Durham University MeditateVR User Manual

Group 1

Adam Seidel (nnng27)

Ben Hughes (krcc82)

Enego Comley (wdsz78)

Pad Pang (gjzt58)

Nathaniel Mosey (wjld56)

Tom Owen (zjmq47)

Table of Contents

1. Intro

Meditate VR is a piece of immersive Meditation software developed for researching the effectiveness of abstraction and gamification elements in meditation. The product has been developed at the request of the client, Professor Alexandra Cristea, a researcher in Psychology at Durham University. The software allows for the researcher to set up different meditation environments and configure meditation settings to test the effectiveness of different meditation experiences through biometric data gathered by the system. After setup the participants are entered into an abstracted meditation world in which they can see visual breathing aides as well as a visual representation of their meditative state through a plant growing in front of them. All data pertaining to the participant's meditation experience (telemetry, biometrics, environment settings) are stored in a file for future research analysis.

This document is a user manual designed to guide researchers, participants, and developers on how to use and maintain the software. ***Section 2. User setup*** is aimed at non-technical users looking to conduct research with the product. .. As such, this section includes details on how to configure the headset and the environment, how to carry out the experiment, and how to access the subsequent data. Users will also find a final subsection on troubleshooting common issues. To make this section accessible, we've minimized the use of specialized jargon as much as possible, but due to the intricate nature of the product technical terms may still be encountered. These have been contextualized and described in layman's terms to the best possible extent. ***Sections 3. System Components***, ***4. Future Development and Ethical Considerations*** and ***5. System Maintenance*** are dedicated to developers and other technical stakeholders. ***Section 3.*** outlines how the system has been developed, its architecture, and covers all functional requirements and non-functional requirements. This is augmented in ***Sections 4.*** and ***5.*** with further information about maintaining and developing the system further, alongside a discussion of ethical and societal implications of the product.

1.1 Fulfilment of Requirements

After first coming to the client with our ideas of how we could fulfil their requirements we produced a comprehensive requirement spec detailing what their specific functional requirements were. The table below outlines these requirements and how we have fulfilled them or why we chose to leave them out of the product.

Requirement Code	MoShCo	Fulfilled	Comments
FR 1.1 **Heart Rate Tracking System**	Must	No	We chose instead to implement a brainwave tracking system using the Epoc X. We chose to do this as we felt the brainwaves gave more detail, ass there were multiple relaxation metrics and after voicing this with the client they agreed that the Brainwaves would be more useful. As it currently works the brainwave system is functional and sends data to the computer but there are problems with how the user wears the headset, which will be resolved when the client obtains a new brainwaves headset.
FR 1.3 **Detailed Plant Growth System**	Must	Yes	We have developed an adaptive plant growth system which takes the user's biometrics and shows them a tree growing with the rate of growth determined by the user's relaxation and stress levels. The tree has passed our user acceptance tests and we believe it will help with meditation.
FR 2.2 **Low Poly Meditation Environment**	Must	Yes	We have developed 2 low poly meditation environments for researchers to use, with 2 distinct locations, a beach and a forest giving the researcher choice in where they want the meditation to take place.
FR 3.2 **Secondary User Config**	Must	Yes	As it is we have implemented a simple user config that can be changed before the program is run.
FR 3.5 **User Telemetry Stored in File**	Must	Yes	The system outputs data about the user's biometrics including relaxation and stress as well as other biometrics into a JSON file stored in the logs folder.
FR 1.2 **Single Script to run the software**	Should	No	After demonstrating the system to the client through the Unity workspace they gave us feedback on how they liked having the ability to configure all parts of the meditation and move the users round the environment as it went on and so we have chosen to keep the project within the unity workspace
FR 2.1 **Visual Meditation Guide**	Should	Yes	There is a breathing object which grows and shrinks according to breathing timings that help to reduce heart rate and stress levels, relaxing the user.
FR 2.3 **Multiple Poly Meditation Environments**	Should	Yes	We have implemented 2 different Low poly meditation environments, each with their own unique assets, an island, and a forest.
FR 2.4 **Music and Atmospheric Sound**	Should	Yes	There is calming atmospheric music that plays during the meditation, to relax the user and improve meditation.
FR 2.5 **Audio Meditation Guidance**	Should	No	We have yet to implement Audio Guidance as we feel this may take away from the abstraction and seem

			annoying for the user. We will look into asking potential users what they think of this
FR 3.1 **Detailed VR User Telemetry**	Should	Yes	Detailed telemetry about the environment setup as well as user biometrics and what position and where the user is looking are all stored in the meditation logs.
FR 2.6 **Detailed User Movement Around Environment**	Could	Yes	By using the Unity workspace, the researcher can move the user around the environment. We are looking into giving the user different parts of the environment they can move into.
FR 3.3 **User self-reflective Data**	Could	No	We chose not to implement the User self-reflective survey as we felt it may be disruptive to the meditation experience having a popup asking the user how their meditation is going.
FR 3.4 **Secondary Biometrics Data**	Could	No	We chose not to implement any additional biometrics measures. Currently we use the brainwaves, and we had the option to use heart rate as well, but we found the heart rate monitor was difficult for the researcher to set up and required lots of additional equipment that went against our idea of having a simple setup. We were also limited to what equipment the client had and though that the brainwaves produced the best data out of all our options, this opinion was shared by the client
FR 3.6 **Data Analysis System**	Could	No	We chose not to implement the data analysis system as we felt our analysis would not be useful for the researcher as they would want to perform their own analysis.

2. User setup

While this section is written for the Meta (Oculus) Quest 2, it is largely platform agnostic and should function on other PCVR headsets such as the Rift S or Pico 4.

2.1 Download Prerequisites and Program

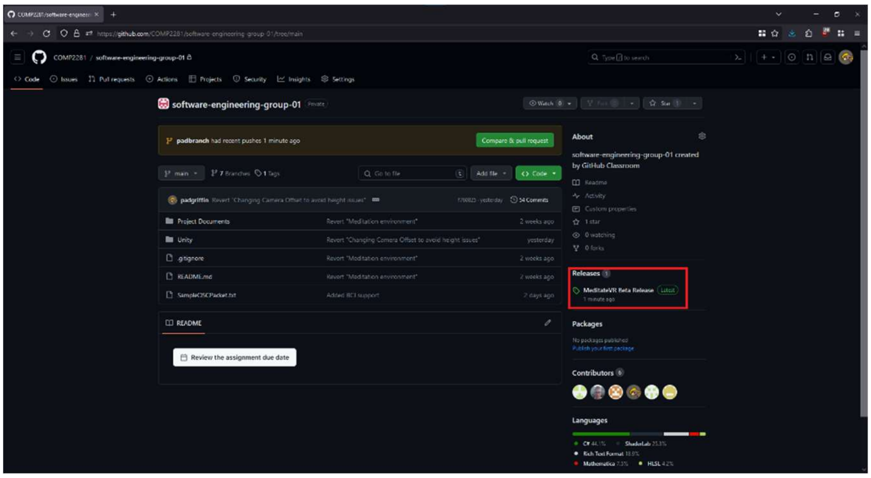
The program requires the use of the Oculus application (for Oculus Link), the Emotiv BCI desktop app, SteamVR and Unity Hub to function. The downloads for these are in the table below. If these links become unavailable, you will also be able to find downloads on google.

Application	Download link
Oculus Application	https://www.oculus.com/download_app/
Emotiv BCI Desktop App	https://www.emotiv.com/products/emotiv-launcher#download
SteamVR	https://store.steampowered.com/app/250820/SteamVR/
Unity Hub	https://unity.com/download

Download the latest version of MeditateVR from the releases tab of the GitHub repository (highlighted) and extract the project files to the desired location on the host computer.

2.2 Setup and Connect Quest 2

Turn on and connect the Quest 2 via Link Cable to the host computer that will run MeditateVR. Ensure that your headset has sufficient power for the duration of the meditation session.



padbranch had recent pushes 1 minute ago

padgriffin Revert "Changing Camera Offset to avoid height issues" f760823 · yesterday 54 Commits

Project Documents	Revert "Meditation environment"	2 weeks ago
Unity	Revert "Changing Camera Offset to avoid height issues"	yesterday
.gitignore	Revert "Meditation environment"	2 weeks ago
README.md	Revert "Meditation environment"	2 weeks ago
SampleOSCPacket.txt	Added BCI support	2 days ago

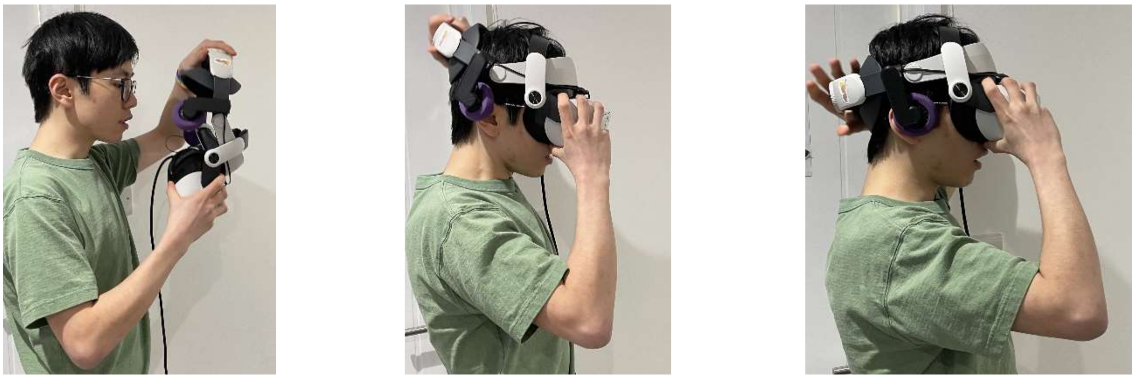
software-engineering-group-01 created by GitHub Classroom

The following is a brief demonstration of wearing the Quest 2:

Step 1: lift the strap and ensure that the headset is on.

Step 2: Put the headset around your head and lower the rear strap around the back of your head.

Step 3: Ratchet the mechanism until secure.



Note: Actual design of strap may vary. Meta Quest 2 with BoboVR M2 Pro and Koss KPH40x mounted headset used for demonstration.

2.3 Setup brainwaves headset

Once the Emotiv BCI desktop app has been installed, and the BCI-OSC plugin has been purchased, the user must follow the instructions on the BCI desktop app to setup the headset. Once this is completed, they need to set up the OSC data stream. To do this you:

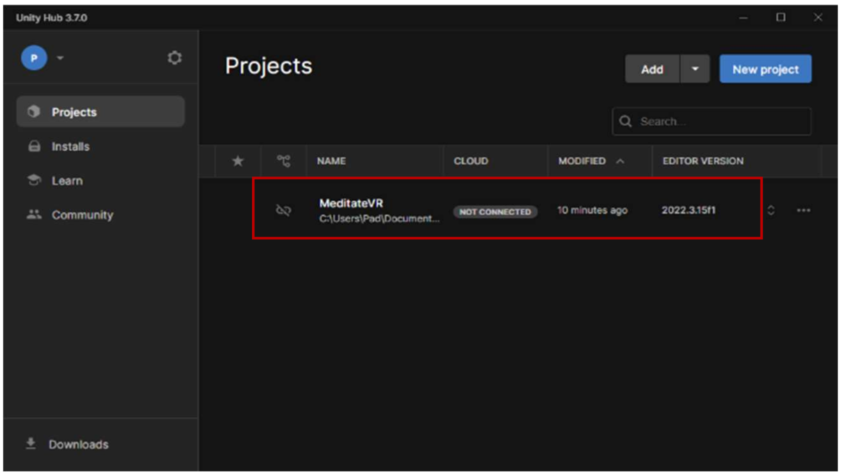
1. Open the BCI-OSC tab.
2. Set the Sending mode to Unicast to Self
3. Set the IP to 127.0.0.1. (Or if the IP address of the computer running the meditation game if it’s being run externally).
4. Set the Port to the one specified in the config. (By default, this will be 8000).
5. Choose the Performance Metrics data stream.
6. Click Start

2.4 Configure program

Various environmental parameters can be adjusted and enabled or disabled using our config.txt file within the assets folder of the Unity project. Inside this file, the researcher will find many different parameters that they can adjust to their liking. The config file is intended to allow the researcher to make changes to the participant's environment session so they can study the effect of these changes on the quality of a participant's meditation. All changes made will be reflected in Meditate VR the next time it is started. Changes to the file are not reflected in live sessions. The parameters of the config file are shown in the table below.

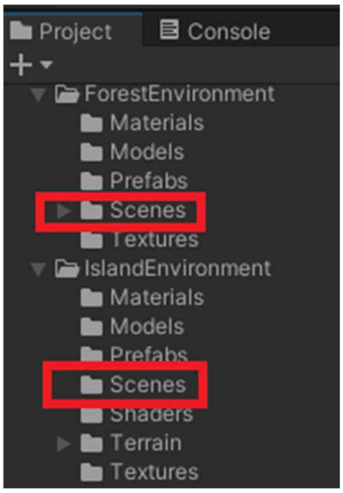
Name	Possible Values	Default Value	Explanation
Environment	Forest, Garden, Beach	Forest	Selects what kind of environment the user will be in
Detail	Low, Medium, High	Low	Selects how detailed the environment will be
Biometric	Brainwave, HeartRate, None	None	Selects what kind of biometric to use, either heartrate which has not been implemented yet, brainwaves or no data to be used, where the system will grow on its own.
PlantGrowthRate	Float between 0 and 1	0.5	A variable that controls how quickly the plant grows
VisualGuidance	true, false	true	Boolean to turn off or on the Visual Guidance
AudioGuidance	true, false	true	Boolean to turn off or on the AudioGuidance
Music	true, false	true	Boolean to turn the Music off or on the
ReflectiveSurvey	true, false	true	Boolean to turn the Survey off or on
Telemetry	false (basic), true (detailed)	true	Boolean to decide how detailed the output telemetry is
BreathingPattern	4-2-4, 4-4-4, 4-7-8	4-4-4	String that decides what breathing pattern to use

2.5 Run program

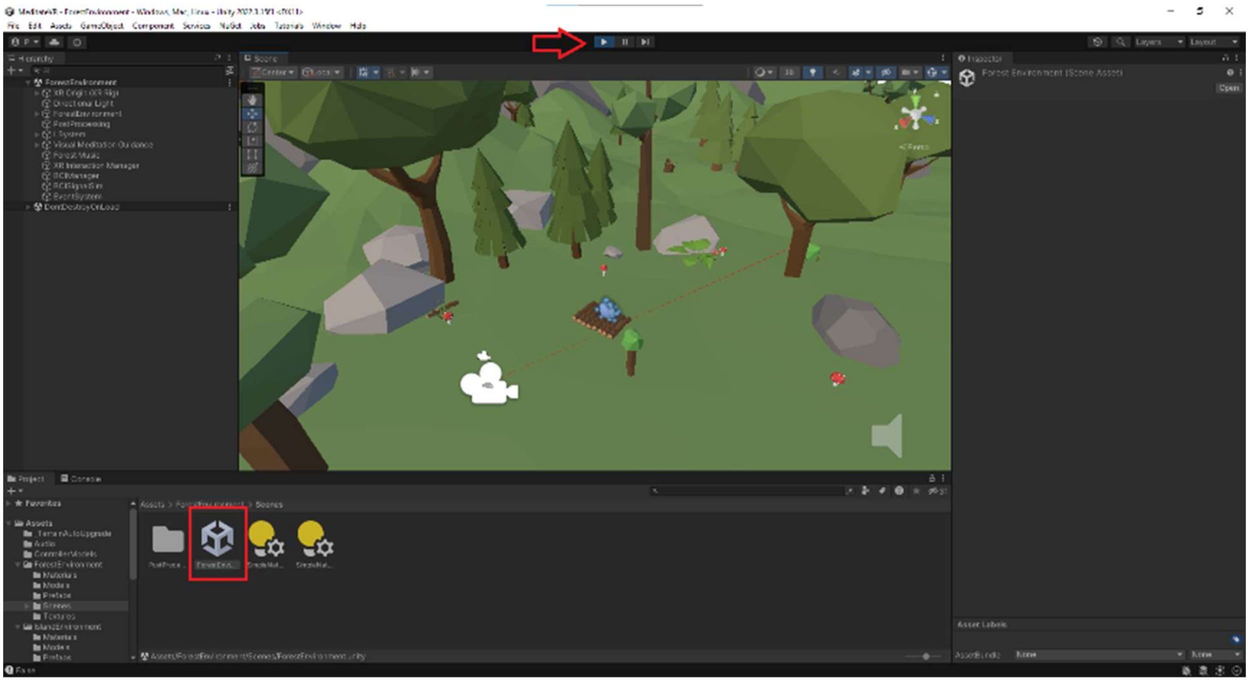


★		NAME	CLOUD	MODIFIED	EDITOR VERSION
		MeditateVR C:\Users\Pad\Document...	NOT CONNECTED	10 minutes ago	2022.3.15f1

Once Unity Editor is open, navigate to the assets tab. You should be able to see "ForestEnvironment" and "IslandEnvironment". Select the Scene Object for the desired environment.



Once loaded, press the play button. You should now be loaded into the VR environment.



Throughout the session, the user's metrics are tracked and logged to a JSON file through our biometrics system.

2.6 Troubleshooting common issues

2.6.1 Static Manager is not present

If the StaticGameManager prefab is not present in the scene then you will get some errors. If you get the “NullReferenceException: Object reference not set to an instance of an object” then you will need to drag the StaticGameManager prefab from the prefabs folder into the scene.

2.6.2 Camera is at incorrect height / Position

Hold down the Oculus button on the right controller until a menu comes up. Click “reset view” to reset the camera.

2.6.3 Performance issues

VR projects such as MeditateVR require large amounts of graphical processing. If performance issues are encountered, then the user should check if the system requirements of the host computer meet those of the project. The recommended system requirements for our project are specified in the table below.

Recommended System Requirements	
OS	Windows 10/11 (64-bit versions)
Processor	Intel Core i7/AMD Ryzen 7 (or greater)
Memory	16GB
Graphics	Nvidia RTX 3070 (or greater)
Storage	16GB available space

Another source of performance issues that we have found is power. When the host computer and or VR headset is not fully charged or connected to a wall power supply, the system will limit the processing power allocated to Meditate VR. This results in the world rendering in a lower quality or lagging as it renders. In our experience, this is particularly applicable to instances of our project that are being ran on laptops. Connecting a power supply to the host device and headset will resolve any lag caused by this issue.

2.6.4 User cannot see anything

Reboot the Quest 2 by holding down the power button on the headset, then reconnect the headset to the host computer via Oculus Link. If the problem persists, there may be an issue with the Link Cable or headset.

2.6.5 The BCI is not connected

If the JSON file is receiving no data, there may be a problem with the BCI. To solve this issue, the researcher should make sure that the BCI is connected properly. They should navigate to the connection page and make sure that all the nodes on the BCI are connected. Once the BCI is connected, the user may wish to continue the current session or start again.

2.6.6 No environments

The environments can be found within the ForestEnvironment and the IslandEnvironemtn folders within the assets folder. From there you can open the scenes to run the game. If these folders cannot be found then you may have to redownload the project from the GitHub repository.

2.6.7 Corrupt files

Corrupt files can result in MediateVR not running or certain features not working. This is likely the result of missing files or out-of-date code. To remedy this, we recommend deleting and reinstalling the project from GitHub. Issues that occur due to further development are the sole responsibility of the new developer. However, the development team are available to contact for limited assistance. Their Durham contact details can be found at the top of the document.

2.6.8 MeditateVR will not run

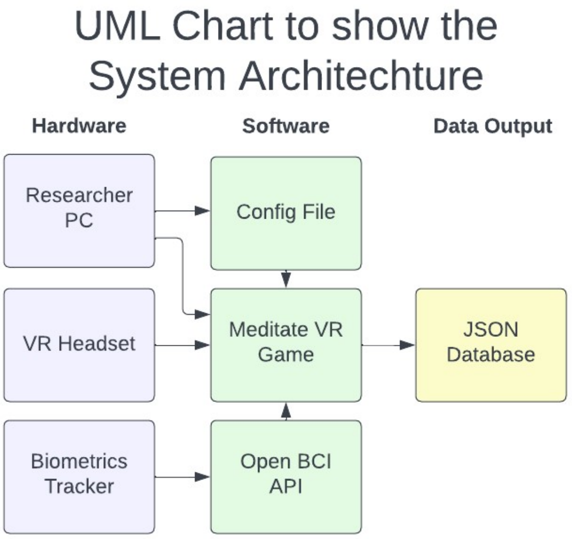
Should any unforeseen errors arise, we recommend first closing and reopening the project. If issues persist redownloading the project may resolve these problems. If the problem remains unresolved, this may indicate a deeper technical issue. In this case, please reach out to one of the members of the team and we would be happy to help resolve any issues.

3 System components

This is a technical section, designed to assist future developers with any changes or improvements they wish to make with the product in future. It goes more into depth about the system and why we made certain decisions during development. It outlines how different components of the system function so that future developers can have a greater understanding of how the system runs.

3.1 Architecture Overview

The system is broken down into 2 main programs, the biometrics API as well Was the Unity game, as shown in the diagram below.



The main parts of hardware for the project are:

- The PC that the researcher uses that will run the API the Meditation Game and store the config file as well as the JSON Database. The PC will run all the processing for the game.
- The VR Headset that the user will wear. This will be connected to the PC as shown in the user setup and is what the user will wear whilst they are within the meditation. It is needed so that the user can interact with the meditation environment.
- Biometrics tracker, the BCI headset that the user will wear whilst they are within the meditation. This headset sends the data to the PC to the Open BCI API which then sends the data into the program. It is needed to gather data about the user's meditation to update the tree as well as be stored within the JSON Database.

The Software parts of the system are:

- The config file is a .txt file with instructions as well as variables that the user will edit to configure the program, the configuration of each experiment is stored in the JSON database.

- The Open BCI API Is an application that connects to the BCI headset and gathers ECG data about the user wearing the headset. The data includes relaxation and stress levels amongst other measures. The data from the headset is sent live to the Unity program through the live data stream on the localhost server.

- The Unity game is the main program which brings together the data and VR Headset to give the user the immersive meditation experience. It takes the biometrics data and uses that to change the growth rate of the plant. The program also displays the meditation environments as well as breathing guidance to the user. The program then gathers data about where the user is looking and what their biometrics are and stores that in the JSON database.

An overview of the file system for the Meditate VR Unity project is shown below. Some folders have been hidden as they are not used for development. The highlighted files are files used to access the scenes.

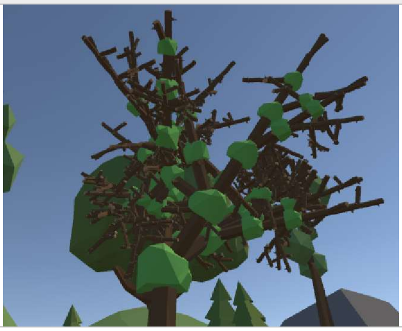
File System Overview

```
.
└── Unity/
    └── MeditateVR/
        ├── Assets/
        |   ├── Audio
        |   ├── ForestEnvironment/
        |   |   ├── Scenes/
        |   |   |   ├── ForestEnvironment
        |   |   |   └── ...
        |   |   └── ...
        |   ├── IslandEnvironment/
        |   |   ├── Scenes/
        |   |   |   ├── IslandEnvironment
        |   |   |   └── ...
        |   |   └── ...
        |   ├── Materials
        |   ├── MeditationLog
        |   ├── Scenes
        |   ├── Scripts
        |   ├── Settings
        |   └── ...
        ├── Library/
        |   ├── APIUpdater
        |   └── ...
        ├── Logs
        ├── Packages
        ├── ProjectSettings
        ├── Temp
        └── UserSettings
```

Most development was performed within the Assets folder as this is where most code and environment elements are stored.

3.2 Plant L-system

To provide instant feedback on the success of a user's meditation session, the system uses an L-System to model a tree that the user can watch grow as they meditate. This tree will then use the information from the Emotive Epoc X to grow more if the user is having a more relaxing session. This allows the user to gain a sense of accomplishment from a successful session. By using an L-System, we have made the plant's growth more granular and organic.

Tree Growing during a meditation session		
Beginning of meditation	Halfway through meditation	End of Meditation
		

L-Systems are a mathematical system, used to procedurally generate realistic looking models of plants. A guide to how they work can be found here [2]. This specifically is a 3d implementation of one, using a tree-based implementation (graph theory trees rather than biological trees). These trees are done using unity's transforms class. Below is a list of the classes required to create this System.

3.2.1 LSystem

This class is used to define the basic behaviour of an LSystem. Its purpose is to both provide access to parameters for other classes, as well as to provide the template function BasicTree that can be used to make the tree in the photos seen above.

Parameters

Name	Explanation
myController	Holds the reference to the script that started this LSystem. It is used to get variables about both the LSystems configuration along with the starting gameobject of the LSystem.
axiom	The LSystems axiom (What the LSystem starts with)
rules	The LSystems rules (A dictionary that describes how the LSystem should grow. Read the guide on LSystems for more details)
lystemGrowthValue	The value that denotes the LSystems growth. This is different from the way LSystems usually work as we wanted to make the growth more organic and varied. Instead of preforming a set number of iterations of the rules over the axiom, this system will use the rules to "grow" each part of the plant after a random amount of "growth".
maxGrowthValue	This is the limit on how much the plant can grow. High values may create performance issues on systems.
growthRate	This is how much to increase the growth value with each growth. It can be used to control how quickly the plant grows.

Methods

Name	Explanation
LSystem	This initialises an LSystem.
BasicTree	This method returns the LSystem used to create the Tree that is currently used in meditation sessions. It does this by setting values for the axiom and rules.
GetNewBranchRotations	This function is used by the LSystem to get the rotations for new branches coming off the tree. This is complex as it both needs to evenly space these branches and not return rotations that means the tree will grow unnaturally.
Grow	This is the function used to increase the growth of the tree based on both the growth rate and user's relaxation.

The following are the classes that make up the LSystems Alphabet. These are the different "operations" that the LSystem can use to grow.

Class Name	Explanation
LSystemSymbol	This is the base class that all other alphabet classes are inherited from. It is used to store variables that all symbols need. This includes the information needed to determine what the Symbol will become as the plant grows. It also has details of the prefab for the Gameobject that will be created when the plant is instantiated.
Branch	This is the symbol used for creating a branch.
Leaves	This is the symbol used for creating leaves
Scale	This is the symbol used to tell the LSystem to increase the size of the plant. In practice this is done by creating an empty game object with a different scale. Because this is an expensive operation, rather than following the traditional LSystem paradigm and repeatedly replacing this scaled with a larger one to create growth, this scale also adds an instance of the LSystemGrowthController to this empty gameobject. This makes the empty gameobject grow to full size as the tree grows.
CreatedGlobalRotation	This is the symbol used for changing the direction of growth. (Making a branch grow to the side rather than only ever straight up).

The following classes are unity scripts (they are inherited from monobehaviour and attached to gameobjects).

3.2.2 LSystemController

This script is attached to the LSystem prefab. It creates the first LSystem using the BasicTree function. It's starting parameters can be tweaked in the unity editor to change the properties of the tree.

Parameters

Name	Explanation
initialGrowthValue	This is growth value that the LSystem should start with.
maxGrowthValue	This is the growth value that the Lsystem should stop at.
growthRate	This is the rate at which the LSystem should grow.
myLsystem	This is the LSystem that is being created.

Methods

Name	Explanation
Start	This creates a basic tree instance of an LSystem.
FixedUpdate	This tells the LSystem to grow.

3.2.3 LSystemComponent

This script is attached to every gameobject that the LSystem creates. Each is based on an LSystemSymbol. It is responsible for making each of the components grow by replacing the gameobject with the next set of gameobjects as defined by the LSystem's rules.

Parameters

Name	Explanation
myLsystem	This is a reference to the instance of the LSystem class that this LSystem is created from.
growthForNextStep	This is the value that the growth in the LSystem class needs to be at for this component to grow.
destroyOnGrowth	This determines whether the component should destroy itself when it grows. This would be used if you were replacing the component with something else. For example, if you wanted the tree to have flowers that grow into seed pods then the flower would need to be destroyed.
grown	This says whether the component has already grown.
nextStep	This is the string used to determine what the component grows into based on the LSystem rules.
getNextStep	This is the function that the component could use to determine what it grows into. (Rather than the nextStep string and LSystem rules).

Methods

Name	Explanation
Start	This function is run when the component is created and sets up the component.
Update	This function checks if the component should grow every frame. If it should it runs the Grow function.
Grow	This function grows the LSystem component by creating all the new symbols that the component needs at growth. In then creates a gameobject and LSystemComponent class for each of those Symbols.

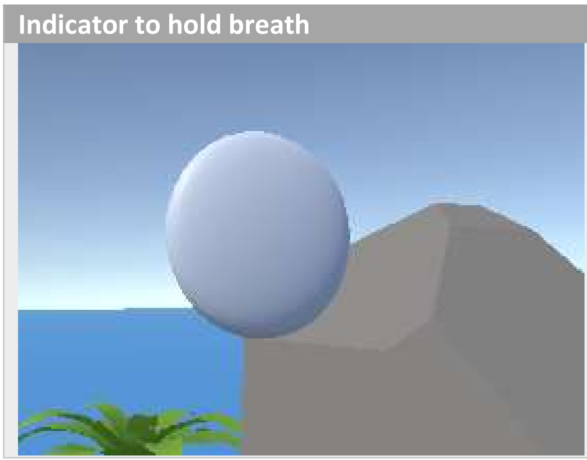
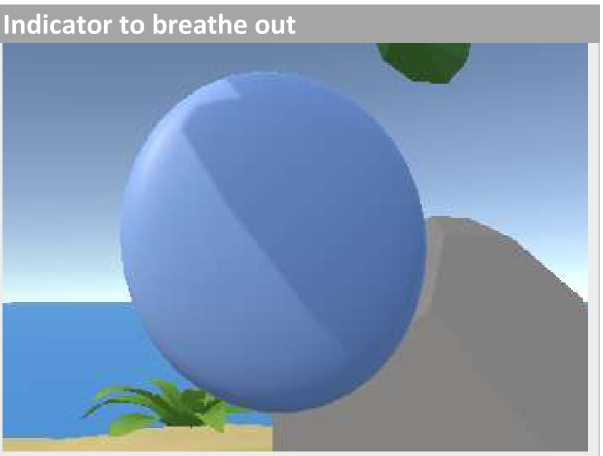
3.6 Config system

The Config system allows the researchers running the program to choose the environment as well as configure different elements of the environment. It works by parsing the config file into a public dictionary from which all parts of the game take their values. The methods for setting variables of the config are shown below.

Method	
SetBreathingPattern	Sets the breathing pattern to what is defined in the config, chooses 4-4-4 breathing if nothing else is chosen.
SetEnvironment	Sets the Environment to what is defined in the config, chooses Forest if nothing is chosen.
SetBiometric	Sets the Biometrics tracking to be what is defined in the config. Sets default value Biometrics if nothing is chosen
SetPlantGrowthRate	Sets the growth rate of the plant growth system to be what is defined in the config. Sets default value 0.5 if nothing is chosen
SetDetail	Sets the detiall level of the environments, default is Low
SetMusic	Sets the Music to be on or off for the environments, default is off

3.7 Breathing guide system

The Breathing guidance system is a visual breathing aid that grows and shrinks to guide the users breathing to a certain pattern. The patterns can be chosen in the config file are Box breathing (4-4-4-4), 478 breathing (4-7-8-4) and 424 breathing (2-4-2-4). These timings are preprogrammed into the system however custom timings can be made in the VisualGuidance script In the scripts folder. These timings are common breathing techniques for calming and relaxing.

Indicator to hold breath	Indicator to breathe out
	

it works on all objects with the Visual Guidance script attached to them and will cause the object to grow and shrink according to the breathing rhythm.

3.8 Music

The meditation environments have different relaxing music that plays in the background to calm the user and enhance their meditation, they can be configured in the configuration file. The configuration file turns all music on or off. If the researcher wanted to get rid of an individual track, they would have to remove the specific track from the scene code.

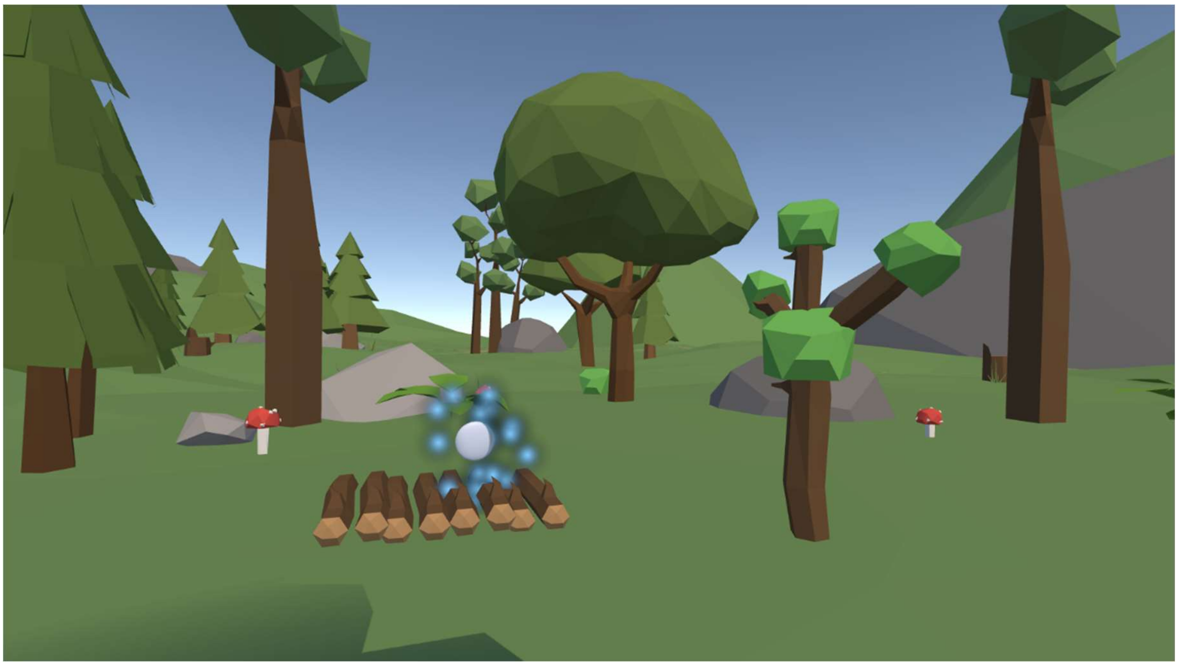
Music Track	Forest scene	Beach scene
forestSound	Yes	No
generalMusic	Yes	Yes
oceanSound	No	Yes

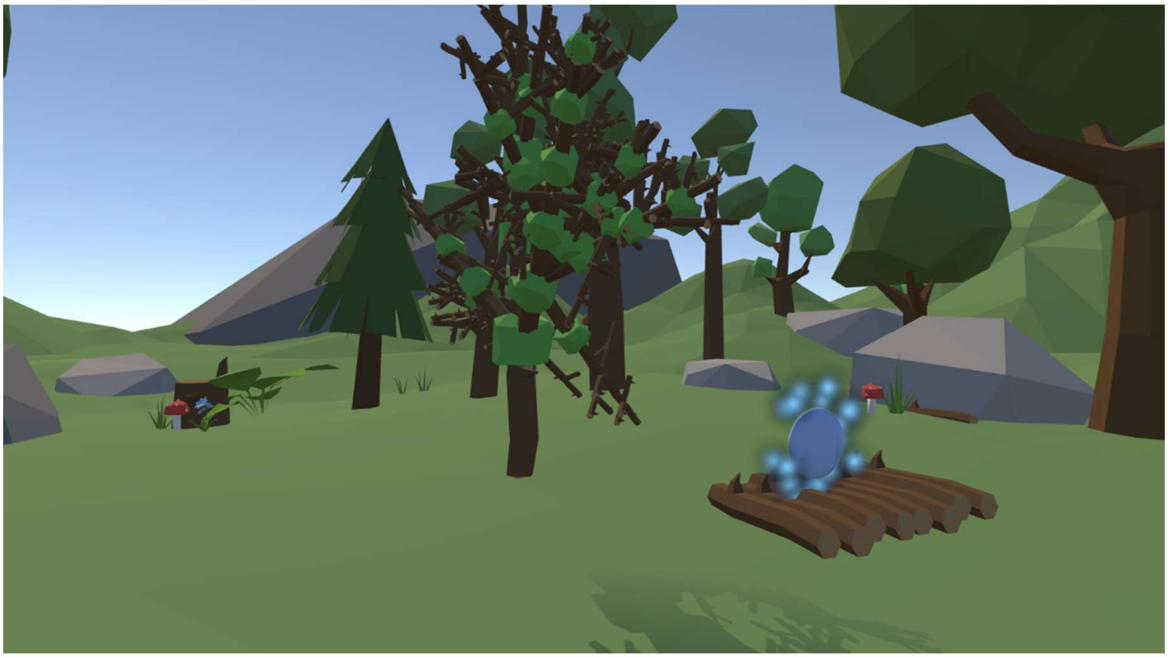
3.9 Environments

There are multiple meditation environments that the researchers can select for the user. The idea of the two different environments is so that the researcher can see which environments are better for meditation.

3.10 Forest

In the forest environment, the user is situated on a pile of logs in the middle of a clearing. The user can look around freely to see the trees, rocks and other plants around them. Here are two different views from the user's perspecitve.





There are a variety of different trees and shrubs as well as rock shapes and mushrooms. This aims to create a diverse environment, so the user has many different shapes to concentrate on. In both pictures you can see a grey/blue ball near to the log pile. This is a visual breathing aid which helps the user to control their breathing. The breathing pattern can be changed in the config file, with the default being 4-4-4-4 box breathing. This is positioned to one side of the log pile so that the user can choose whether they want to use it or not. There is also a tree (not shown) that grows while the user is meditating. This will help to relax them as they see the tree growing as they get calmer. In this environment, you can see hills and mountains in the background.

3.11 Island

In the island environment, the user is situated on some wooden planks in the centre of the island. The user can see the features of the island and the ocean as it reaches the horizon. Here are two different views of the island, one from the users position and the other from further away.

The island consists of a grassy centre with a beach nearer the ocean, there is a wooden path into the water as well as some debris on the beach. As discussed in the forest environment, there is a grey/blue ball near to the user's position which can be used as a breathing guide. Similarly, there will be a tree that grows according to the user's relaxation. In this environment, you can see the water meet the horizon in the distance.

3.12 BCI system

In order to gain a deeper understanding of the user's state of mind during the meditation, the solution uses the Emotiv Epoc X to observe the users. This device can record the user's brainwaves and use these to determine 6 performance metrics. These metrics include relaxation, attention, interest and excitement. The majority of these are not used by the system but are stored for researchers to look at later. The relaxation level is used to control how quickly the plant grows.

These performance metrics are calculated in the program that is used to control the Emotiv Epoc X called Emotiv BCI. The system then uses the BCI-OSC extension to send these metrics using the Open Sound Control Protocol over UDP to the Unity game. This means that the Epoc X could also be setup on a different computer. Unity receives this data using the Rug.OSC nuget plugin. Below, are details of the class BCIManager used to receive these performance metrics in Unity.

Parameters

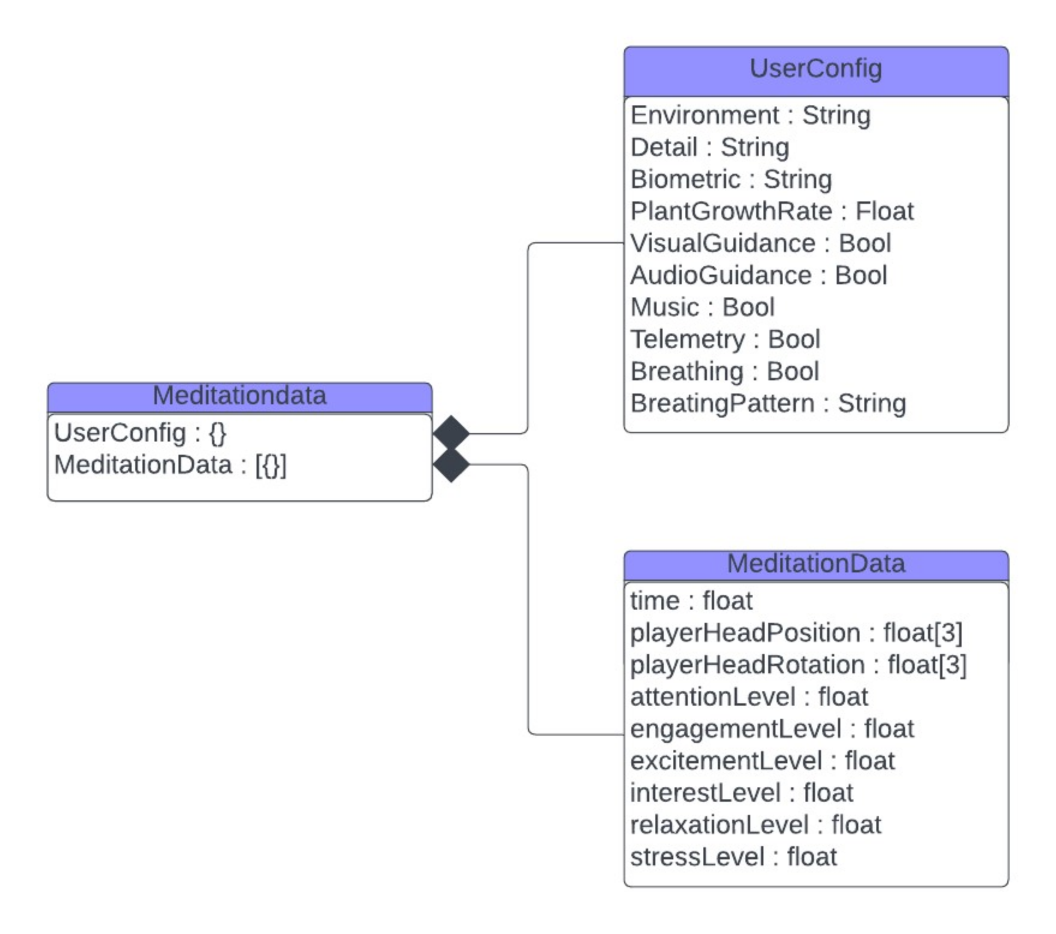
Name	Explanation
receiver	This is the instance of rug.osc's OscReciver class that manages receiving packets from open sound control streams.
recivingThread	This is the thread that the class uses to receive the stream. Because it can take a while to receive a packet, this process must be done in a seperete thread. This stops the game from having to wait regularly for long periods of time.
port	This is the network port that the reciver is receiving packets on.
listener	This is the instance of rug.osc's OSCAddressManager class. This class takes packets that are received from the receiver. As defined in the Start function, if each of the data streams in these packets have a certain address, it runs the relevant function. In this case, it runs a function for each performance metric that it receives.

Methods

Name	Explanation
Start	This function initialises the BCI manager. It does so by creating a listener and receiver. For the listener it attaches the SetPerformanceScore metric for each of the data streams for performance metrics.
Update	This function is run every frame, and it checks if an OSCPacket has been received. If so it starts listening for a new one.
ListenForPacket	This function is how the class listens for a packet. It does so by creating a new thread in receivingThread. This thread runs the GetOSCPacketFunction.
GetOSCPacket	This function uses the reciver to recieve for a packet. Once the packet is received, it gives it to the listener object, so that OSCAddressManager can process the data.
SetPerformanceScore	This function is run when the listener object sees a performance metric in OSCMessage's streams. It then sets the relevant performance metric parameters.

3.13 Data storage

The Data stored is the configuration, and meditation data as the program goes on. The Schema for the database is shown in the table below.

JSON document Schema


There are 2 objects:

- UserConfig which stores all of the config for the program when it was run.
- MeditationData which stores all of the metrics at any given time.

3.14 Testing results

In this section we will outline some of the user acceptance tests we carried out.

3.14.1 Sound Test

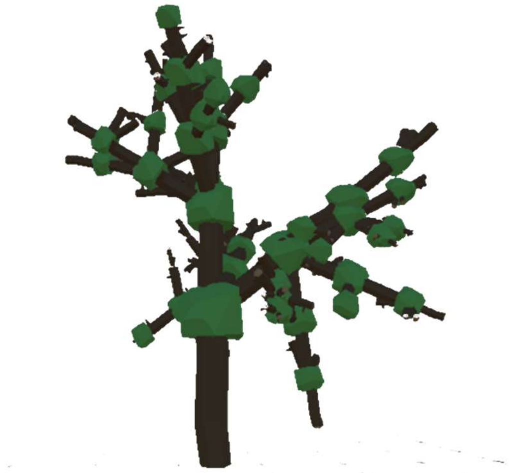
We asked 20 university students to rank on a scale of 1-5 stars on how relaxing the music in our forest environment was. The responses were:

1 Star	2 Stars	3 Stars	4 Stars	5 Stars
0	2	3	8	7

This gives us an average rating of 4 stars which is what we required in the test plan report. Moreover, we asked them to say whether they preferred the music alone, or with the forest and rain noise as well. 70% said they preferred the music with the rain noise. The overall feedback on the forest environment audio was positive, meaning we passed NFR 3.2 and NFR 3.2.

3.14.2 Tree Appearance Test

We asked 20 university students to pick which plant our growing tree most resembled. Their responses were:



Tree	Bush	Flower	Other
19	0	0	1

We asked the person who responded other to say what they thought it resembled and they said that it looked like a mushroom. This indicates that the tree was largely easily identifiable which passes the initial test plan.

3.14.3 Game Cohesiveness Test

We asked 20 university students to pick the odd one out of three scene pictures. Two of them were our scenes and one of them was of a previous project scene. We recorded how many votes each option received.

Island Scene	Forest scene	Previous scene
0	0	20

This test result shows that our game is cohesive (and therefore passes NFR 3.1) as every student was able to identify which one we had not created.

4 Future Development and Ethical Considerations

4.1 Possible Future Development

Through development of the project, we have tried to implement the features we feel enhance both meditation experience and research usefulness as much as possible and have focused our efforts on implementing these features. During development there were many additional features we thought would enhance the software but did not have time to implement or we decided could affect the quality of implementation of our other features. Some of these possible future ideas are outlined in this section.

4.1.1 Multiplayer Meditation

A feature we thought would be interesting to implement, as well as what Professor Cristea outlined in the spec was that of a simultaneous multiplayer system. We had many ideas of how this could be implemented, including a Meditation Leaderboard, Garden where you could see other people's meditation sessions plants and a simultaneous multiplayer in which 2 users are in the same environment, meditating at the same time. We did not choose to include the leaderboard and garden as we thought they would take away from the research use as well as the way it could stress users in trying to outdo other user's meditation. We chose not to pursue the live multiplayer as we thought it would take too much time to implement and we were unsure of the best way to approach cooperative meditation.

4.1.2 Heartrate tracking system

During development, we planned to implement both the brainwaves and heart rate tracking systems to provide researchers with the most amount of data possible. We found that the easiest way to implement the heart rate tracking system was to use a Fitbit supplied by Professor Cristea. The problem with this was that the Fitbit required a specific account on a mobile phone as well as a server to connect to the Fitbit API. We made the decision to not use the Fitbit due to these characteristics, as it went against our idea of having the system being easy to set up and requiring minimal external devices. Another possible way to track heartrate was using the heartrate tracker with a chest strap, but we decided against using it as many women who had previously used the tracker were uncomfortable with using the chest strap. There could be ways for these systems to be used in future development, or other systems could also be used. There are also other biometrics that can be used for measuring relaxation, such as body temperature and breathing patterns which could be interesting to experiment with.

4.1.3 Data Analysis System

We have developed a system to store data in a JSON database, but we also floated the idea of having a live dashboard of all the biometrics as well as environment settings so that researchers can have a live look into how the meditation is going. We have not included it as we felt it was not crucial to the project, but it would still be a useful tool for research and would be a valuable addition to the project for future development teams.

Environment Params
Tree Growth
Game View
Heart Rate
Emotion Levels

4.1.4 More Environments

We have created several low poly simplistic environments, but possible future developments could introduce more environments with different varying levels of abstraction to research how abstraction affects meditation. On top of that the existing environments could be made to be more interactive and have more depth.

4.1.5 Implementation Of more Gamification elements

As outlined in Professor Cristea's research[source] there are many gamification elements that could be implemented in various forms into the meditation project. We implemented Progression through the plant growth system, sensation through the VR and progression through the tree. We encourage future developers to read through Professor Cristea's research [1] and come up with new ways to implement these elements.

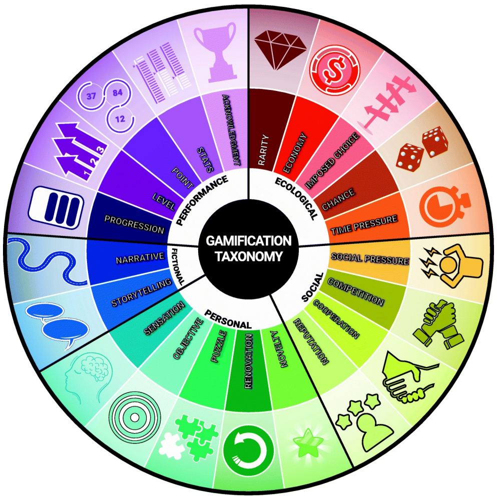


Figure 1 Gamification Elements taken from [1] Fig2

4.1.6 Create new plants.

Whilst conducting research, a researcher may be interested in testing the effects of different plants on the user's relaxation and satisfaction. The LSystem can be changed to create many different plants, from shrubbery to visually appealing 3d shapes. By changing the LSystem, a researcher could then see if these plants would influence the user. A much more complicated plant may create more stress for the user, while a simple one could lead to a loss of interest.

4.2 Ethical and Societal Impacts

There are many potential ethical and societal issues that our software could have due to the external software we are using to monitor brain waves and the potential for future multiplayer modes. It is important that we stay aware of these issues so that we (or future developers) can avoid creating software that is vulnerable to discrimination and societal problems.

4.2.1 Biometric issues

One major potential issue that has been identified is that there may be problems with the previous groups biometric data tracking system. During the design phase of this software implementation, it was decided that the method of biometric data tracking used could not be done using a chest strap. This is because in previous projects it was found that some users felt uncomfortable wearing these due to how they must be put on. As a result of this ethical concern, we decided to use a BCI brainwaves headset for our primary data collection with the possibility of a watch-based heart rate monitor as a secondary data collection method.

Another potential issue is that the external BCI software could have ethical problems with it. We will make sure that it does not have any ethical problems with it as of the product handover but as the company making the BCI software update and 'improve' their product, there is a possibility for unethical decisions to be made. For example, the headsets may only be tested on males and so they may not fit the female head shape as well which would discourage females from participating in the meditation sessions. This would be a major problem, so it is imperative that the researchers make sure that this does not become an issue by looking at ethical reports on the BCI provider.

4.2.2 Accessibility

One accessibility problem is that some people find VR games to cause nausea. This is a major problem as if a meditation game nauseates a person, then it means that it is not worth using as it is very hard to be relaxed whilst feeling sick. We have identified this as a problem and addressed this problem in our requirement specification. We require the game to stay above 60FPS. This benchmark is widely used as the threshold for causing nausea [3] so staying above this will help to mitigate the issue. Despite this, some users still report nausea regardless of the framerate which means that they will not be able to use this app to its full potential.

Another accessibility problem is the cost of the equipment. The combined cost of the VR headset and the BCI headset is around £1,000. This means that the product is not accessible to a large portion of the public. However, this product is intended as a piece of research software. This means that cost is not a major factor as the researcher already has all the equipment. If this research was scaled up to use 5 or more different headsets, then it might be best to change to a less expensive biometric tracker, but seeing as only one person will use it at a time, the results make the BCI headset worth the cost given its ability to measure detailed brain metrics. Similarly, if the product was released as a downloadable game, it would be best to switch the biometric tracking to a cheaper alternative, for example, a heart rate tracking watch.

Some users may struggle with the ease of use of the equipment. The BCI headset is difficult to set up and the ability to read data is dependent on the user's head shape and haircut. The software also requires the ability to turn your head to look around the environment. It is quite hard for a user to put on all of the necessary equipment and most likely will require a second person to help attach the headset and BCI. This is mainly an issue with VR headsets in general and not a problem with our software specifically, so we decided to include in game features which require the use of hands because it is unlikely that someone without the use of their hands would buy a VR headset.

4.2.3 Multiplayer

In our future development guide, we have identified that a multiplayer game mode could be added to this software. This has a lot of ethical issues tied with it.

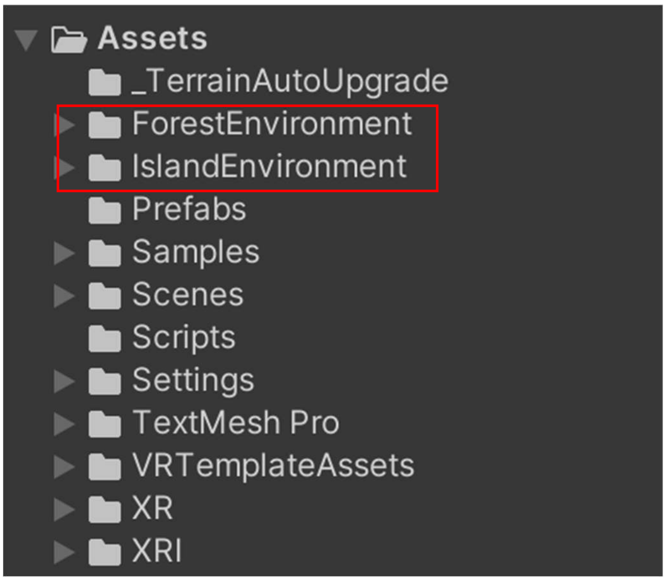
As soon as multiplayer interactions are allowed in a game then online bullying and harassment becomes an issue. There are a few ways to get around this issue. One way is to block communication between players online and only have the feature of meditating with other players present. This is not a particularly good solution as it would mean that players cannot congratulate others on good meditation. Another solution is to limit players' communication by only allowing them to say a set of phrases, for example, "good job" or "you're doing great". This is quite a good solution as players could get the benefits of meditating as a group with positive reinforcement. A final solution to this problem would be to have open communication with censorship on certain words. This is also a very good solution as players sometimes find communication through a few select phrases irritating. The problem with this is that it is nearly impossible to censor all harmful or upsetting words.

This decision would be made by the future developer, but it would be advised that it be the second or third option. Each of these have their advantages and disadvantages but it is imperative that bullying, harassment and online abuse will not be able to happen in a multiplayer environment.

5 System Maintenance

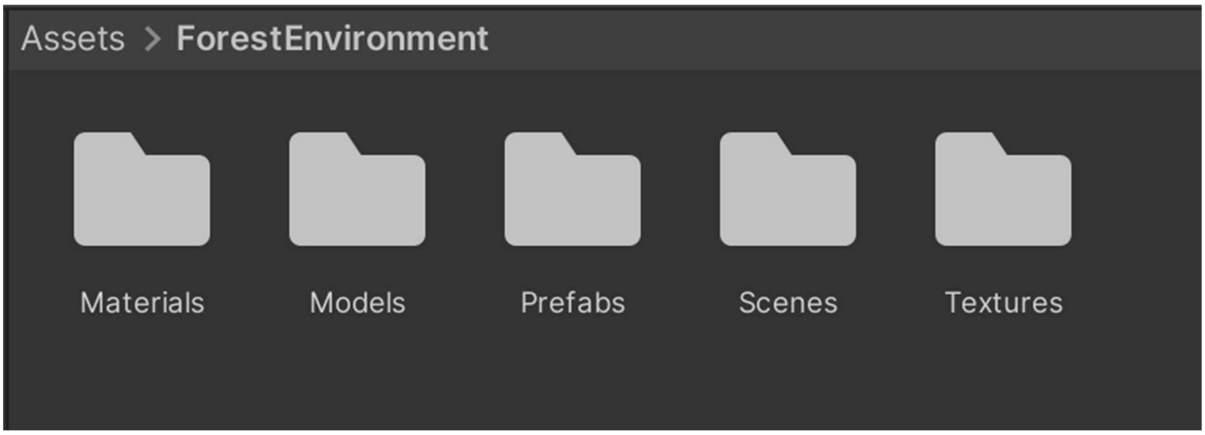
5.1 Meditation Environments Maintenance

The scenes for the two mediation environments are stored in the assets folder of the Unity project. Both the Island and Forest environments have their own folder that holds all the assets for the scene and the scene itself. If desired the scenes can be updated or even new scenes added to allow the researcher to test the effectiveness of different environments on the quality of the end user's meditation.

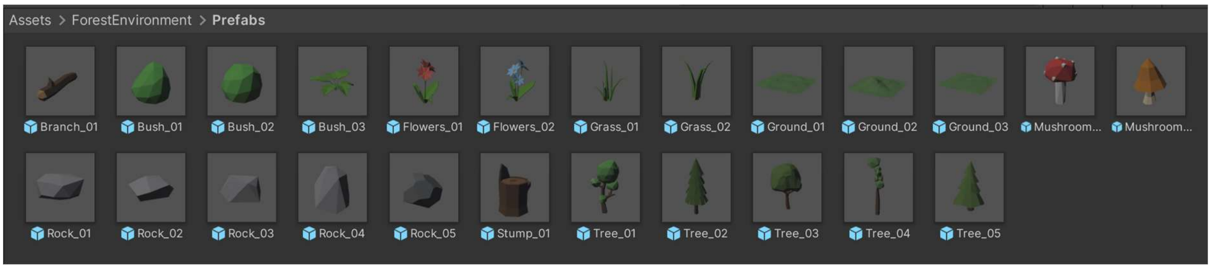


Assets
_TerrainAutoUpgrade
ForestEnvironment
IslandEnvironment
Prefabs
Samples
Scenes
Scripts
Settings
TextMesh Pro
VRTemplateAssets
XR
XRI

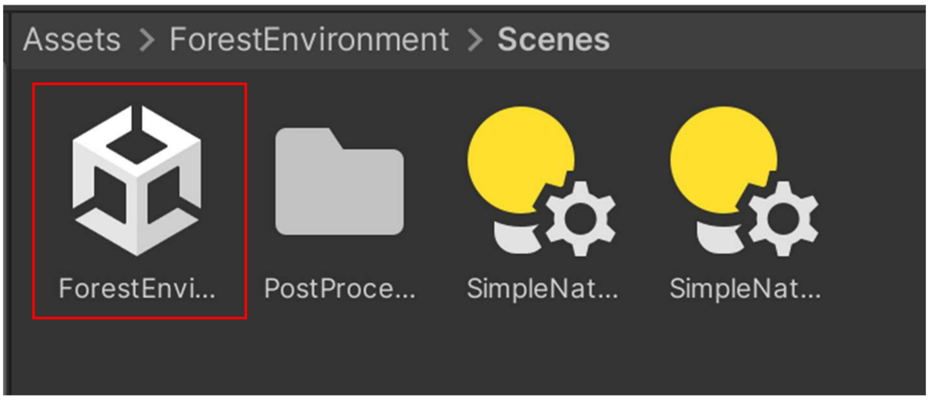
Marked above are the two folders that hold all the files for their respective scenes. In the event a new environment is to be added the developer should create a new folder ending in the word “Environment” to follow current naming conventions.



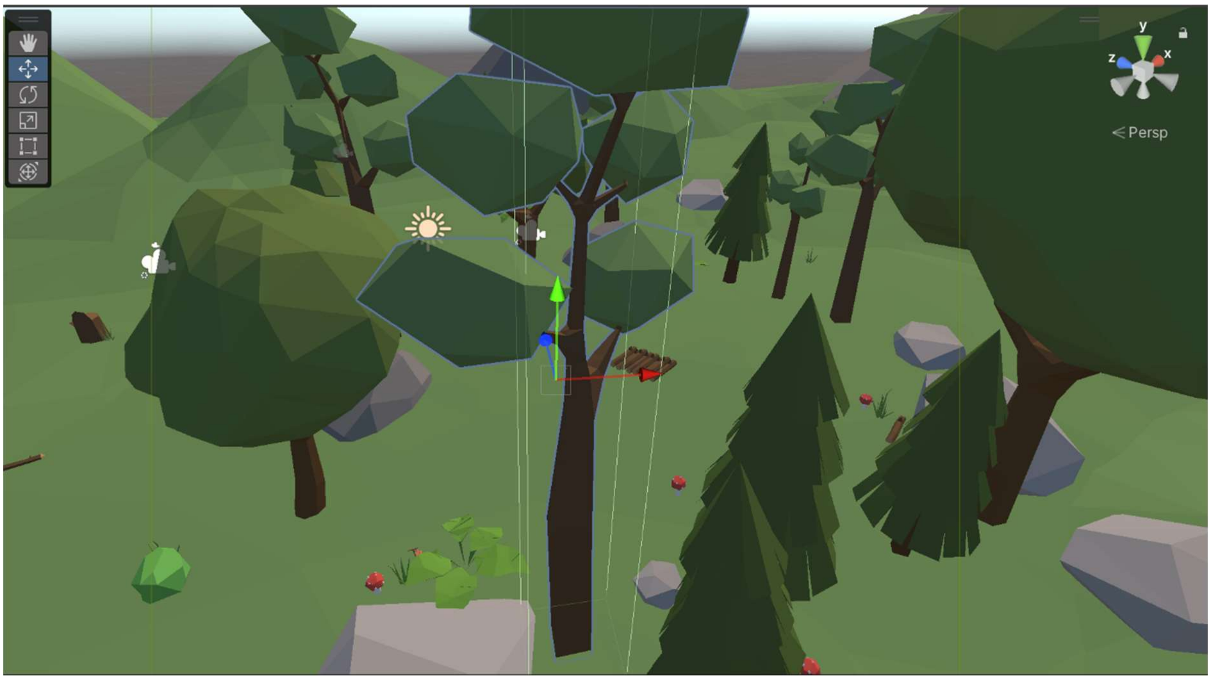
Inside each folder, there are folders for assets and a scenes folder, this is where the main scene is held. If changes are to be made to an existing scene the developer would be advised to familiarise themselves with the different prefabs available in the asset pack. As both of the default scenes were taken from pre-made asset packs they offer a wide range of models that can be added to the scene.



The above photo shows all of the prefabs included in the Forest environment pack. Not all prefabs have been used for the forest and island environments so utilising them could be an effective addition. There is also the possibility of including other prefabs from different asset packs. A good resource for assets is the Unity Store, which has a wide range of assets, many freely available that can be downloaded and added. The two environments included with our build are both constructed from low-poly asset packs in order to create a consistent theme and feel between the two environments. If adding another scene, the developer could look to create a scene that does not follow a low-poly structure and analyse how this affects the user's meditation experience.



The scene can be accessed inside the scenes folder. Once opened unity will display the scene and allow for assets to be added or moved.



Assets can be moved by selecting the asset and then using one of the 3 directional arrows inside the Unity scene editor. If more precise positioning and sizing are required, then the right-hand side inspector panel can be used.

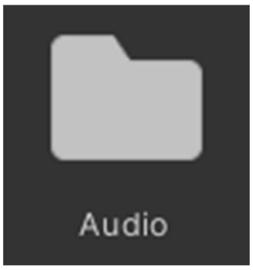
5.2 User Configuration Maintenance

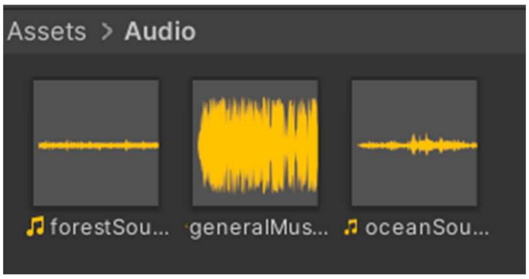
The User Configuration operates by opening the file config.txt from the assets folder within the UserConfig script it then parses the file ignoring all comments and stores the values of the config variables in public variables that can then be accessed by each system.

If new variables are put into the config file they will be parsed however they will not be saved. In order to save new variables, you will need to declare new public variables and then pass the values that are parsed into these variables.

5.3 Music and Atmospheric Sounds Maintenance

The music and atmospheric sound system operates by using an audio source to make noise in the user's environment. The MP3 files are stored together in the Assets/Audio folder and are named appropriately.





The main danger to the maintenance of these files is corruption. In this case, the best course of action would be to download the files again off the GitHub page. The developer would then copy the files to the location of the corrupted ones and replace them. If the developer decides to change the music or atmospheric sound or decides that they want to layer more audio sources into the environments, they can download more MP3 files from the website https://pixabay.com/music/search

After testing, it was decided that the best position for the audio source was in the centre of the environment because when the audio source was offset, the volume would change based on the direction the player was facing. To some users, this created an eerie effect, so we decided to place the source in the centre. In future development, movement around the environment may be added. In this case, we would recommend attaching the music sources to the player to avoid the previously stated problem. Another potential problem that a future developer may encounter is if they want to add bird sounds into the trees. We would recommend that the developer position the audio sources inside the tree assets so that the directional audio would seem accurate to the user. We would also recommend that the developer makes backups of the audio files to help counter corruption.

5.4 BCI Maintenance

The majority of the computation required for analysing the user's brainwaves is done in an external program (Emotive BCI) there is not much maintenance required. This also means it is very simple to change certain parts of the BCI system.

As long as it is still compatible with the Emotive BCI software, it is extremely simple the change the BCI device. Follow the same startup procedure described above, this time using a different device.

The headset can also be made to send more data to the unity program. The Emotive BCI app also supports sending Facial expressions and mental commands. To do this, you merely also select these data streams in the OSC tab. You would then also need to attach more functions to addresses in the BCIManager's listener object. These functions would then run when the listener detects those data streams.

5.5 Plant Growth Maintenance

LSystem trees can be very easily added to scenes. As long as there is a StaticGameManager prefab present in a scene, all one needs to do is drag the LSystem prefab in /Assests/Prefabs into the scene and then set the relevant LSystemController parameters.

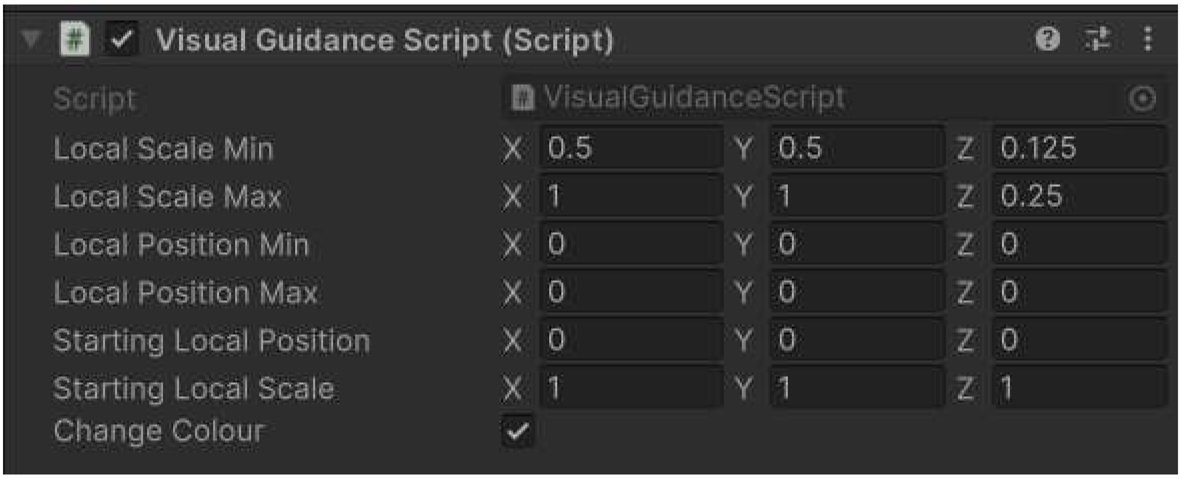
The plants that are generated using the LSystem can also be changed. The BasicTree function defines the plant that all LSystems use, however, another function could be created to make say a fern or bush. To do this one would only need to change the starting axiom and rules.

The assets used to great the graphics of the tree can also be very easily changed. Each prefab needed for the Tree is currently stored as parameters of the LSystemPrefabAlphabet script attached to each StaticGameManger prefab. These prefabs can either be swapped out for other prefabs, or themselves edited to change how the tree looks.

5.6 Meditation Guidance Maintenance

The meditation Breathing Guidance system operates off a custom timing loop and creates a float called scale, which stores the scale of the breathing system as a value between 0 and 1. This value Is 1 when the breathing is to be held and 0 when breathing is to be empty, with sinusoidal transitions between these values for breathing in and out. The script for this can be found in the scripts folder and can be attached to any object in the Unity space. The script causes the object to grow and shrink according to the scale variable as well as change colour from light blue to a darker shade of blue.

In order to attach the script to a given object navigate to the attach scripts and add the script to the object.



Visual Guidance Script (Script)

Script	VisualGuidanceScript		
Local Scale Min	X 0.5	Y 0.5	Z 0.125
Local Scale Max	X 1	Y 1	Z 0.25
Local Position Min	X 0	Y 0	Z 0
Local Position Max	X 0	Y 0	Z 0
Starting Local Position	X 0	Y 0	Z 0
Starting Local Scale	X 1	Y 1	Z 1
Change Colour	☑		

The breathing times can be configured in the config file or through the array `StaticGameManager.MainManager.breathingPattern` the timings are breathing in, holding, breathing out, holding.

6 Conclusion

Thank you for reading our user manual and using Meditate VR. We hope this manual has provided you with clear guidance on how to set up and use the program effectively.

If you have any questions, concerns, or ideas for further developing the project, please do not hesitate to contact any member of our team. We would be happy to assist you and welcome your feedback to improve our product.

References

[1] Toda, A.M., Klock, A.C.T., Oliveira, W. et al. Analysing gamification elements in educational environments using an existing Gamification taxonomy. Smart Learn. Environ. 6, 16 (2019). https://doi.org/10.1186/s40561-019-0106-1

[2] www-archiv.fdm.uni-hamburg.de. (n.d.). An Introduction to Lindenmayer Systems. [online] Available at: https://www-archiv.fdm.uni-hamburg.de/b-online/e28_3/lsys.html [Accessed 6 Mar. 2024].

[3] Louis, T., Troccaz, J., Rochet-Capellan, A. and Bérard, F. (2019). Is it Real? Measuring the Effect of Resolution, Latency, Frame rate and Jitter on the Presence of Virtual Entities. Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces. doi:https://doi.org/10.1145/3343055.3359710.