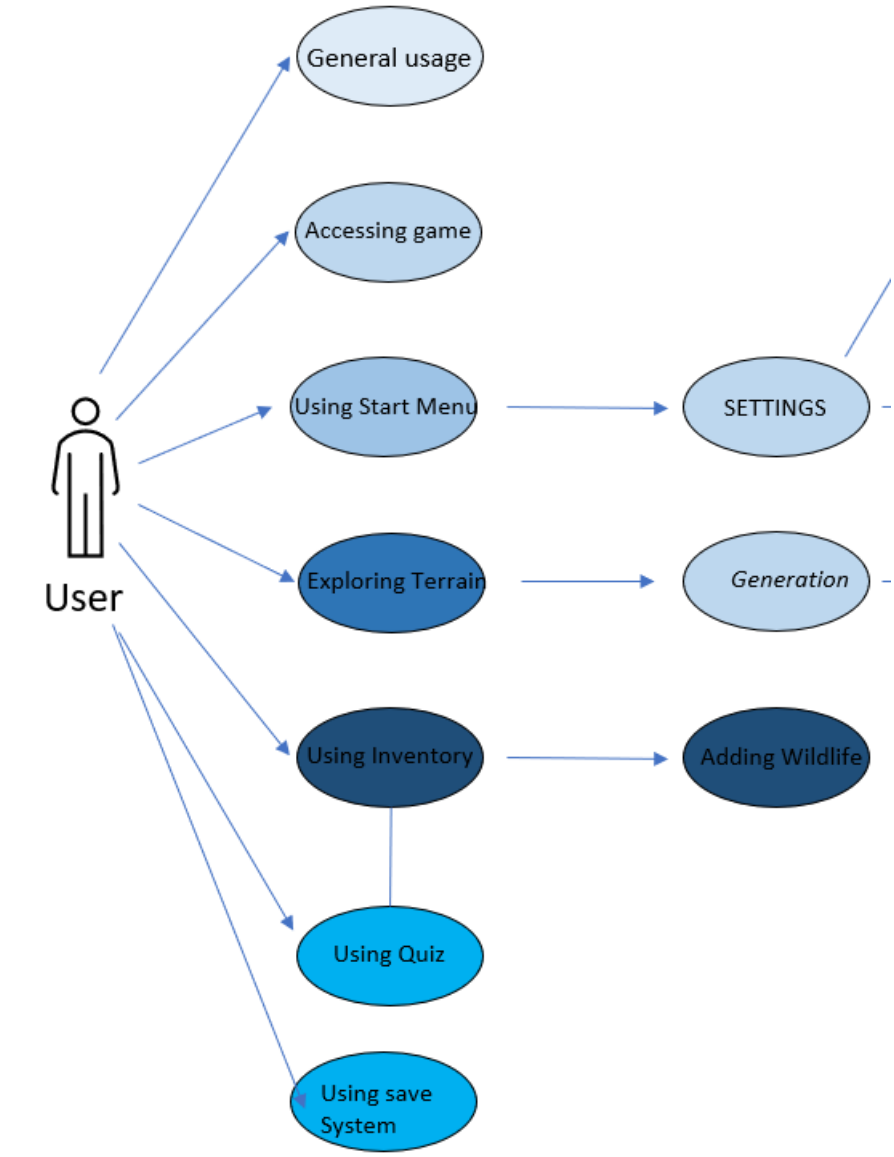


FIGURE 44-2ND ITERATION

Use Case Diagram 3rd iteration

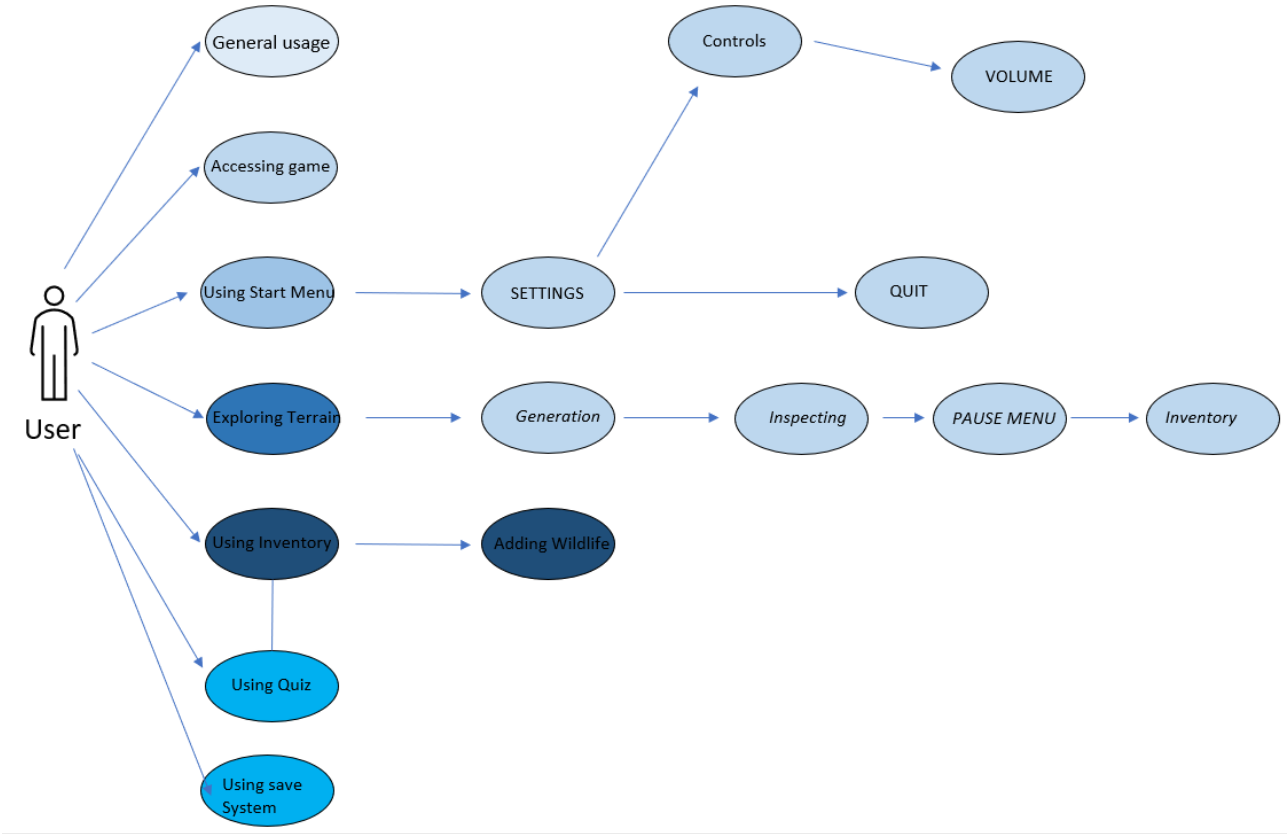


FIGURE 45-USE CASE DIAGRAM

Use Case Scenarios

USE CASE	1	General Usage of Wild-Ireland
Description of Goal in Context	Navigating through application. Briefly looking at each feature. Use game as if user for first time to get all requirements.	
Post Conditions, Success End Condition	System will work well. Each feature works successfully with no errors.	

Main Flow:		
Step	**Action**	**Alternate**
1.1	**Download Game**	**E or EF**
1.2	**Launch Game**	
1.3	**View Loading Screen**	
1.4	**View Start menu**	
1.5	**Select Settings**	
1.6	**Adjust Volume**	
1.7	**Look at Controls**	
1.8	**Select back**	
1.9	**Select Play**	**E or EF**
1.10	**View Cutscene**	
1.11	**Walk Around Terrain**	
1.12	**Inspect Animal or Plant (wildlife)**	**E or EF**
1.13	**View inventory**	
1.14	**View Video**	
1.15	**Start Quiz**	**E or EF**
1.16	**Add Animal to inventory**	
1.17	**Exit Notebook**	
1.18	**View Pause Menu**	
1.19	**Select Leave Game**	
1.20	**Select Quit in Start menu**	
EXCEPTIONS or ERROR Flow Description		
Step	**Branching Action**	
1.1.1	**Download Failed due to users memory full.** **User Clears memory and retrys download.**	
1.9.1	**Game not loading in. Cutscene having trouble playing**	

	Error message shown, user exits game and restarts.	
1.12.1	**Inspect prompt not showing.** **User moves closer or tries other animal.**	
1.15.1	**Quiz not showing questions** **Error message displayed and user restarts quiz.**	

USE CASE	**2**	**Accessing Game**	
Description of Goal in Context	**User will be able to download game, Install the game on their computer. Launch the game.**		
Post Conditions, Success End Condition	**Download goes smoothly. Each feature works successfully with no errors.**		
Main Flow:			
Step	**Action**		**Alternate**
2.1	**Go to download link**		**E or EF**
2.2	**Click downlload link**		
2.3	**Download Game**		**E or EF**
2.4	**Install game**		
2.5	**Click icon**		
2.6	**Launch Game**		
2.7	**Load into game**		**E or EF**
2.8	**View Start Menu**		
EXCEPTIONS or ERROR Flow Description			
Step	**Branching Action**		
2.1.1	**Users PC does not meet game requirements**		

	Access on different computer.	
2.3.1	**Memory Full.** **Delete data and retry download.**	
2.7.1	**Error loading into game.** **Restart game and try loading in again.**	

USE CASE	**3**	**Using Start Menu**
Description of Goal in Context	**User will be able navigate through menu features. All features will work successfully.**	
Post Conditions, Success End Condition	**Navigation goes smoothly. Successful with no errors.**	
Main Flow:		
Step	**Action**	**Alternate**
3.1	**Launch Game**	
3.2	**View start menu**	
3.3	**Click options**	
3.4	**Click controls**	
3.5	**View controls**	**E or EF**
3.6	**Press back**	
3.7	**Click volume**	**E or EF**
3.8	**Adjust volume**	
3.9	**Press back**	
3.10	**Press start game**	
3.11	**Press quit game**	
EXCEPTIONS or ERROR Flow Description		

Step	Branching Action	
3.5.1	Controls don't match Change controls.	
3.7.1	Volume on wrong settings	

USE CASE	4	Exploring Terrain/Environment
Description of Goal in Context	User will be able navigate/move through environment. All features will work successfully.	
Post Conditions, Success End Condition	Exploration goes smoothly. Successful with no errors. Inspecting, generation, notebook, Pause Menu.	
Main Flow:		
Step	Action	Alternate
4.1	Press start game	
4.2	Load into environment	E or EF
4.3	Move throughout terrain	
4.4	Find wildlife	
4.5	Click prompt	
4.6	Inspect animal/plant	
4.7	View notebook	
4.8	View information	
4.9	Press back	
4.10	Explore more	
4.11	Press save	E or EF
4.12	Inspect more objects	
4.13	Interact with animal AI	

4.14	**Press Pause Menu**	
4.15	**Select exit to main menu**	
EXCEPTIONS or ERROR Flow Description		
Step	**Branching Action**	
4.2.1	**Environment Fails to generate** **Make procedural scripts more efficient**	
4.11.1	**Game fails to save.** **Error message Prompts.**	

USE CASE	**5**	**Using Inventory**	
Description of Goal in Context	**User will be able to access information on the inventory. All features will work successfully.**		
Post Conditions, Success End Condition	**inventory feature are displayed and functional. Successful with no errors. Videos, paragraphs, pictures, Quiz, exit, add feature.**		
Main Flow:			
Step	**Action**		**Alternate**
5.1	**Press button for inventory**		
5.2	**View info on wildlife**		
5.3	**Read text**		
5.4	**View Pictures**		
5.5	**Watch videos**		**E or EF**
5.6	**Add species to inventory**		
5.7	**Succesfully added prompt**		**E or EF**
5.8	**Click quiz**		
5.9	**Partake in quiz**		
5.10	**Press done**		

5.11	**Press exit notebook**	
5.12	**Continue exploring**	
EXCEPTIONS or ERROR Flow Description		
Step	**Branching Action**	
5.5.1	**Videos wont load.** **Error message displayed.**	
5.7.1	**Failed to add to database.** **Animal not added.** **Displays error prompt to retry.**	

USE CASE	**6**	**Using Quiz**	
Description of Goal in Context	**User will be able to access quiz in the notebook and test themselves on the knowledge they have learnt. All features will work successfully.**		
Post Conditions, Success End Condition	**Notebook feature are displayed and quiz is functional with no errors.. Successful with no errors. Answer questions , display score, re-take quiz.**		
Main Flow:			
Step	**Action**		**Alternate**
6.1	**Press button for Notebook**		
6.2	**Press quiz button**		
6.3	**Accessing quiz**		
6.4	**Start quiz**		
6.5	**Answer questions**		**E or EF**
6.6	**Tick boxes**		
6.7	**Click next question**		
6.8	**Submit quiz**		**E or EF**
6.9	**Prompt message**		

6.10	**Press done**	
6.11	**View score**	
6.12	**Re-take quiz**	
6.13	**Click done**	
6.14	**Exit quiz**	
6.15	**Return to game.**	
EXCEPTIONS or ERROR Flow Description		
Step	**Branching Action**	
6.5.1	**Questions not appearing** **Restart quiz,**	
6.8.1	**Quiz not submitting.** **Results not appearing.** **Error message shown.** **Retry.**	

USE CASE	**7**	**Using Save System**	
Description of Goal in Context	**User will be able to successfully add animals to notebook database and save the game with working database using Easy-Save.**		
Post Conditions, Success End Condition	**Everything will be added to the database Easy-Save successfully with no errors.**		
Main Flow:			
Step	**Action**		**Alternate**
7.1	**Launch Game**		
7.2	**Click start game**		
7.3	**Load terrain**		
7.4	**Add animal to invenotry**		

7.5	**Press pause menu**	
7.6	**Click save game**	**E or EF**
7.7	**Exit game**	
7.8	**Click start game**	
7.9	**Select load game**	
7.10	**Load saved game**	**E or EF**
EXCEPTIONS or ERROR Flow Description		
Step	**Branching Action**	
7.5.1	**Save gaem name alredy used** **Prompt message to choose other name**	
7.8.1	**Failed to load game** **Prompt user to try again.**	

3.7. System Design

In this section, the system design will be outlined. The art style, the models, the animal AI behaviours, the audio, and the map/level design will all be discussed and shown.

3.7.1. Inspect system

As discussed in the research section, there will be an inspection system in this game. This means that when the user or player is roaming around the environment upon coming across an animal or some trees or plants, they will receive a prompt as shown in the 2nd iteration prototype that will enable them to inspect the object. When they have inspected the object, they will be able to view the wildlife and press a button to be added to their inventory.

This system will be created by using scripts in C# to use methods like Raycasting as discussed in the research section to allow the user to inspect the object and to be given a prompt to inspect it. Classes will be created to handle this system and it will be linked to the inventory system. This is because the user will need to be able to add the inspected item to the inventory.

3.7.2. Inventory system

As discussed in the research section, there will be an inventory system in this game. This system will allow the user to add animals or wildlife like plants or trees to their inventory. once these animals are in the inventory, they will be able to view the wildlife and look at information about the specific wildlife they have selected.

This will all be created by writing scripts in C# to create the inventory item data for the objects and build the system to handle the items. Different classes will be created to handle the items that are being collected and display the user interfaces for the inventory system.

```
[CreateAssetMenu(menuName = "Inventory Item Data")]
public class InventoryItemData : ScriptableObject
{
    public string id;
    public string displayName;
    public Sprite icon;
    public GameObject prefab;
}
```

FIGURE 46-SAMPLE INVENTORY CODE SNIPPET

3.7.4. Save system

As discussed in the previous section there will be a system in the game that allows the user to save and load their progress in the game. This system will enable the user to save their game progress from inside the game by using the pause menu and selecting the save game button. This will prompt he user to use a save name. when the game is saved the users wildlife, they have collected in their inventory will be saved. if the user exits the game and tries to choose the play game option again. They will be given a load game prompt where they can select which game save data they want to load before entering the environment again.

This will have three main components and classes. The first is the game classes which will be written in C# that will need data stored. Second is the save game class which is also written in C# that will hold the data. thirdly there will be the

serialization system to handle the saving and loading of the Binary File as discussed in the research section.

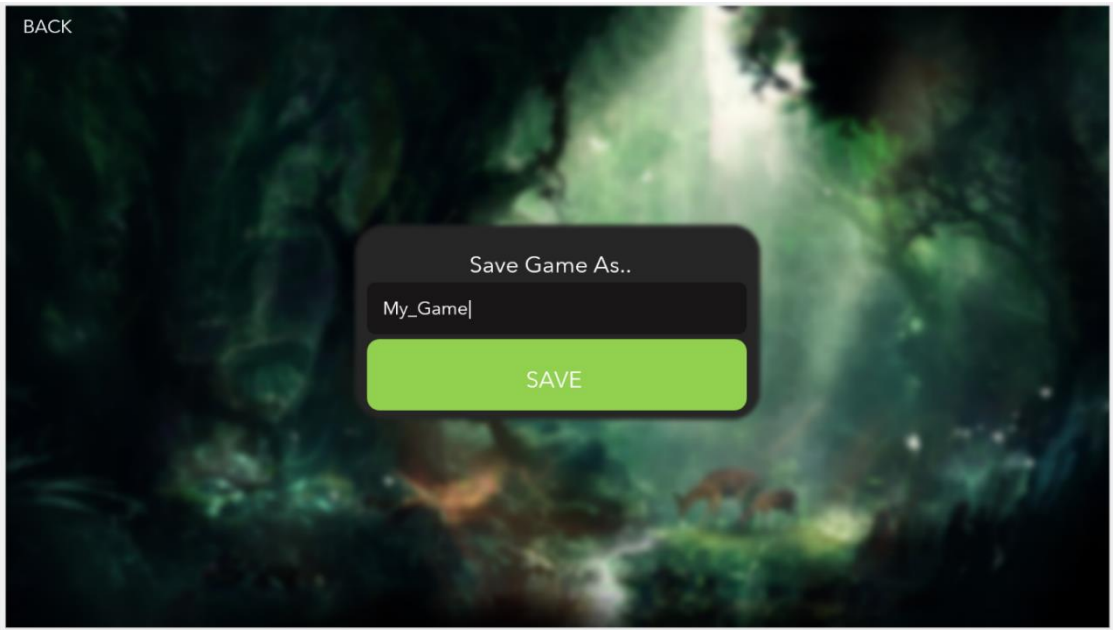


FIGURE 47-PROTOTYPE SAVE SYSTEM

3.7.5. Procedural animation

As discussed in the research section the animation for this project will be procedural. Meaning that it will be generated for each animal by writing code in C# scripts.

Each animal will have their own C# scripts for animation. Most of them will be quite similar apart from a few minor parameter adjustments. For example, the scripts might use Raycasting to finds the position of where an animal's foot should land next when it is running or walking. They might also use interpolation to move the animal from the previous position to the new one.

This method of animation will be applied to every animal.

An issue or risk that might occur when using procedural animation is that when creating different scripts for each animal, some scripts might have to be completely different for some of the animals. This would greatly increase the workload of animating the animals.

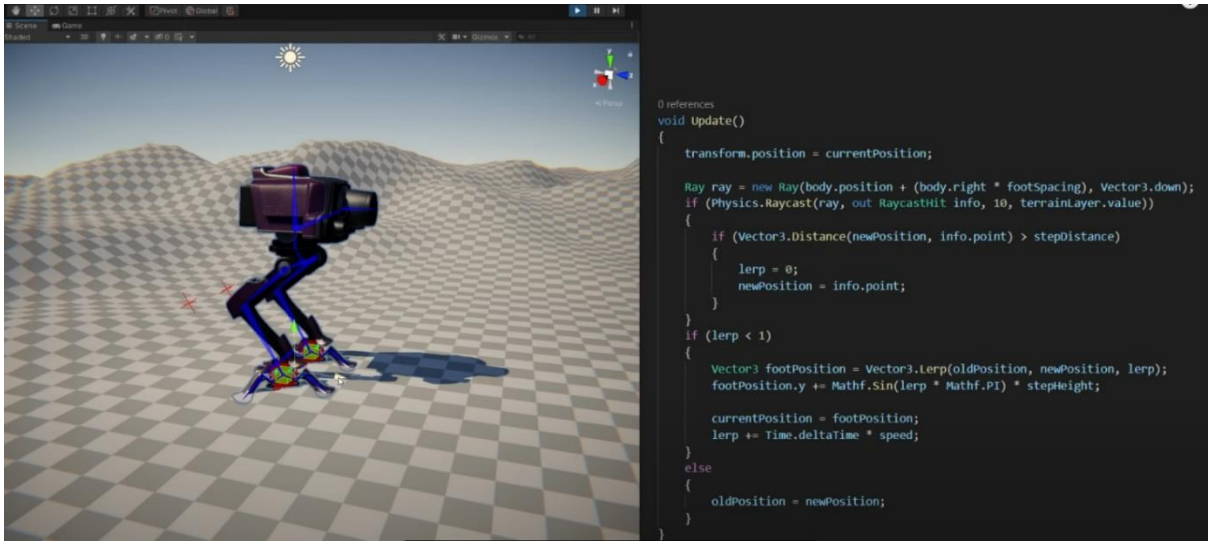


FIGURE 48-SAMPLE PROCEDURAL ANIMATION

3.7.6. Map / Level Design

The map that the user will be able to explore will be a forest terrain which includes all the models discussed in the previous sections. Each time the user will start the game a new random map will be generated. The method in which this will be accomplished is by using a form of procedural generation called **Perlin noise**. This will mean that the forest and terrain will be pseudo randomly generated each time the user plays the game. The forest and wildlife in the game will be in different positions and look slightly different every time. This terrain will act as the place where the user can explore and look for wildlife as a player. This will be done by Writing scripts in C# to allow the objects and forest to be generated.

FIGURE 49-SAMPLE PERLIN NOISE TERRAIN

3.7.7. Animal AI

In this system the animals that the user will be able to inspect throughout the terrains will need to use AI to allow them to behave and move throughout the environment. This will be done by using the NavMesh feature in unity as decided in the previous section.

The animals will be able to roam randomly around the terrain. The animals will also stop and rest whilst there are wandering around. These animals will act differently based on each type of animal and when the user gets close to an animal it will react in a certain way. The behaviours for the animals are in the table below. Each animal will have a behaviour script written in C# that allows them to roam around the terrain by wandering and resting using NavMesh. Each animal will also have a behaviour script that allows them to either approach the player when they are close to them or sometimes run away. Some animals will only run away from the player, and some will do both behaviours.

Behaviours Table

Animal	**General Behaviour**	**Behaviour Around Player**
Red Deer	-Wandering -Resting	-Runs away from player
Red Fox	-Wandering -Resting	-Approaches player -Runs away from player
Red Squirrel	-Wandering -Resting	-Approaches player -Runs away from player
Badger	-Wandering -Resting	-Approaches player -Runs away from player
Common Frog	-Wandering -Resting	-Runs away from player

3.7.8. Art

For the moment, a low poly art style will be used. This will save time when creating models. However, if there is enough time, a higher quality more realistic style of modelling and art will be created and used throughout the game. The colours used will match the colours of the forests and wildlife in Ireland. Below are sample real-life pictures taken in Wicklow, Ireland.(21)

3.7.9. Models

In Ireland there is lots of wildlife. In this project only a handful of animals, plant, trees, and other objects will be chosen to be put in the landscape. This is because attempting to create too many models will be too time consuming. The chosen species will be discussed below and how they will be designed.(22)

Animals

This project is based on wildlife in Ireland so all animals and wildlife will be chosen that is native to Ireland. Below are the animals that will be in the game. The user will be able to inspect and add these animals to their inventory. There will be five different animals in the game, a list of the animals and a mind map was created to showcase the animals and what they look.

- Red Deer
- Red Fox
- Red Squirrel
- Irish Badger
- Common Frog

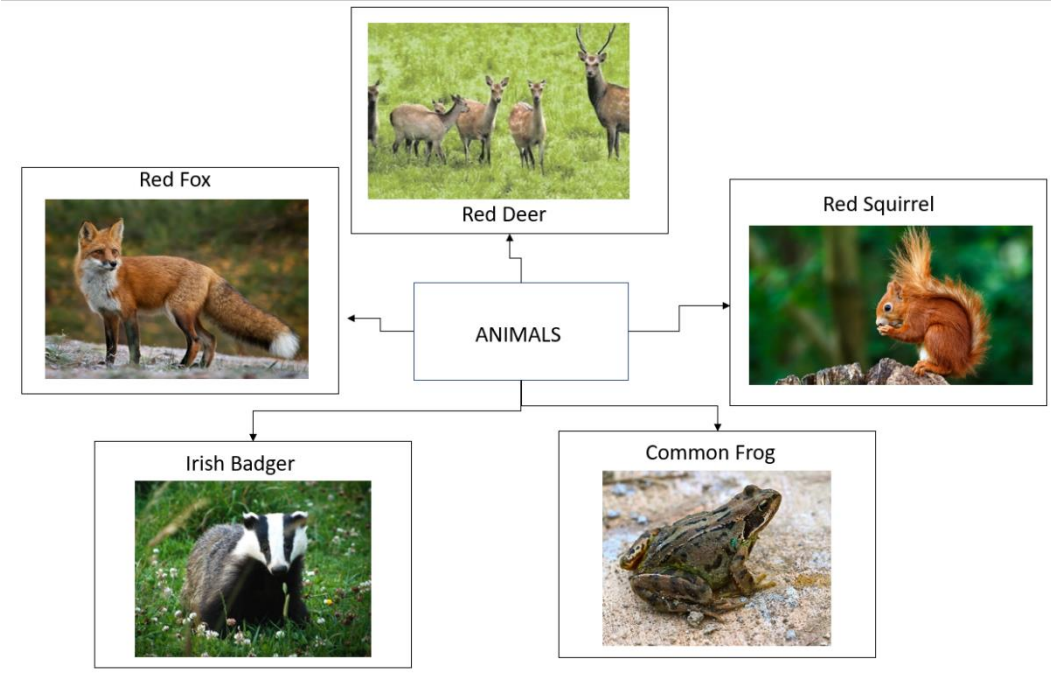


FIGURE 50-ANIMAL MIND MAP

Plants

There will be seven different plant wildlife in the environment. Each plant will be able to be inspected and added to the user's notebook inventory. They are as follows:

- Willow Tree,
- Maple
- Pine,
- Aspen
- Silver Birch.
- Douglas Fir
- Ferns
- Grass
- Bushes

Debris

The terrain will also contain various rocks, sticks and pebbles to make the terrain look more realistic.

- Two types of rocks
- Two pebble types
- One stick type

3.7.10. Audio

In this game, the audio will have an important part to play as it allows the user to be immersed in the game and listen to all the different sounds you would hear out in the wild in Ireland today. There will be sound effect for the animals, wildlife and a soundtrack for the game which will play in the background.

The table below shows a list of all the different audio sources that will be needed in the game for each Game Object.(23)

Audio Table

Game Object	**Audio Source**
Deer	Deer Noises
Fox	Fox Noises
Squirrel	Squirrel Noises

Badger	Badger Noise
Frog	Frog Noises
Terrain	Forest Noises Wind Bird noises.
Player	Footstep Noises
Main Menu	When buttons are clicked in the main menu there will be a clicking sound effect. Music when in menu.
Pause Menu	When buttons are clicked in the Pause menu there will be a clicking sound effect.
Cutscene	Narrator Audio and Music Audio
Inventory	When wildlife is added to the inventory audio will play.
Quiz	There will be quiz audio for button clicks and completion.
Loading screen	Music for loading screen

3.8. Conclusions

In this section, we explore and discuss the different aspects of design. It first looks at version control and setup of the project followed by the design methodology where it discusses the use of GitHub and the chosen ADDIE Model. It then talks about the system architecture along with the system Requirements. After this, the early prototypes are shown by presenting them in picture form. Use case scenarios are thoroughly examined looking at the different outcomes for this game and finally the System Design in relation to models, art style and audio are explained.

4. Development

4.1. Introduction

In this section, the overall approach taken in the system development for this project is discussed in detail alongside the prototype. The development will discuss many features such as the terrain generation, the wildlife AI, asset modelling, UI creation, animation, coding methods and educational techniques. The player movements and components along with all the necessary game components such as audio and how the game will be made available to the user. All development of usability features will be outlined.

4.2. Prototype

The first part of the development process was to create a new project in unity. A GitHub repository was created as discussed previously, it was cloned into a folder and then the project was pushed to the repository using the command 'git push origin main'. After this was completed the first prototype development was started.

The goal for the first prototype was to create the 'Start Menu' screen and the 'Options Menu' Screen in Unity. The layouts of the screens and the screens interacting with each other will be shown.

GitHub Link: https://github.com/jakebolger/Wild-Ireland-FYP

4.2.1 Prototype Development

The first screen that was created was the 'Start menu'. The first step was to create a new scene in unity. For this scene the background of the 'Start Menu' consisted of a forest background with trees, bushes, rocks, and a Frog. To do this, models needed to be created for the forest objects.

The first step to create these models was to create them using Blender. Overall, there were six trees created Birch, Aspen, Maple, Willow, Pine and Douglas Fir. The first tree modelled in 3d was the Birch tree. To create this in Blender the 'Sapling Tree Generator' Add on was used. This allows you to create custom trees from scratch by adjusting the trunk, branch and leaves of the tree to create the desired tree. Images from google were used as a reference to copy the appearance of the chosen trees.(24)

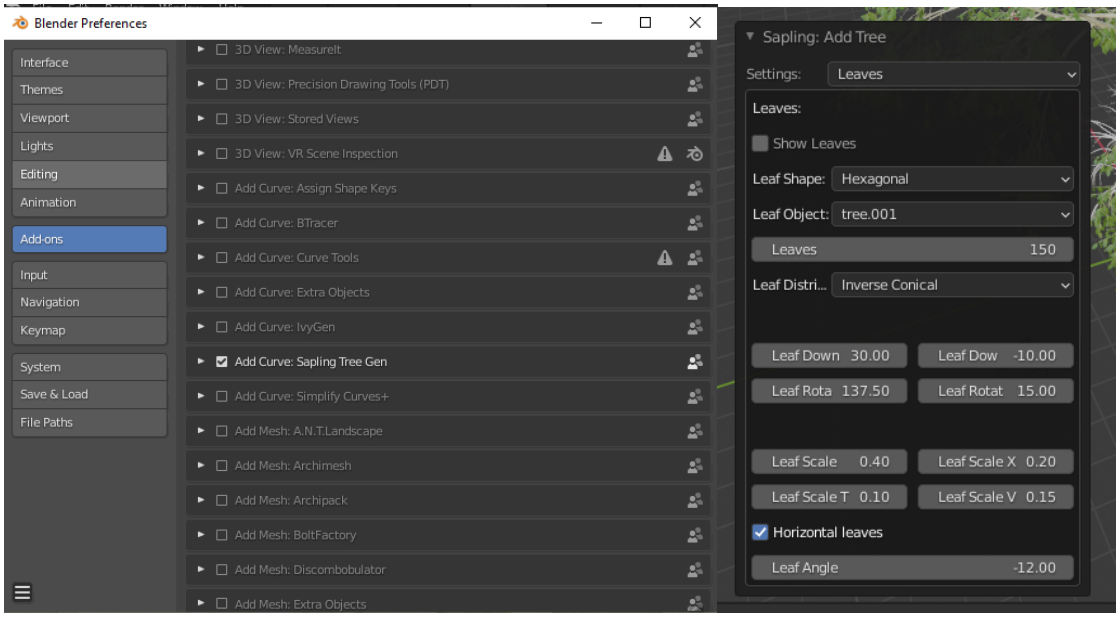


Figure60-tree gen add on

Once the models were created in Blender they were exported as FBX files so that they could be imported into the Unity Engine. In Unity the tree models were imported as new assets and placed in the background of the 'Start Menu' Scene.

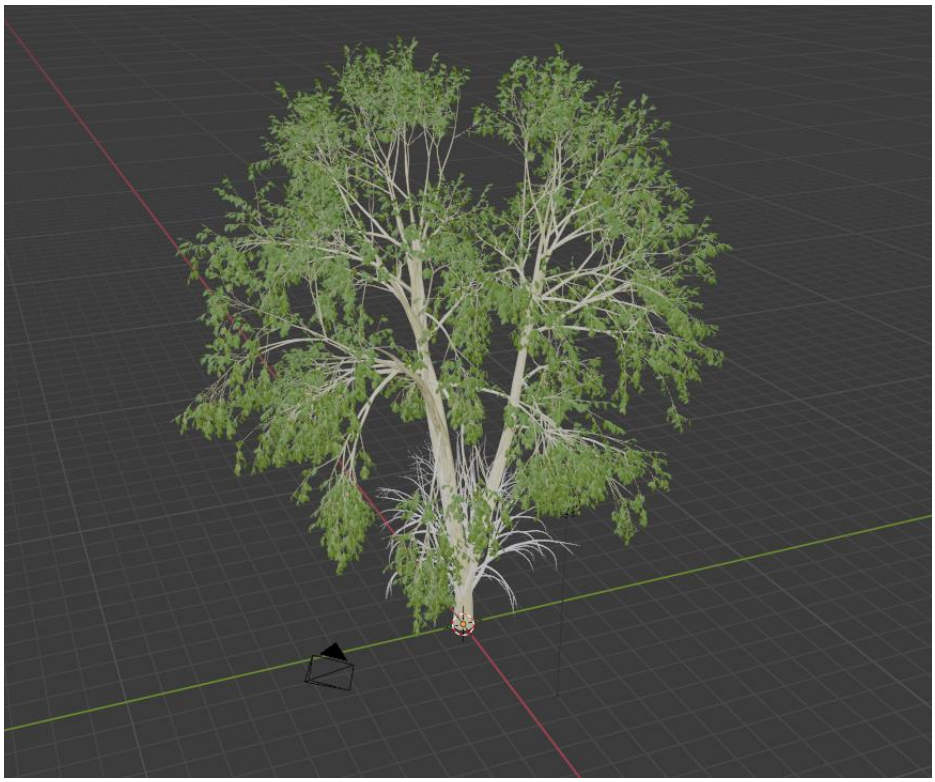


FIGURE 51-BIRCH TREE MODEL CREATED

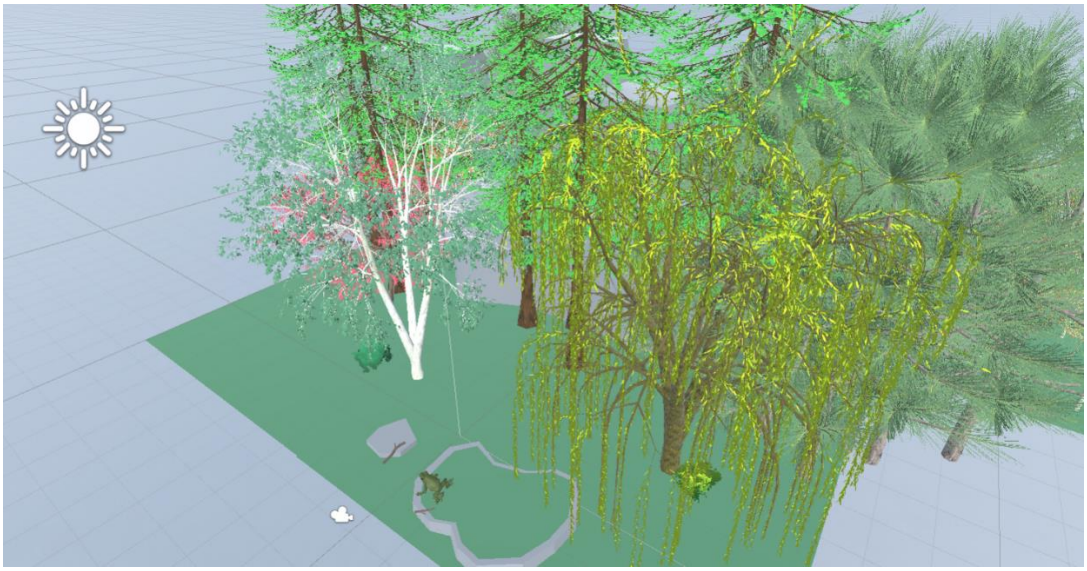


FIGURE 52-SCENE PLACEMENT

The next step was to create the menu UI. To do this a Canvas object was created in Unity which allows you to add text and Buttons to the screen. The title, the Play button, the options button, and the leave game button were added to the menu by creating button objects and text objects and adding them to the scene.

FIGURE 53-UI

The next screen that needed to be created was the ‘Options Menu’ the same method was used to create this menu as the ‘Start Menu’ except the buttons were different and had different functions. An options title was added, along with a volume title and controls title. Under the volume text a volume adjuster was created as an object in the hierarchy in Unity. This means the user will be able to adjust the volume.

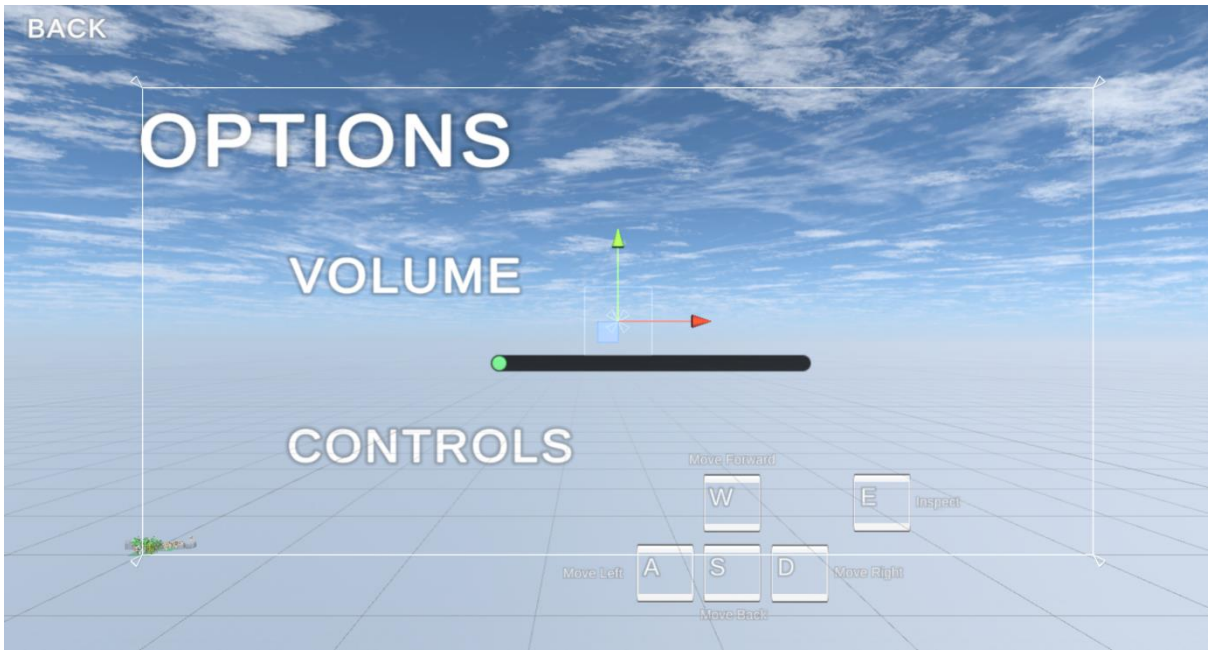


FIGURE 54-UI 2

Pictures of the controls were added under the controls title.

Finally, a back button was added to the ‘Options Menu’ to return to the ‘Main menu’.

To allows these screens to interact with each other and switch between the main menu and the options screen a C# script for the Menu was added. This script was added to the Main Menu, and it allow the user to press the play button and load into a new scene. It also allowed the user to press the ‘OPTIONS’ button and switch to the ‘Options Menu’. In the script there were two functions one for the PLAY button and one for the QUIT button.

MainMenu.cs

Miscellaneous Files — MainMenu

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using UnityEngine.SceneManagement;
5
6   public class MainMenu : MonoBehaviour
7   {
8       public void PlayGame()
9       {
10          SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
11      }
12
13      public void QuitGame()
14      {
15          Debug.Log("QUIT!");
16          Application.Quit();
17      }
18  }
```

FIGURE 55-CODE FOR SCREEN INTERACTION

Next free assets from the assets store were downloaded to get models for the grass, the sky, and the frog model. The reason these assets were downloaded and not created manually was because this is just a first prototype, and the models were used to fill in some of the terrain that was missing. These assets will not be used in the final system and will be replaced by custom models created in Blender. These models were added to the background scene along with an audio source that plays forest sounds on the background of the scene. The initial prototype was then complete.

FIGURE 56-START MENU PROTOTYPE

FIGURE 57-OPTIONS MENU PROTOTYPE

4.6. Conclusions

In this section the first two screens of the main menu and the options menu were created in unity. A script was written in C# to allow interaction between scenes, screens, and buttons. Models were created in blender of trees and exported into unity to be placed throughout a scene. A frog model was added to the scene. The two screens are displayed at the end.

5. Testing and Evaluation

5.1. Introduction

This section will outline the testing methods used in this project during its development. GitHub Version control will be discussed along with a White and Black Box testing plan. Tests groups will be made for users to try out the system and find errors. The final part of this section will discuss the evaluation of the project, outlining the issues and feedback.

5.2. Plan for Testing

For the duration of this project, tests will be run frequently. This will be done through Unity to make sure the game is functioning properly. As discussed in the Design section, the use of GitHub will help with testing so that if there are errors with the system, having backups will be useful so that the system can be returned to its original state before the errors were encountered.

White box testing and black box testing will also be performed as discussed below. However, these will not be the only ways of testing. There will also be other methods of testing in Unity, in scripts, the database, the models, and AI testing.

White Box Testing

White box testing allows the tester to examine the structure, design, and coding of a chosen system. This method will be used to improve the overall usability experience for the user and increase the design strength, whilst finding smaller errors throughout the application as well.

This project will involve a lot of trial and error regarding the development process. Each feature will be tested until they are suitable to be shown to a third-party user such as someone outside the project or a testing group. When errors arise, they will be recorded to be later solved.(25)

Black Box Testing

Black-box testing allows for scenarios or use cases to be tested where the system or software has errors. An example of these scenario errors would be a user inputting something like a username with the wrong syntax and the application won't output an error message. This will be classified as an error for black box testing. An example in this project would be if the user tried to

inspect an animal and the wrong animal is shown in the user's notebook, or the animal isn't shown at all.

For this system, user testing will be the most efficient way to outline the potential faults in the game. Group testing will be organized where possible end users will use the application and look for potential errors. Any issues identified in group testing and testing will single end users will be recorded either on a computer or by video if the test is conducted via an online call. Forms or surveys will be created and used to record feedback as well.

The final testing phase will consist of an alpha version of the game made available to play and be tested. This will identify the final error so that the finishing touches can be applied.(25)

Unity Testing

As the project gets bigger, there will be a lot more classes, scripts, and objects in Unity. This will make it more difficult to change parts of the project without getting an error. To bypass this issue, automated testing will be used to help ensure all the parts of the project are working as expected by using the Unity Test Framework package. This will save time as it won't rely on manual testing, and it enables you to edit and test your project in both play mode and edit mode.(26)

Scripts testing

When testing code and scripts, using a debugger enables you to inspect the source code whilst the game or application is running. This will be accomplished by using visual studio as a debugger to debug and check for errors in the C# code.

Database Testing

When backend testing the database two main features will need to be tested, the inventory system and the save game feature. This will be manually tested by running the game and manually using the features to see if they work. These features will also be tested by third party users. This process will be repeated until no error remain and the user can use the features without as intended.

Model Testing

When testing the models, they need to be tested for accuracy against real life wildlife. To do this, pictures will be examined to and compared to the models being created. This will involve a lot of trial and error. The models will be shown to third party users in the final testing stages to make sure they don't have any errors.

AI Testing

When testing the AI, the game will have to be played by both the developer and the third-party users in a variety of different scenarios. This means there will be a lot of trial and error to get the AI to work without errors. For example, the user will have to approach an animal AI Agent multiple times to see how it reacts to check for errors or to check for the right response.

Test Plan Table

Test No.	Test Description	Expected Outcome	Pass? - ✓/✗
1	Does the game successfully download?	The game will download to the user's computer.	
2	Does the game launch when selected?	The game will start successfully.	
3	Does the loading screen appear?	When game is launch or loading terrain loading screen will appear	
4	Does the Game take long to load?	Will take less than a minute?	
5	Does the Start menu appear?	Menu appears.	
6	Is the music playing?	Music playing on loop.	
7	Does the settings button work?	Bring you to options.	
8	Can you adjust the volume?	Volume slider adjusts game volume.	
9	Can you View the controls?	Controls are visible to user.	
10	Does the back button work?	Back button brings you back	

		to previous screen.	
11	Does the quit button work?	Quit button exits the application.	
12	Does the start game button work	Play button loads in the terrain.	
13	Does the game load quickly?	Should take less than 1 minute.	
14	Does the Game work when loaded?	User able to navigate through menus and move throughout terrain.	
15	Is the User allowed to manoeuvre?	User can use keyboard to manoeuvre throughout terrain.	
16	Does the Controls match the default?	The controls match the controls in the options menu.	
17	Are the objects spawned in?	The terrain, forest and animals are spawned in.	
18	Does the game look as intend?	The game looks as is intended. Lighting is correct.	
19	Can you interact with animals?	animals react to the environment and player	
20	Can you interact with wildlife?	The wildlife approaches or runs away from player.	
21	Does the AI react to user?	Certain species approach player, certain species	

		don’t, some run away.	
22	Does the AI work?	The AI performs rules set in scripts.	
23	Is the terrain Generated?	The terrain is generated, and the user can walk on it.	
24	Is The environment generated?	Trees and grass are generated.	
25	Does the inventory work?	Inventory is functional.	
26	Does the inventory work when animal added?	Animal is shown in inventory.	
27	Does the Quiz work?	Quiz is functional.	
28	Do all the Action buttons work?	Buttons in quiz create an action.	
29	Does the settings menu work?	Options menu is functional.	
30	Do the buttons in the menu Allow you to adjust volume, view controls and quit?	Functional.	
31	Does the save feature work?	The user can save their progress. They can also load their progress.	
32	Does the Resume button work?	The user can resume game.	
33	Are there any glitches?	No.	
34	Are there any performance issues?	No.	
35	Is the game running slow?	No. depends on user’s computer.	
36	Are the error messages displayed?	Yes.	
37	Does the information display in information screen	Correct information is displayed.	

38	Does the video Play in inventory?	Yes, video plays.	
39	Can you exit the inventory?	Pressing exit exits the inventory.	
40	Does the inspect prompt show for wildlife?	Prompt appears when near wildlife. Or looking at it.	
41	Is the in-game music playing?	Music is playing in background.	
42	Is the sound effect/Audio working?	Sound effects functional.	
43	Are all animations being executed?	Animals animated successfully.	

5.3. Plan for Evaluation

In this project the evaluation is a high priority, this is because the user experience when using this system is extremely important. The application should be easy to use, and not require the user to be wondering how to use it or not know what the application is about. When this system is evaluated by a variety of end-users, the developer and possible industry professionals, it will mean that the system has been evaluated in many ways so the standard of usability will be at its strongest. This section will outline what approach was taken to evaluate the system and the metrics used as well.

5.4. Conclusions

This section discussed the testing and evaluation of this project. The testing discussed was white box and black box testing alongside using GitHub for backups and end-user testing. The evaluation consists of feedback from lots of potential end-users which was considered and used to change and successfully complete the system development.

6. Issues and Future Work

6.1. Introduction

This Final section of the project will reflect on the entire system. It will discuss possible future work for the project and then conclude the project by talking about its key points.

6.2. Issues and Risks

In this project there a several challenges or issues that are not yet resolved. These issues are as follows:

- Low amounts of knowledge in Rigging in Blender
- Struggling to find way of using LIDAR technology
- Lack of Experience with coding animal AI.
- No experience in using the Easy Save Database.
- Lack of time to Organise testing groups.

To solve these issues the following actions will be taken:

- Complete research in rigging, use tutorials and look at online courses to improve knowledge.
- If unable to acquire phone that will work with LIDAR, Model the terrain assets in blender instead.
- Undergo research for AI, watch tutorials and look at academic papers.
- Practice using database and watch tutorial on how to use it. Then, read the guidelines.
- Set aside time to organize test groups and make sure they are as efficient as possible.

Next, the risks will be discussed. They are as follows:

- Only able to work on project on PC mostly from home as laptop isn't very powerful.
- Workload could be too much, as system is feature heavy.
- Not meeting required deadline.
- Prototype not fully functioning for Demo.
- Research links might not be reliable sources of information.

To solve these risks the following actions will be considered:

- Work on more complex programs such as Unity and Blender from home. Work on applications such as word and GitHub on laptop.
- Reduce number of features that might contribute to a work overload.

- Manage time efficiently by completing work within plenty of time.
- Make sure testing phases is completed to allow for a fully functioning prototype.
- Don't use Wikipedia as a source or information that might be unreliable.

Risks in Blender

- The textures and materials might not be exported correctly.
- might cause unity to run slower
- handling the models might cause problems for example if you created a tree model in Blender, there could be thousands of leaves on that tree.

Risks in Unity

- using too much memory or space in the project
- slows the project or game down.
- Assets like materials and textures take up lots of memory.

Risks in AI

- seems to be lots of minor error and bugs with the NavMesh system, when navigating around complicated obstacles.

Risks in Animation

- some scripts might have to be completely different for some of the animals. This would greatly increase the workload of animating the animals.

6.3. Plans and Future Work

Plans for future work on this project will be shown using a GANTT chart. this chart will outline the plans for this project and show specific dates and topics to be completed. When more progress is made on this project, a second GANTT chart will be used if any changes are made. By the end of this project the two GANTT charts will be compared. This will show the differences between the two and how the plans differ from the final product.

By the end of the project features that could have been implemented into the project will be discussed, along with possible game mechanics that might improve the game by using the feedback given by the end-users for this project. Features such as the procedural generation, AI, UI, Modelling will be discussed as to whether they should be improved or changed.

6.3.1 GANTT Charts

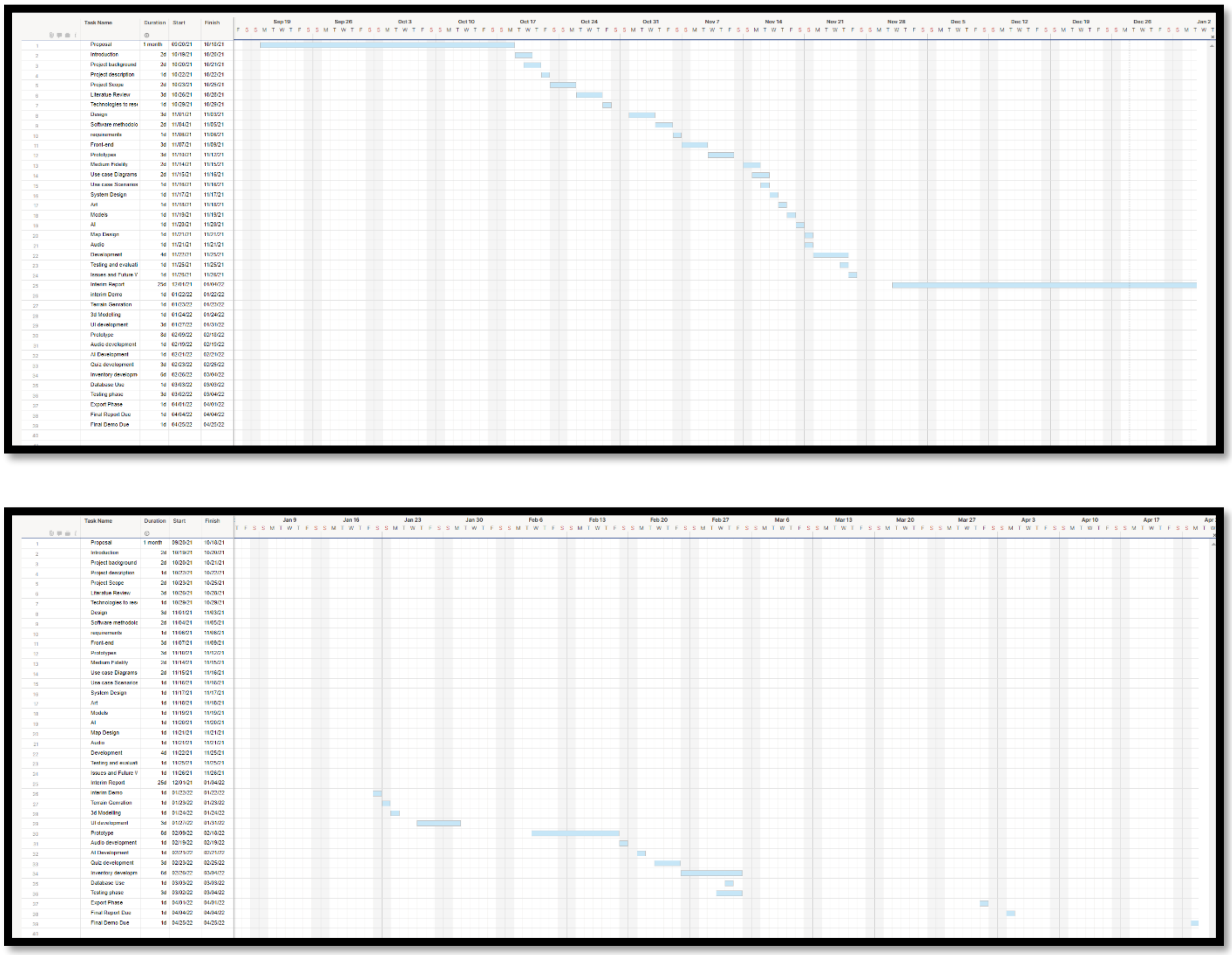


FIGURE 58-GANTT CHART

	Task Name	Duration	Start	Finish
1	Proposal	1 month	09/20/21	10/18/21
2	Introduction	2d	10/19/21	10/20/21
3	Project background	2d	10/20/21	10/21/21
4	Project description	1d	10/22/21	10/22/21
5	Project Scope	2d	10/23/21	10/25/21
6	Literatue Review	3d	10/26/21	10/28/21
7	Technologies to resea	1d	10/29/21	10/29/21
8	Design	3d	11/01/21	11/03/21
9	Software methodology	2d	11/04/21	11/05/21
0	requirements	1d	11/06/21	11/06/21
1	Front-end	3d	11/07/21	11/09/21
2	Prototypes	3d	11/10/21	11/12/21
3	Medium Fidelity	2d	11/14/21	11/15/21
4	Use case Diagrams	2d	11/15/21	11/16/21
5	Use case Scenarios	1d	11/16/21	11/16/21
6	System Design	1d	11/17/21	11/17/21
7	Art	1d	11/18/21	11/18/21
8	Models	1d	11/19/21	11/19/21
9	AI	1d	11/20/21	11/20/21
0	Map Design	1d	11/21/21	11/21/21
1	Audio	1d	11/21/21	11/21/21
2	Development	4d	11/22/21	11/25/21
3	Testing and evaluatior	1d	11/25/21	11/25/21
4	Issues and Future Wo	1d	11/26/21	11/26/21
5	Interim Report	25d	12/01/21	01/04/22
6	interim Demo	1d	01/22/22	01/22/22
7	Terrain Genration	1d	01/23/22	01/23/22
8	3d Modelling	1d	01/24/22	01/24/22
9	UI development	3d	01/27/22	01/31/22
0	Prototype	8d	02/09/22	02/18/22
1	Audio development	1d	02/19/22	02/19/22
2	AI Development	1d	02/21/22	02/21/22
3	Quiz development	3d	02/23/22	02/25/22
4	Inventory developmen	6d	02/26/22	03/04/22
5	Database Use	1d	03/03/22	03/03/22
6	Testing phase	3d	03/02/22	03/04/22
7	Export Phase	1d	04/01/22	04/01/22
8	Final Report Due	1d	04/04/22	04/04/22
9	Final Demo Due	1d	04/25/22	04/25/22

FIGURE 59-GANTT CHART SHEET

Bibliography

1. Beyond Blue – epicgames.com [internet]. [Cited 2021 November 1]. Available From: https://www.epicgames.com/store/en-US/p/beyond-blue
2. The hunter: Call of the Wild – callofthewild.thehunter.com [internet]. Cited 2021 November 1]. Available from: http://callofthewild.thehunter.com/
3. Zoo Tycoon – steampowered.com [internet]. Cited 2021 November 1]. Available from: https://store.steampowered.com/app/613880/Zoo_Tycoon_Ultimate_Animal_Collection/
4. Unreal Engine - unrealengine.com [internet]. Cited 2021 November 1]. Available from: https://www.unrealengine.com/en-US/
5. CryEngine – cryengine.com [internet]. Cited 2021 November 1]. Available from: https://www.cryengine.com/
6. Armory – armory3d.org [internet]. Cited 2021 November 1]. Available from: https://armory3d.org/
7. Unity – unity.com [internet]. Cited 2021 November 1]. Available from: https://unity.com/
8. Programming languages – makeuseof.com [internet]. [Cited 2021 November 7]. Available from: https://www.makeuseof.com/tag/unity-game-development-languages/
9. Blender – Trustradius.com [internet]. [Cited 2021 November 7]. Available from: https://www.trustradius.com/products/blender/reviews?qs=pros-and-cons
10. Cutscene – medium.com [internet]. [Cited 2021 November 7]. Available from: https://medium.com/geekculture/creating-a-cutscene-in-unity-aaec5042aaab
11. Inventory System – youtube.com [internet]. [Cited 2021 November 7]. Available from: https://www.youtube.com/watch?v=SGz3sbZkfkg
12. Inspection System – youtube.com [internet]. [Cited 2021 November 8]. Available From: https://www.youtube.com/watch?v=au3GrDuJaEw&t=557s
13. Procedural generation – adrianb.io [internet]. [Cited 2021 November 8]. Available From: https://adrianb.io/2014/08/09/perlinnoise.html

14. NavMesh – unity3d.com [internet]. [Cited 2021 November 8]. Available From: https://docs.unity3d.com/ScriptReference/AI.NavMesh.html
15. Procedural animation – youtube.com [internet]. [Cited 2021 November 15]. Available From: https://www.youtube.com/watch?v=acMK93A-FSY&t=115s
16. Project no 1 -tudublin.ie [internet]. [Cited 2021 November 15]. Available From: https://library-cc.tudublin.ie/search/?searchtype=X&searcharg=dt211c+or+DT228+or+DT282+or+DT255&sortdropdown=-&SORT=DZ&extended=0&SUBMIT=Search&searchlimits=&searchorigarg=Xdt211c+or+DT228+or+DT282%26SORT%3DDZ
17. Project no 2 -tudublin.ie [internet]. [Cited 2021 November 15]. Available From: https://library-cc.tudublin.ie/search/?searchtype=X&searcharg=dt211c+or+DT228+or+DT282+or+DT255&sortdropdown=-&SORT=DZ&extended=0&SUBMIT=Search&searchlimits=&searchorigarg=Xdt211c+or+DT228+or+DT282%26SORT%3DDZ
18. Design methodology – ceur-ws.org [internet]. [Cited 2021 November 15]. Available From: http://ceur-ws.org/Vol-1394/paper_9.pdf
19. ADDIE Model – learnupon.com [internet]. [Cited 2021 November 15]. Available From: https://www.learnupon.com/blog/addie-5-steps/
20. Unity System architecture – google.com [internet]. [Cited 2021 November 15]. Available From: https://www.google.com/search?q=inty+system+architecture&rlz=1C1JJTC_enIE927IE927&sxsrf=AOaemvLjQf-W0UbMWvs2ln-nGv4YvuGppA:1641213337427&source=lnms&tbm=isch&sa=X&ved=2ahUKEwi6gOmPzJX1AhV4QkEAHbo7CZEQ_AUoAXoECAEQAw&biw=2133&bih=1041&dpr=0.9
21. Art – youtube.com [internet]. [Cited 2021 November 15]. Available From: https://www.youtube.com/watch?v=QTM2Yr_EibU
22. Models – vincentwildlife.ie [internet]. [Cited 2021 December 16]. Available From: https://www.vincentwildlife.ie/species
23. Audio – unity3d.com [internet]. [Cited 2021 December 16]. Available From: https://docs.unity3d.com/Manual/class-AudioSource.html

24. Tree Creation Blender – youtube.com [internet]. [Cited 2021 December 16]. Available From: https://www.youtube.com/watch?v=jMDRc4hJwvA&t=507s
25. Testing White and Black – geeksforgeeks.org [internet]. [Cited 2021 December 16]. Available From: https://www.geeksforgeeks.org/differences-between-black-box-testing-vs-white-box-testing/
26. Unity testing – unity3d.com [internet]. [Cited 2021 December 16]. Available From: https://docs.unity3d.com/Manual/testing-editortestsrunner.html
27. Scripts testing – unit3d.com [internet]. [Cited 2021 December 28]. Available From: https://docs.unity3d.com/Manual/ManagedCodeDebugging.html
28. Gantt chart – smartsheet.com com [internet]. [Cited 2021 December 28]. Available From: https://app.smartsheet.com/sheets/48hphGP8p6Prfm3Rhgc3cp383Q54rj78W8rxMqQ1?view=gantt
29. Design – The Art of Game Design [Book]. [Cited 2021 January 1]. Available From: The Art of Game Design by Jess Schell