Plymouth University

School of Computing, Electronics and Mathematics

PRCO304

Final Stage Computing Project

2018/2019

BSc (Hons) Computing with Games Development

Liam West

10522700

Astray

Acknowledgements

Firstly, I would like to thank Dr Dan Livingstone for his patience and supportive, critical feedback when I needed it most.

Secondly, I would like to thank all members of the ISS team for an incredible journey on the CGD course.

Finally, my family and friends for being supportive throughout my time at University.

Abstract

The document includes a retrospective on the development of a survival game representing an individual's mental state and wellbeing, which has been accomplished through the immersive techniques of visual and auditory features.

An investigation was conducted based on how games could be used as a therapeutic medium for sufferers; once knowledgeable on the symptoms and treatments, the data was transcribed into gameplay mechanics within a fictional world.

Comparisons were drawn between most appropriate software packages, methodologies, and architectural decisions. The report also includes how important issues regarding legal, social, ethical and professional were factored into the development process. Finally, the paper contains a detailed explanation into the weekly and bi-weekly sprints leading up to the final release, that is followed by a conclusive analysis of the outcome of the project, which far differs from the initial scope and objectives defined in early development.

Table of Contents

1. Word Count

10,634

2. Code Link

Plymouth University GitHub Repository:
https://github.com/Plymouth-University/prco304-final-year-project-lwest2

3. Introduction

An individual’s mental health and wellbeing is undoubtedly an essential topic of discussion, and the gamification of such, can educate those with a lesser understanding of mental illness as a whole, or another’s viewpoint on the world. (Flemming, T. M. et al., 2017) [1], emphasises six-game categories used as mediums for psychological and cognitive therapy; ‘… exergames, virtual reality, cognitive behaviour therapy – based games, entertainment games, biofeedback, and cognitive training games…’. All of which achieve different supportive features for a diverse range of mental health problems. The author’s proposed idea was initially described to be a gamified representation of an individual’s mental state, rather than a coping mechanism for sufferers, however with (Flemming, T. M. et al., 2017) [1] statement in mind, a game focused around mental health would significantly redirect focus of ‘cognitive resources’, and would idealistically act as therapeutic escapism from their perception of reality. This is perfect for a game with the context of psychosis, a mental disorder involving hallucinations, delusions and the ‘loss of contact with reality’ (Nimh.nih.gov, 2019) [2]. This is aimed towards an audience who would like to experience a gamified representation of psychosis, or an audience seeking escapism through entertainment-based games.

The initial aim was to create a playable demo with a focus on mental health; a fully immersive experience through a spatialized auditory environment, alongside binaural recordings to further enhance psychosis. In addition to this, the visuals should withstand a high-level of fidelity towards realism, which has been achieved through the use of Unity3d’s experimental High Definition Render Pipeline (Unity, 2019) [3]; a tool built upon Unity3d, providing advanced volumetric lighting techniques and intuitive features such as Shader Graph, and Visual Effects Graph (VFX).

The original scope and functional requirements of the project differ drastically from the final iteration of the project. However, the first minimum viable product (MVP) was devised to be as follows. A split narrative whereby the character must make decisions that could alter their path, puzzles to be implemented within this narrative in order for there to be a noticeable progression, textured models including enemies, weapons, and the environmental foliage (trees, plants, bushes). Furthermore, including enemies with artificial intelligence (AI) using the Unity3d navigation mesh, menu systems, shaders for abstract visuals to represent psychosis, an intuitive and responsive character controller, and finally spatial and binaural audio. The minimum awesome product (MAP) would endeavour a dynamic weather system, further enhanced movement and environment and a tutorial for controls.

The backbone of the report will discuss and outline; the projects background, objectives and deliverables, software, development methodology, the legal, social, ethical and professional issues, project management, architecture, the development process, functional and usability testing, end-project report, post-mortem and finally, conclusions.

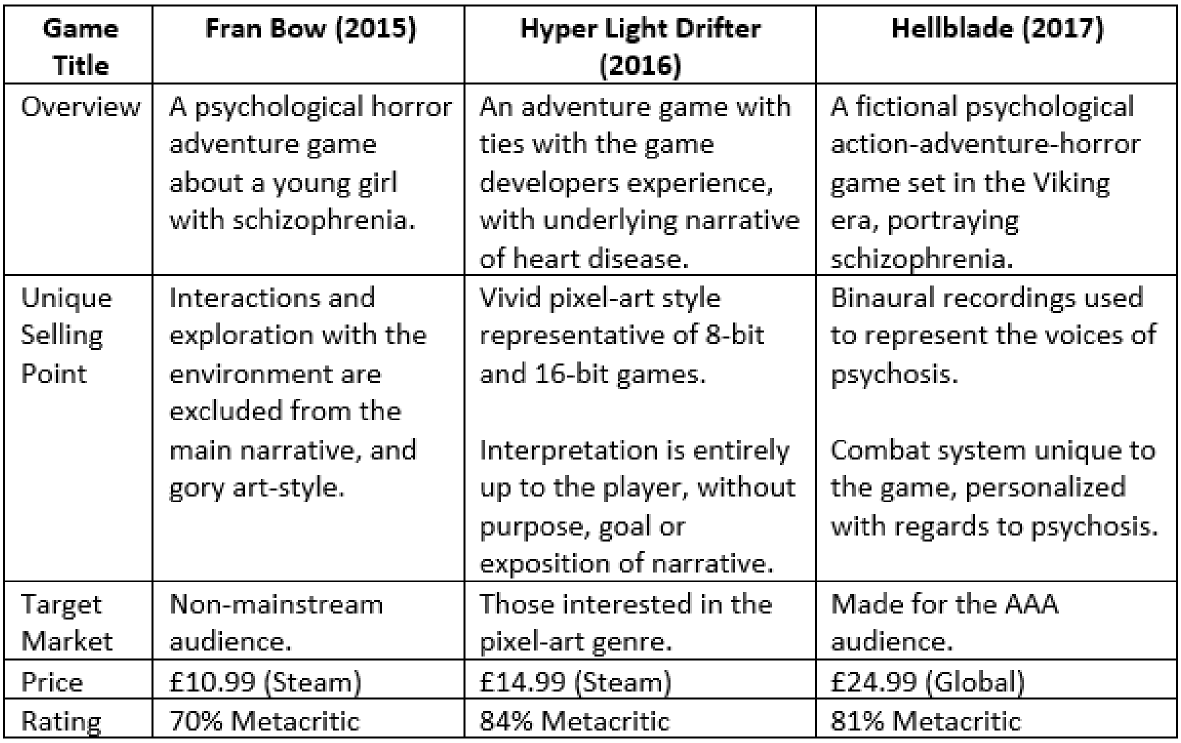
4. Background, Objectives and Deliverables

4.1. Background

As previously mentioned, the game mechanics intend to intertwine narrative gameplay with representative mechanics of mental illness, or an abstraction of an individual's mental state. The use of experimental Unity technologies inhibits immersion as part of the game visuals, with physically based rendering increasing details of textures on high poly meshes. High poly meshes are problematic due to high computational processing required, and therefore the environment needs to contain fewer foliage density as initially scoped. Binaural recordings and Foley sounds will also play a large part into the immersion of other senses, bringing the scene to life with spatial auditory cues and background ambience. Other objectives include the use of the ScrumBan methodology which will be measured through the efficient use of project management tools, and finally, an online presence will keep the target audience active and present throughout the release, which will be achieved through Twitter and a personal developer blog.

The main principle as reiterated as part of the introductory section is to educate players of the symptoms of psychosis and to aid therapeutic methods through entertainment-based games. This is justifiably the purpose of the project; however, a market analysis of games with attributes to mental illness or mental well-being will provide essential understanding towards how these symptoms have been implemented and represented, as well as potential competition for the game's release. Fran Bow (Fran Bow, 2015) [4], Hyper Light Drifter (Hyper Light Drifter, 2016) [5], and Hellblade (Hellblade, 2017) [6] are all game titles that represent mental illness in various mechanics and art styles, at different heights in the market (See Table 1).

Table 1 Competition that represent mental illness in various forms.



Game Title	Fran Bow (2015)	Hyper Light Drifter (2016)	Hellblade (2017)
Overview	A psychological horror adventure game about a young girl with schizophrenia.	An adventure game with ties with the game developers experience, with underlying narrative of heart disease.	A fictional psychological action-adventure-horror game set in the Viking era, portraying schizophrenia.
Unique Selling Point	Interactions and exploration with the environment are excluded from the main narrative, and gory art-style.	Vivid pixel-art style representative of 8-bit and 16-bit games. Interpretation is entirely up to the player, without purpose, goal or exposition of narrative.	Binaural recordings used to represent the voices of psychosis. Combat system unique to the game, personalized with regards to psychosis.
Target Market	Non-mainstream audience.	Those interested in the pixel-art genre.	Made for the AAA audience.
Price	£10.99 (Steam)	£14.99 (Steam)	£24.99 (Global)
Rating	70% Metacritic	84% Metacritic	81% Metacritic

All of these titles use violence or horror in some form, which could provoke a negative response from some of the target audience since it was directed with mental illness gameplay and mechanics. There is a gap in the market for a passive genre representative of mental illness, whereby the character must interact with the environment on player's own accord to benefit themselves, rather than mindlessly slashing away at enemies. The initial scope of the project included a combat system with enemy encounters; however, changes resulted in a portrayal of the opposite, with a calmer approach, yet consequential if the game goals are not met.

4.2. Objectives

The objectives were outlined in the Project Initiation Document (See Appendix Item B).

4.3. Project Outcomes

The project outcomes were outlined in the Project Initiation Document (See Appendix Item B).

4.4. Deliverables

The functional requirements and deliverables were outlined in the Project Initiation Document (See Appendix Item B).

5. Software

5.1. Engine

The criteria for the selected game engine was based on the current availability (affordability), learning curve, online resources (third-party), tools provided, stability, and licensing choices. Unity3d (Technologies, 2019) [7] and Unreal (Unrealengine.com, 2019) [8] are both supportive of three-dimensional mesh rendering and 'allow development under both macOS and Windows' (Dickson, et al., 2017) [9], which is most important due to the target audience being those using the Windows operating system, although there is a possibility for a broader approach, expanding onto macOS as well.

In terms of availability and affordability, Unreal is free; however, gross revenue above a threshold of $3,000 (per quarter), Epic (Epic Games, 2019) [10] takes 5% royalties (Pachoulakis and Pontikakis, n.d.) [11]. Even though the intentions are for the game to be non-commercialised, this should be taken into consideration in the comparative analysis between Unity3d. Unity uses three distinct licensing agreements, awarding the customer with a range of exclusive features. The 'Personal' license listed as free, is most famous for individuals that have not generated annual revenues of more than $100,000. On the other hand, the 'Plus' licensing agreement assumes that the individual has not raised more than $200,000. Finally, the Pro license allows no limitations on revenue or funding, although for a costly $125 per month (Assetstore.unity.com, 2019) [12]. Epic taking 5% of royalties could factor in significantly for high-end AAA game developers, but Unity and Unreal conclusively offer similar features for low budget developers using their free licensing agreements.

Unreal and Unity3d both provide community forums, support, video tutorials and documentation related to the engine. Unity arguably contains more excellent supportive content, with live sessions, workshops, and easily accessible training courses. Even though the extensive amount of support, both use different programming languages and therefore knowledge from one cannot be transferred fluently onto the next. Unreal uses the industry standard C++ with a visual language named Blueprints (great for beginners). Unity has slowly been incorporating its visual programming languages using Shader Graph and Visual Effects Graph, but primarily uses C# (not typical for game industry standards) and JavaScript (Dickson et al., 2017) [9]. The author's capabilities are most impactful on C#, and as to not hinder the development of the project, Unity would be the ideal preference with subjectively a minor learning curve.

Unity3d has an abundance of online resources on their asset store abiding by the EULA agreement dictated in their terms of service unless specified otherwise by the asset creator (Unity, 2019) [13]. This has also been discussed under Section 7.1 of this report. Unreal similarly states in its marketplace distribution agreement that assets may be used commercially once purchased (Unrealengine.com, 2019) [14]. The Unity3d store is likely to contain a larger quantity of assets on display due to its launch date in late 2010, on the other hand, the Unreal marketplace was released in 2014 (Unreal Engine, 2019) [15]. The utilization of third-party assets is vital in tight deadlines and takes the risk of over-scoping into account. The likelihood of more choice available on the Unity asset store is beneficial to the level design of the project.

Ultimately, Unity far outweighs the Unreal Engine in several areas. A downfall of this engine is in its ease of designing high-quality visuals. Unreal is far superior at first glance, although Unity has attempted in recent months to improve its render pipelines, releasing the LWRP (Lightweight Render Pipeline), alongside HDRP (High Definition Render Pipeline), discussed in the upcoming section.

5.2. Engine Pipeline

The High Definition Render Pipeline was chosen, as the characteristics of such fit the objective for developing an immersive experience through high-quality visuals (objective 1.a.). Indifference to the Lightweight Render Pipeline, HDRP provides physically based rendering, unified and coherent lighting, and features independent of the rendering path (Legarde, 2019) [16]. These features are targeted towards high-end computers and consoles, on the other hand, LWRP is pushed predominantly towards virtual reality and mobile games, with limited real-time lighting features (Cooper, 2019) [17].

5.3. Integrated Development Environment (IDE)

Visual Studio’s Community Edition (Visual Studio, 2019) [18] is the official IDE designed by Microsoft, packaged with Unity3d upon the first installation; including full compatibility, an extensive range of plugins, and debugging capabilities. Professional and Enterprise editions are a costly addition and come with few add-ons unnecessary for indie game development. MonoDevelop IDE (Monodevelop.com, 2019) [19] was also considered, although it has been deprecated since the beginning of 2018, with subpar auto formatting of C# code and fewer features due to its outdated version.

5.4. Repository Management

Regular commits of an on-going project to prevent data loss that allows the transferring of data across machines and time-efficient reverting of changes. GitHub (GitHub, 2019) [20] was identified as the ideal repository management platform with standard features such as commit, push, pull, merge and branch. However, its respectable industry standpoint sets it apart from other repository platforms, becoming the largest ‘social coding platform’ (Hu et al., 2016) [21] back in 2016. The report evidenced its relevancy within the industry due to its flexible workflow for large teams, nevertheless still making it a credible platform for a single individual. Furthermore, if influent with Git commands, GitHub Desktop (GitHub Desktop, 2019) [22] is a downloadable visual interface that makes the most prominent commands easily accessible to the user. Git Large File Storage (Git Large File Storage, 2019) [23] is an open source extension which assigns pointers onto files that were predefined by the ‘.gitattributes’ file. This made it possible to commit large image files (for example; lightmaps) to the repository instead of onto external drives, as GitHub has placed a strict limit on file sizes above 100 megabytes and a repository size of 1GB, extendable with a monthly fee (Help.github.com, 2019) [24].

5.5. Sound Design

Audacity (Audacity, 2019) [25] was chosen as the preferred free auditory editing software due to its relevance to the project’s audio objectives (objective 1.b). Stereophonic and monophonic channels made it suitable for binaural recordings (Maijala, 1997) [26] to be split between left and right outputs. Adjustable volume levels of channels also supported binaural hearing so the recordings would become easily definable between the left and right output channels. Additionally, background noise from the microphonic hardware was controlled with the filter’s default to the software. Finally, binaural beats were generated with Sine waves of different frequencies on separate channels; a neurological phenomenon that can activate various emotions depending on the frequency range, ranging from a sub-conscious unsettling feeling or therapeutic calmness (Lane et al., 1998) [27]. For background music and synths, Studio One 3 Prime (PreSonus, 2019) [28] was opted for due to its range of pre-set instruments. This version of the software is a free entry-level available to students and beginners.

5.6. Artistic Design

Adobe Photoshop CC 2019 (Adobe.com, 2019) [29] was used for user interface design and other graphic content that was not otherwise in the game before the implementation of third-party assets. Alternatively, GIMP (GIMP, 2019) [30] is a free, open-source graphics editor with similar features. Photoshop was the preferred editor as it offered a professional experience and a slick, easy-to-navigate interface.

6. Method of Approach

Three types of methodologies exist to secure a rigid method of approach for the success of a project, as suggested by (Bajali and Murugaiyan, 2012) [31], which details comparison between the Waterfall model, V-model and Agile model. Historically, the Waterfall model was formally introduced as early as 1970 (Winston and Royce, 1970) [32] and illustrated sequential steps with minimal iterations with the requirements. Each step must be thorough and complete before continuing onto the following tasks; analysis, design, development testing, implementation, and maintenance. This is beneficial for large government-funded projects or business plans with high probable risk towards health and safety (for example; power plants), where extensive documentation of phases is required, and a clear and concise time frame is given. Although, (Larman and Basili, 2003) [33] assess how the paper instead draws forward this approach for the simplest of projects, and that incremental steps would be preferable. Game development does not put forth the simplest of tasks, whereby requirements may change, unexpected events may hold activities back, and the functional testing of code would need to be tested vigorously throughout the development lifecycle.

Another methodology with similarities of Waterfall is the V-model (Bajali and Murugaiyan, 2012) [31], also known as the Validation and Verification model. This is the least flexible but steers towards Agile with developers and testers working in parallel, becoming involved with the modification of requirements. Once requirements are modified, then the redocumentation of testing needs completing, and these steps need to be verified before moving forwards which could cause a delay. The deadline was imperative, and therefore any model with delay or lesser flexibility in contrast to an Agile model would be rejected.

Agile development transfers the steps of the Waterfall model into an incremental approach that '… responds quickly to the changing requirements of the project.' (Bajali and Murugaiyan, 2012) [31]. The Project Initiation Document of this project devised a plan of action, but requirements were never compulsory, and 'Astray' evolved into a more purposeful application, differing from the initial scope. Scrum and Kanban are methods of utilizing an iterative approach. Scrum is often associated with project teams that work together and regularly interact through a retrospective meeting held by the 'Scrum Master' (Cervone, 2011) [34]. These meetings are held after weekly or bi-weekly sprints from items pulled from the backlog; a list of requirements transformed into short-term features and tasks to complete. This approach employs the client to take an active stance in the development cycle, to improve on everchanging ideas towards an eventual release of the final prototype version. On the other hand, Kanban is a mechanism that leads to the flow of the project, and the circulation of sprints. Tasks are evaluated based on their difficulty and assigned a numerical value representative of such; there is a defined limit to how many of these tasks can be in-development at any one time, corresponding to the numerical value that was given (Anderson and Reinertsen, 2010) [35].

The decision was made to adopt both of these into one methodology named ScrumBan, suited for an individual using agile. Requirements are analysed and broken apart into backlog items, which are then grouped with other relevant tasks. These groups are pulled together into weekly or bi-weekly sprints decided on the subjective difficulty, necessary research, and probability of failure described in risk assessment. Each card is given a number value relative to this as suggested by Kanban. Functional testing and re-evaluation of requirements are complete throughout sprints. These additions of the Kanban workflow visualize the ‘cost of delay’ (Reddy, Anderson and Benson, 2016) [36] and assure prioritization of work, suggesting that lousy resource management with Scrum may lead to an overwhelming amount of work; ScrumBan is a means to move the project forward.

7. Legal, Social, Ethical and Professional Issues

7.1. Legal

All third-party assets have been sourced from the Unity3d Asset Store and therefore abide by the Unity EULA agreement as written in the legal terms (Unity, 2019) [37]. This suggests that assets are under fair use and can be a part of a commercialised product if the author so chooses. Any clashes between the EULA agreement and third-party licenses, the third-party license shall prevail; therefore, the rights to the product may differ between assets. The assets used have been referenced within the report documentation (See Section 10). A discussion of legal factors regarding software and third-party tools and assets can be found in Section 5 of the report.

The Book of the Dead: Environment pack had particular licensing agreements, with some third-party institutions disclosing that their tools are for non-commercialised products or purely educational. Nevertheless, these tools were not used, and the rest of the assets were used under Unity's Asset Store EULA (See Appendix Item M.1). Post processing, Render Pipeline core, High Definition Render Pipeline, Pro Builder and Shader graph all use Unity's companion license (Unity, 2019) [38], which notes that the software are the property of Unity and shall not be modified and cannot be sold as a competitor product. Thirdly, sound files that were additionally sourced on top of the created initially assets were approved for fair-use under the CC0 1.0 Universal (CC0 1.0), also known as the Creative Commons 0 act, which proposed that the modification and use of the work are within the public domain (Creativecommons.org, 2019) [39].

The code repository and licensing of the released product is licensed under GNU General Public License v3.0 (See Appendix Item O) which states there must be referenced availability of source code. This was chosen over the permissive licensing agreement of Apache License 2.0 as the author would like to work fully credited, to increase an online developer presence for future industry endeavour.

In order to classify the age restriction of a game, the author must register as a PEGI signatory; however, for a non-commercialised indie product, a statement of such classification would suffice. PEGI (Pegi.info, 2019) [40] age rating of 12 would be appropriate for this type of game; the realism conveyed in the environment makes it unlikely to portray fantasy characters (ParentInfo, 2019) [41] and therefore any age rating below this would be unethical and have legal consequences. Arguably, the insanity of the player and loss of the game could be seen as the death of a human, even when not visualized or explained, which could fit it within the age bracket of 16 or more. There is no use of alcohol or illegal substances, or any form of nudity due to the first-person point of view. Conclusively due to the implied death of the player, PEGI 16 age rating was given.

7.2. Social

Both ethical and social issues intertwine. The project may receive some backlash if the mental illness or state depicted is not represented relatively or correctly in the perspective of the audience, that is why research has been conducted (See Appendix K.1 and K.2) so that the audience receives the delivered project in its utmost accurate form. Incorrect information conveyed is not ethical and has been considered throughout the project in terms of symptoms depicted through visual and auditory content.

7.3. Ethical

All research that involved human participants was subject to the University's Ethics policy. Individuals were asked to participate in research to improve the user experience, with the right to withdraw at any time and have any data removed. Additional consent was asked for, for any recording of data through a different medium such as a camera or microphones. All of the requirements associated with the Ethic's policy was to ensure the protection and anonymity of the participant.

To be targeted towards an audience as an entertainment-based game for therapy, regarding the mental state of an individual within a fictional world, and also targeted towards others interested in acquiring knowledge of the subject, it was reasonable and fair to release the game for free. The topic is sensitive to some and therefore pricing the game costly could mean enduring backlash from respective users by taking advantage of a sensitive topic for commercial gain.

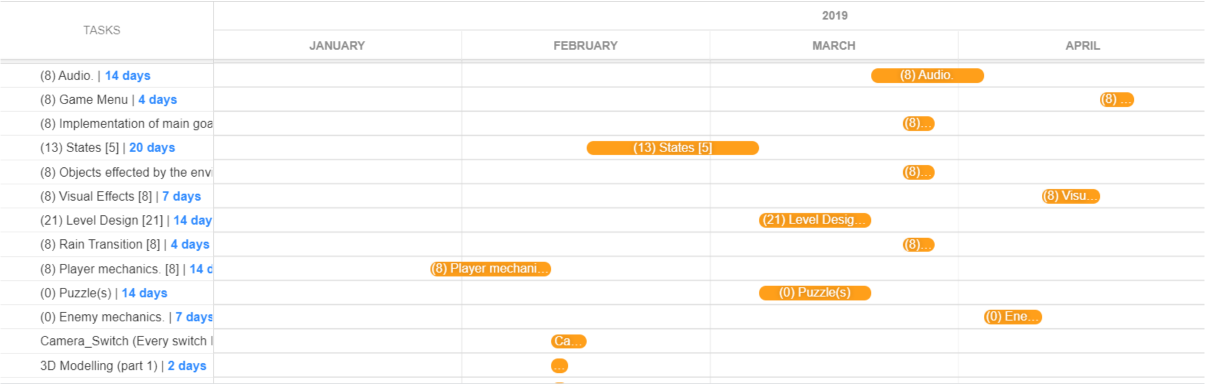
7.4. Professional

For professional standards, third-party tools were referenced throughout documentation even if not implied. These tools were part of the development process and information regarding this would be accessible to individuals interested in game development. Tutorials that were followed and code help from public forums were appropriately referenced at the points in which they were used within the code.

Regarding professional issues, the author accepts full responsibility for the work and is liable for any inaccuracies presented about mental health or mental state.

8. Project Management

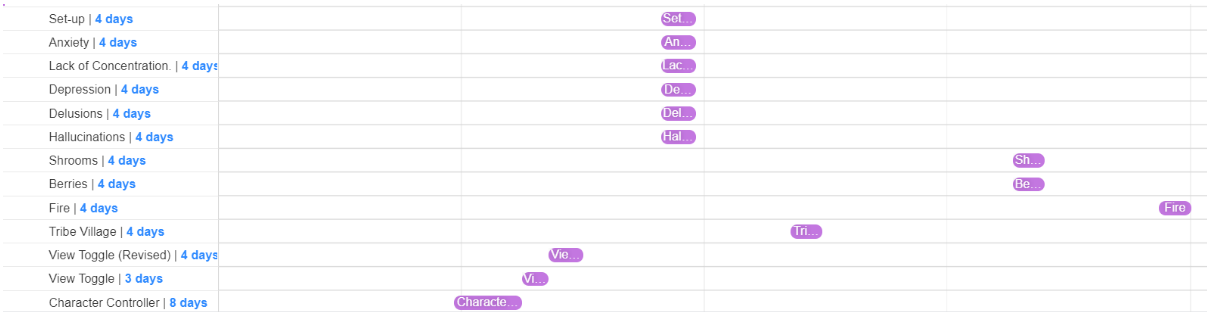
The project was managed through Trello (Trello.com, 2019) [42] as opposed to Kanbanflow (Kanbanflow.com, 2019) [43] because available plugins assisted with the ScrumBan workflow, and the board was easily customizable to adapt to a personalized approach to the methodology. Elegantt (elegantt.com, 2019) [44] was used to illustrate a roadmap for the project, as well as a time bracket for weekly or bi-weekly sprints. This allowed for timings of sprints to be reallocated in the case of untimely or unexpected occurrences concerning the risk management analysis. The full roadmap devised a plan of action with elongated periods after the expected release date, allowing for space for sprints to be extended, or further iterations, if not complete within the allocated timeslot. A high-level roadmap was created with stage releases (See Figure 1); each of these releases was given numerous sprints, which devised more specific time brackets within each release (See Figure 2).



TASKS

(8) Audio. | 14 days
(8) Game Menu | 4 days
(8) Implementation of main goa
(13) States [5] | 20 days
(8) Objects effected by the envi
(8) Visual Effects [8] | 7 days
(21) Level Design [21] | 14 day
(8) Rain Transition [8] | 4 days
(8) Player mechanics. [8] | 14 d
(0) Puzzle(s) | 14 days
(0) Enemy mechanics. | 7 days
Camera_Switch (Every switch l
3D Modelling (part 1) | 2 days

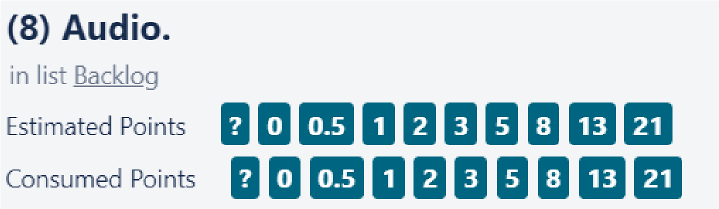
Figure 1 High level roadmap of releases from 28th January 2019 to 21st April 2019. This gives approximately 24 days for further iterations and exception plans.



Set-up | 4 days
Anxiety | 4 days
Lack of Concentration. | 4 days
Depression | 4 days
Delusions | 4 days
Hallucinations | 4 days
Shrooms | 4 days
Berries | 4 days
Fire | 4 days
Tribe Village | 4 days
View Toggle (Revised) | 4 days
View Toggle | 3 days
Character Controller | 8 days

Figure 2 Low level roadmap of sprints within the release deadlines. Each sprint contains features and tasks for reaching the specified goal.

Scrum for Trello (Scrumfortrello.com, 2019) [45] in correlation with Burndown for Trello (Burndown for Trello, 2019) [46] increased productivity and held true to the ScrumBan methodology. An arbitrary point system was allocated onto each release (See Figure 3), which was defined by the author to be the number of estimated hours due to difficulty and prior research required. Points could be consumed at intervals and update the Burndown chart respectively (See Section 10.7).



(8) Audio.

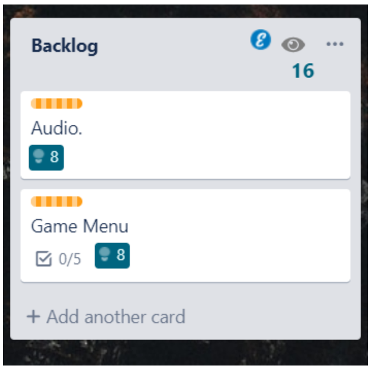
in list Backlog

Estimated Points ? 0 0.5 1 2 3 5 8 13 21

Consumed Points ? 0 0.5 1 2 3 5 8 13 21

Figure 3 Estimated and consumed points of the Audio release. Currently estimated time of completion to be 8 hours.

Functional requirements from the Project Initiation Document were added to the backlog (see Figure 4), which was added upon throughout the development lifecycle; this changed the project aims with some of the initial releases being cancelled. Time constraints and change in scope meant that modelling, enemy mechanics, combat systems and puzzles with relation to a narrative were all cancelled; however, these were replaced with new releases relevant to the new aims.



Backlog 8 16

Audio.
8

Game Menu
☑ 0/5 8

+ Add another card

Figure 4 Backlog of functional requirements grouped into releases.

Backlog items were pulled into the 'Planning' column of the Trello board; tasks were dictated by the functional requirements within the releases. After initial planning, these cards would be moved forth into the 'To-do list'. 'In development' set a limit of how many of these sprints could be transferred here. A maximum of six hours was assigned to this column, which was beneficial to the flow of the project as discussed as part of the method of approach. Even though testing would be considered throughout the sprint, finalized testing would be conducted before formally moving onto the next release. Bugs were also added into their column, which were free from release; however, it was important for these to be fixed as quickly as possible; if these bugs were to increment, then the complexity of producing a solution would increase exponentially. On top of this, research, planning, references, documentation, and milestones were stored within the Trello board to keep track of items that may be forgotten.

The project was additionally managed through weekly highlight reports with the supervisor, discussing feedback relevant to the sprint, objectives met, and how the approach could be improved within the following week, which has similarities towards retrospective meetings held by a team using the Scrum methodology. The deviation of some sprints (See Section 10.3) lead to exception reports, which prepared for the upcoming sprint in the reflection of the last, the author also returned to update risk management based on previous events. Although risks were appropriately defined after iterations, failure to factor in usability testing into the project management scheme resulted in the poor quality of user experience. This suggests that incomplete documentation due to the Agile methodology results in misinterpretations of time management.

9. Architecture

This section provides insight over Object Orientated Principles used to improve functionality, expandability, and coherence. An editor panel within the editor folder uses sliders to modify variables associated with the prodrome and acute phase of psychosis; anxiety, depression, lack of concentration, delusions, and hallucinations. This could be manipulated at runtime before the use of the variables in scripts, however now they serve to fulfil the purpose of the mental state mechanic. These variables control audio, post-processing and adjustments to player mechanics, and use changeable multipliers to determine the rate at which these will increase, as the player's sanity declines (See Section 10.4).

Some parts of the system are loosely coupled through encapsulation, a practice for maintaining code. The majority of scripts use private variables which can be accessed through public Get and Set methods. These methods add a step of the validity of the data being parsed and only allow specific variables to be modified. Private methods also act similarly; for example, the instantiation of the dialogue coroutine which stops the previously instantiated coroutine from continuing. Expandability of the system traverses' hand-in-hand with encapsulation, the key bindings script allows for easily modifiable bindings, and additions of such.

High cohesion is another principle adhered to; identifiable mechanics have been separated into their specific methods for readability and organization. As a result, code reuse is minimal. The following stages explore in-depth the architecture of the system with code examples.

10. Stages

Stages of development were split between sprints as accustomed to the ScrumBan methodology. After each weekly or bi-weekly sprint a log of the objectives complete was drafted into Highlight Reports (See Appendix Item C – J).

10.1. Sprint 1 – Ending 07/02/2019

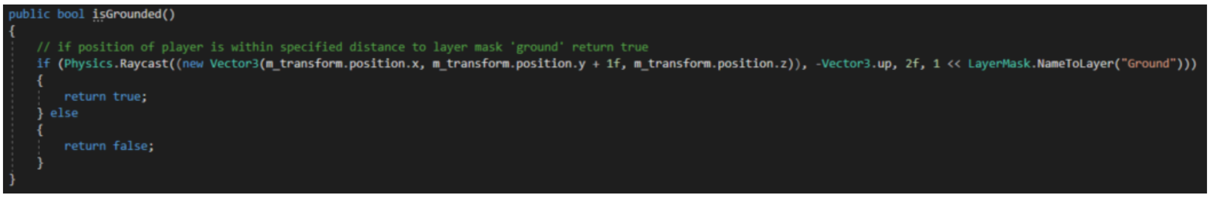
At the beginning of the sprint, an important decision was made to use the Unity3d version 2018.3.3f1, which included the High Definition Render Pipeline (HDRP) package. This package nullifies many resources on the Unity asset store due to its recent exposure to the public, and with sparse support due to its complexity. It is also very computationally intense as discovered throughout development; therefore, it needed to be balanced over the weighting between the high-quality visuals and performance. However, the HDRP package benefits an immersive gameplay experience, with improved visuals, volumetric lighting, and fog; therefore, this was chosen to be the root of development.

The narrative was also devised in early stages of development, discussed broadly within the Project Initiation Document (PID) (See Appendix Item B) and seen in the Narrative Plan (See Appendix Item K.3). Arguably a narrative can become a bottleneck for game development, whereby the over ambitions of the narrative can open difficulties in 'fun' game mechanics. However, it is to be considered the opposing point of view, (Fictionalgames.blogspot.com, 2019) [47] concludes that '... instead of having the gameplay describe the player's overall experience of the game, the narrative will provide this structure.'. With the narrative in mind, level design and game mechanics can be built around such, to create more transparent game flow, and an intertwined experience of narrative and gameplay.

Initial development began with a Rigidbody Controller for the player. It was important for the player to feel precisely in control of their characters movement, and enough visual and auditory feedback in return, yet portray realism for increased immersion. Two problems were indicated during this process.

One of which was that the Rigidbody controller felt like a rolling ball, not a human being. This decreased the immersion, but also the necessary feedback to the player. Secondly, while climbing onto objects such as blocks, the capsule collider would stick to the edge of the wall, resulting in unnecessary fall damage after dropping down. After numerous testing and attempts to fix this issue through the use of correct physics materials and dynamic collisions, the conclusions to this problem is that the physics engine for Unity was not mimicking the solution that the author had previously envisioned.

Fall damage was calculated through the amount of airtime a player took after leaving the ground, with a multiplier affecting the damage taken upon landing. The unnecessary fall damage was due to the inaccuracy of detecting if the player was grounded, firstly an attempt to raycast downwards towards an object within the layer mask of "Ground" would not correct return values (See Figure 5). With the Character Controller component, Unity provides access to an “isGrounded” Boolean, fixing and simplifying the code.



```
public bool isGrounded()
{
    // if position of player is within specified distance to layer mask 'ground' return true
    if (Physics.Raycast((new Vector3(m_transform.position.x, m_transform.position.y + 1f, m_transform.position.z)), -Vector3.up, 2f, 1 << LayerMask.NameToLayer("Ground")))
    {
        return true;
    } else
    {
        return false;
    }
}
```

Figure 5 Checks if the player is grounded with a Raycast.

The implementation of the Player Controller contained four logical steps (See Figure 6), firstly gathering the necessary inputs from the key bindings script, secondly setting the speed, which was dependent on the keys pressed, setting the direction via the vertical and horizontal axis, and finally checking if the crouch was toggled.

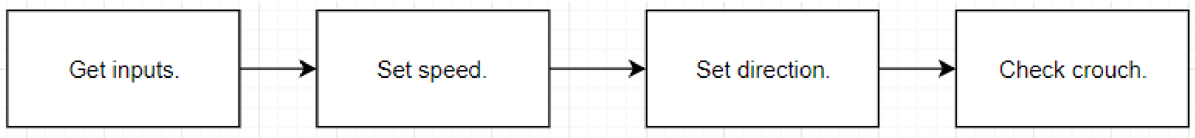
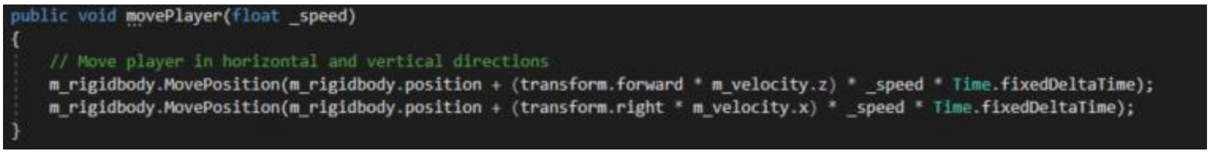


Figure 6 Flowchart representing Update method of the original movement script.

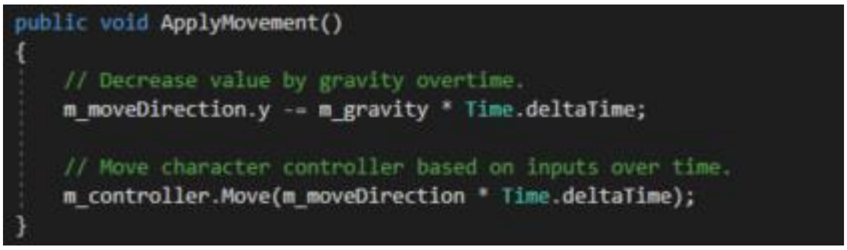
The addition of the Unity Character Controller component alongside the developed player movement and rotation scripts was used to combat this. This Character Controller introduced numerous player mechanics that gave the player freedom to explore different environments. Jumping, crouching, different movement speed types, and a dampened look rotation improved the quality of gameplay. These will feedback into later story-narrative gameplay and will become an available feature in the long-run.

Figures 7 and 8 demonstrate the differences between the new and improved script using the character controller, rather than the player’s Rigidbody. The Player Controller allowed the simple execution of the Move method.



```
public void movePlayer(float _speed)
{
    // Move player in horizontal and vertical directions
    m_rigidbody.MovePosition(m_rigidbody.position + (transform.forward * m_velocity.z) * _speed * Time.fixedDeltaTime);
    m_rigidbody.MovePosition(m_rigidbody.position + (transform.right * m_velocity.x) * _speed * Time.fixedDeltaTime);
}
```

Figure 7 Applying movement to the Rigidbody of the player.



```
public void ApplyMovement()
{
    // Decrease value by gravity overtime.
    m_moveDirection.y -= m_gravity * Time.deltaTime;

    // Move character controller based on inputs over time.
    m_controller.Move(m_moveDirection * Time.deltaTime);
}
```

Figure 8 Applying movement to the Character Controller of the player.

With a toggle crouch, the player was able to stand up and clip through objects above them, after standing up-right. This was fixed with the use of a raycast. Various types of raycast were tested; however, there were only two feasible solutions due to the radius of the character representing a cylindrical shape. CapsuleRaycast was firstly implemented, however with further functional testing; the player would still be able to toggle crouch under objects if they were to move quickly enough. Instead, a SphereCast was directed upward, and detect if an object were hit. If an object were detected, then the player would be unable to toggle the crouch and become blocked.

For prototyping, the installation of ProBuilds and ProGrids was utilized to create a miniature level for testing (See Figure 9). This would be used for white boxing levels later on, and then replaced with modelled and textured objects. ProGrids is intended for snapping objects correctly within grid-space, enticing a cleaner looking environment, while ProBuilds provides the various shapes and tools for objects.

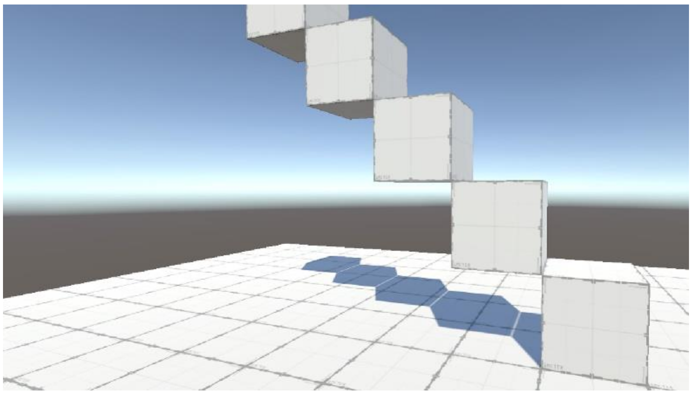


Figure 9 Prototype and testing level using ProBuilds and ProGrids.

Atop of these features, player management scripts were implemented. Health will become a unique selling point of the game, as for when the game progresses the sanity will decline; therefore, the player's maximum health will also decline to make the game progressively more difficult for the player. A steady climb of challenge, but easily adjustable. The player management scripts take control of health, the damage is taken from fall damage or other sources, and regeneration of health over time if the player has not been attacked.

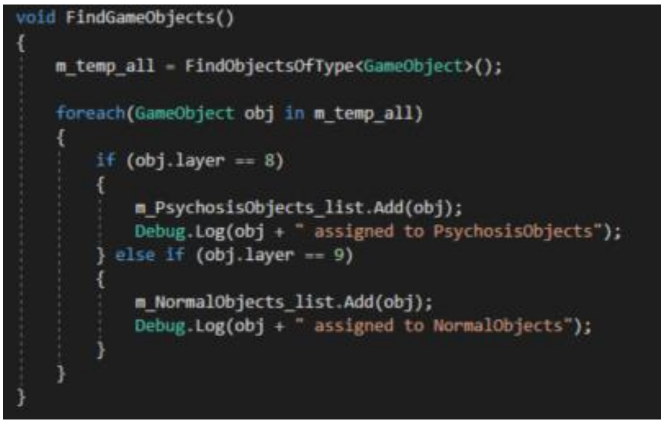
To provide inputs for every script with little to no code re-use, a key binding script was created with 'Get' methods, so that other scripts could reference the variables without accidental modification; therefore, adhering to encapsulation, one of the fundamental concepts of Object Orientated Programming (OOP) (Synder, A., 1986) [48]. If setters were to be added, the confined methods could ensure 'compatible changes,' changing the key bindings based on the player preferences. However, the setters were not implemented in 'Sprint 1' but considered for future implementation in later stages.

10.2. Sprint 2 – Ending 14/02/2019

After some careful reconsideration of the purpose of the game, the decision was made on a new approach to allow the player to take control of the anxiety, hallucinations, and so forth. This allowed the individual to switch to a less anxious world, or the same location with less fearful dangers. This was the result of a more thorough report, with a purpose to decrease anxiety, while upholding a compelling fictional narrative.

The first attempt at this mechanic was by adding two relatively identical scenes, with two player models based on each. This acted as a way to differentiate between the psychosis and the ability to control it. For example; hallucinations of enemies would in-turn become less of a threat, and pathways may be unveiled. This feature also changed the mindset towards the player's overwhelming list of components, separated between objects. Instead of the creation of an empty player-manager object held necessary scripts with relevance to the player, and therefore slight alterations to the scripts were required, such as a reference for the player object and camera. A vector was taken to create a seamless transition for the controllable character between two cameras, relative from an empty object in the scene that was identically placed in the second scene. Then the vector was applied to the secondary player prefab. This positioned the player precisely in the position of the other, but in its opposing area. Also, the rotational transform of the camera was set for the other camera (determined by the active camera).

Even though this was a seamless solution, having two levels within one scene would cost high amounts of processing power. A second solution was preferred, which was more efficient and still had layers of potential. Two cameras masked only appropriate layers ("Normal" and "Psychosis") on one player prefab. This meant that switching between the two cameras would result in only the selected layers being rendered. However, the colliders of these objects were still in-use as the resulting render mask would only affect the objects visually. A list was created for both "Psychosis" and "Normal" layered objects; the colliders of such were toggled off depending on the camera mask currently selected (See Figure 10).



```
void FindGameObjects()
{
    m_temp_all = FindObjectsOfType<GameObject>();

    foreach(GameObject obj in m_temp_all)
    {
        if (obj.layer == 8)
        {
            m_PsychosisObjects_list.Add(obj);
            Debug.Log(obj + " assigned to PsychosisObjects");
        } else if (obj.layer == 9)
        {
            m_NormalObjects_list.Add(obj);
            Debug.Log(obj + " assigned to NormalObjects");
        }
    }
}
```

Figure 10 Separating Game Objects within the scene depending on their assigned layer.

Once the Game Objects were allocated to their relevant listings, the Update method would lapse over three method calls. Firstly, the inputs for toggling the view were assigned. When the button is pressed, a Boolean will switch to the opposite value. The final method detects if a switch between cameras is required, with the colliders of relevant layer objects to be set inactive (See Figure 11).

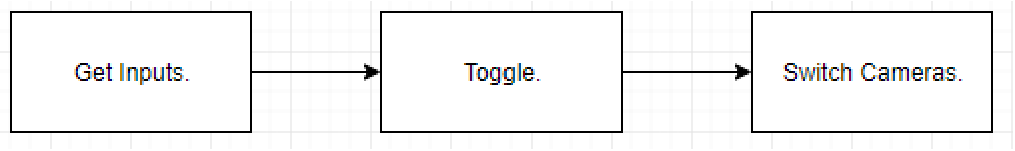


Figure 11 Flowchart representing Update method of switch camera script.

Following this, the decision was made to gather third-party assets due to time constraints on the initial scope. In order to immerse the player with dynamic visuals and detailed textures, The Book of the Dead: Environment asset pack (Assetstore.unity.com, 2019) [49] was imported. These assets were all from the Unity Asset store with licensing to be sold for commercial gain if required, with the exception to a few third-party tools that needed further permission or contain different licensing agreements if used; however, these tools were not used (See Appendix Item M). This asset pack contained a significant number of foliage and rock variations. Secondly imported was the Boneskin Settlement pack (Assetstore.unity.com, 2019) [50], which contained suitable buildings and huts associated with the Stone Age era, purposeful to the game context. Lastly, Mesh Effects (Assetstore.unity.com, 2019) [51] contained shader variations using the mesh of an object, with the original intention to be used in later developments for hallucinations the player may perceive.

10.3. Sprint 3 – Ending 21/02/2019

There were some troubles associated with sprint three of the project. Risk assessments and measures were taken in respect of these unexpected events.

At the beginning of the week's sprint there were priorities of a report due on the 18th February for module AINT308. This provoked uncertainty within the week's sprint, as workload would have to be prioritised accordingly. The attempt was made to combat the risk and catch up during the upcoming weeks by pushing for significant progress in level design.

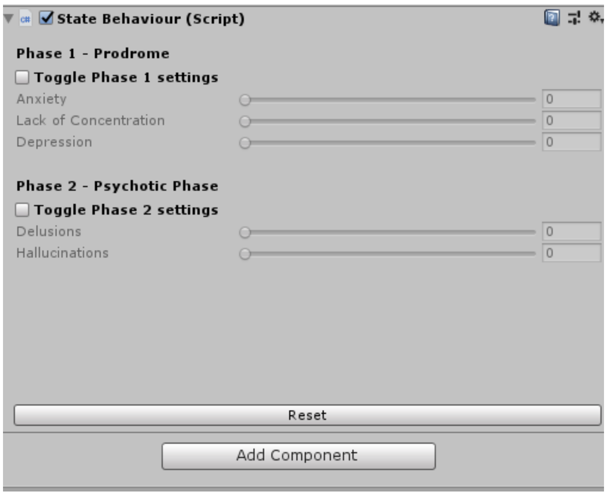
The second section of this risk assessment outlines measures evaluated from an Internet Service Provider outage. Virgin Media had an outage throughout the entirety of Plymouth for four days on the 14th February 2019, so progress on the level design was ground to a halt due to the inability to push changes or research effectively. The initiative was taken to talk to the provider; however, a due date for the fix was not provided. Work was not able to be traversed to the University computers due to a large number of files required to transfer over, and the router problems persisted. In light of this problem, the use of the University systems within a separate project folder, or branch sufficed.

For what could be accomplished within this week, psychosis research was conducted, with the condensed and relevant points of the findings, bullet-pointed in the miscellaneous deliverables (See Appendix Item K.1 – K.2). This research produced valuable information about symptoms, causes, and jargon associated with psychosis.

10.4. Sprint 4 – Ending 27/02/2019

It was essential to blur the differences between the states of psychosis easily. With the research conducted (See Appendix Item K.2), the addition of the ‘Prodrome phase,’ the ‘Acute Phase’ (Earlypsychosis.ca, 2019) [52], and their possible symptoms were implemented into an Editor Window of Unity. The first stage of psychosis, named the Prodrome, is a medical term that signifies the early diagnoses of symptoms, while the Acute phase is considered to be the stages in which the patient undergoes hallucinations and delusions.

The Unity Editor Window meant dynamic adjustments of these effects could be made while running the game as a way of visually presenting the solution to individuals of interest. Also, the initial intention of later usage in usability testing, for an understanding of when to indulge the player in each of these symptoms, and if the emotions reflect the expected (See Appendix Item N). However, the Editor Window became a needlessly convoluted approach to adjusting symptoms while in Play Mode, as the date would need to be serialized into scriptable objects, and then processed by scripts outside the Editor Window. Instead, a custom editor inspector panel was much more appropriate for this type of work (See Figures 12 and 13).



☑ State Behaviour (Script)

Phase 1 - Prodrome

□ Toggle Phase 1 settings

Anxiety 0

Lack of Concentration 0

Depression 0

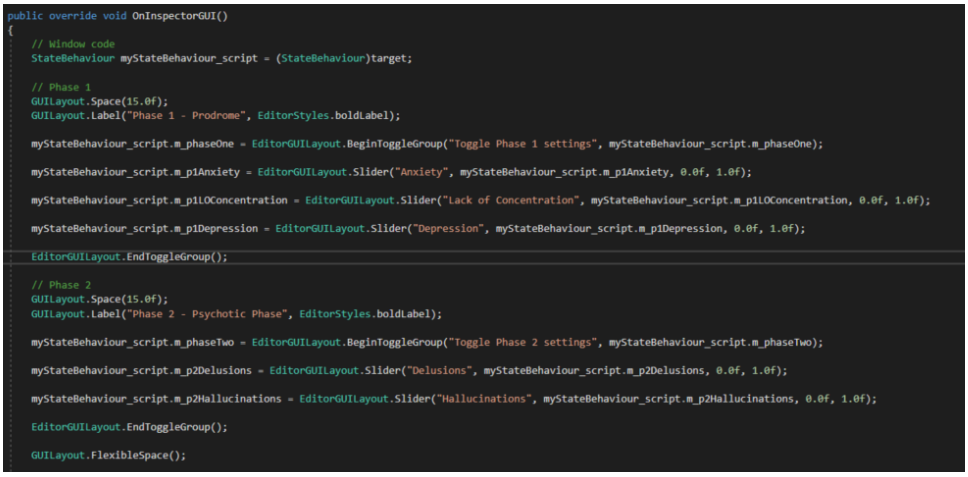
Phase 2 - Psychotic Phase

□ Toggle Phase 2 settings

Delusions 0

Hallucinations 0

Figure 12 The state behaviours of psychosis visualized on an Editor Panel. The Prodrome phase had associated symptoms of Anxiety, Lack of Concentration and Depression, while the Acute phase consisted of Delusions and Hallucinations. All of which were in direct correlation with the research conducted in Appendix Item K.1 and K.2.

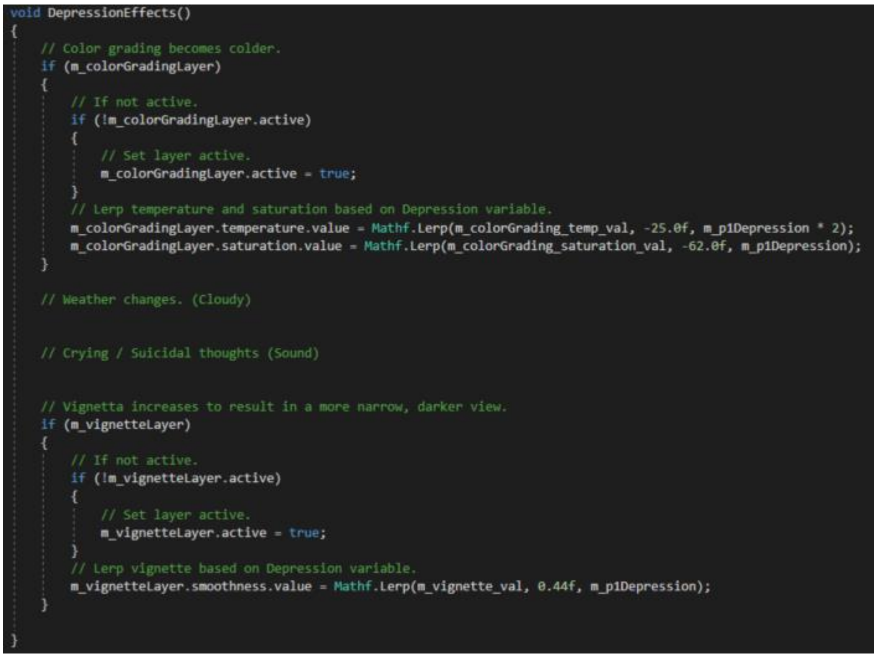


```
public override void OnInspectorGUI()
{
    // Window code
    StateBehaviour myStateBehaviour_script = (StateBehaviour)target;

    // Phase 1
    GUILayout.Space(15.0f);
    GUILayout.Label("Phase 1 - Prodrome", EditorStyles.boldLabel);

    myStateBehaviour_script.m_phaseOne = EditorGUILayout.BeginToggleGroup("Toggle Phase 1 settings", myStateBehaviour_script.m_phaseOne);

    myStateBehaviour_script.m_p1Anxiety = EditorGUILayout.Slider("Anxiety", myStateBehaviour_script.m_p1Anxiety, 0.0f, 1.0f);

    myStateBehaviour_script.m_p1LOConcentration = EditorGUILayout.Slider("Lack of Concentration", myStateBehaviour_script.m_p1LOConcentration, 0.0f, 1.0f);

    myStateBehaviour_script.m_p1Depression = EditorGUILayout.Slider("Depression", myStateBehaviour_script.m_p1Depression, 0.0f, 1.0f);

    EditorGUILayout.EndToggleGroup();

    // Phase 2
    GUILayout.Space(15.0f);
    GUILayout.Label("Phase 2 - Psychotic Phase", EditorStyles.boldLabel);

    myStateBehaviour_script.m_phaseTwo = EditorGUILayout.BeginToggleGroup("Toggle Phase 2 settings", myStateBehaviour_script.m_phaseTwo);

    myStateBehaviour_script.m_p2Delusions = EditorGUILayout.Slider("Delusions", myStateBehaviour_script.m_p2Delusions, 0.0f, 1.0f);

    myStateBehaviour_script.m_p2Hallucinations = EditorGUILayout.Slider("Hallucinations", myStateBehaviour_script.m_p2Hallucinations, 0.0f, 1.0f);

    EditorGUILayout.EndToggleGroup();

    GUILayout.FlexibleSpace();
```

Figure 13 State window code, each slider affects a variable in the state behaviour script.

As required to represent psychosis symptoms, these effects needed multiple features to interpolate between linearly. The following are the original thoughts to display these symptoms in the game, limited by visual and auditory immersion; post-processing effects adjustments, camera and player mechanic adjustments (rotation and movement), weather adjustments symbolic of mood, and sound design (delusions and thoughts).

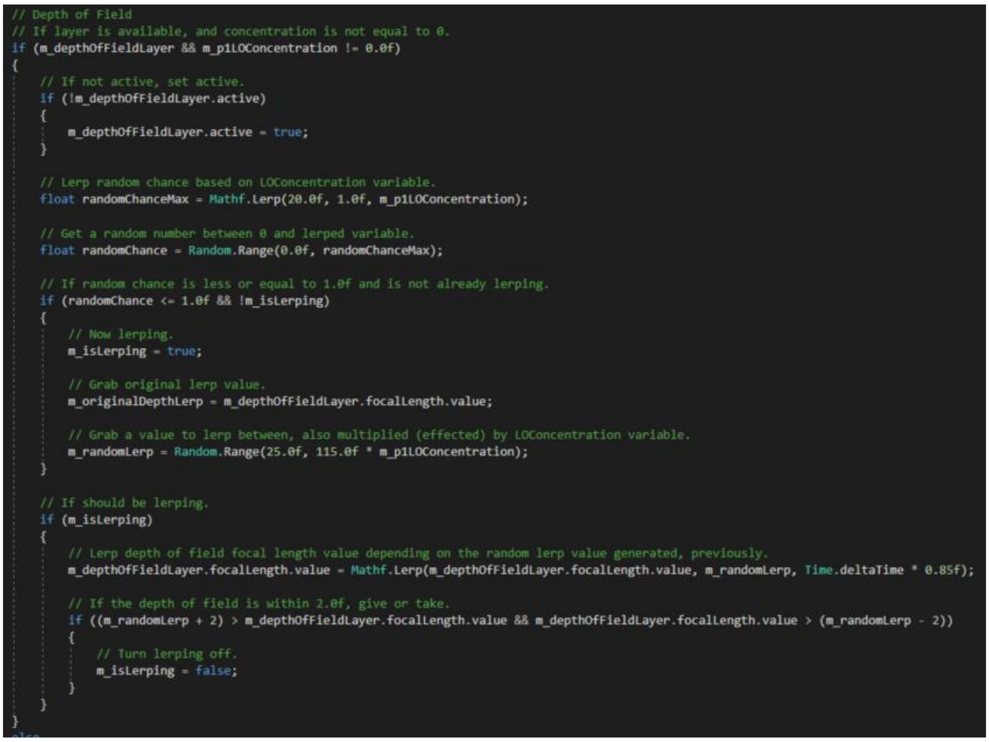
The post-processing effects and camera effects were adjusted and mapped with anxiety, depression and lack of concentration sliders at this stage. These were not set in stone, as user testing would provide valuable feedback onto whether the sensation and emotional feedback could be improved. For example; a grey and cold effect was implemented using temperature and saturation values of the colour grading post-processing effect for depression, with a more intense vignette (See Figure 14).



```
void DepressionEffects()
{
    // Color grading becomes colder.
    if (m_colorGradingLayer)
    {
        // If not active.
        if (!m_colorGradingLayer.active)
        {
            // Set layer active.
            m_colorGradingLayer.active = true;
        }
        // Lerp temperature and saturation based on Depression variable.
        m_colorGradingLayer.temperature.value = Mathf.Lerp(m_colorGrading_temp_val, -25.0f, m_p1Depression * 2);
        m_colorGradingLayer.saturation.value = Mathf.Lerp(m_colorGrading_saturation_val, -62.0f, m_p1Depression);
    }

    // Weather changes. (Cloudy)


    // Crying / Suicidal thoughts (Sound)


    // Vignetta increases to result in a more narrow, darker view.
    if (m_vignetteLayer)
    {
        // If not active.
        if (!m_vignetteLayer.active)
        {
            // Set layer active.
            m_vignetteLayer.active = true;
        }
        // Lerp vignette based on Depression variable.
        m_vignetteLayer.smoothness.value = Mathf.Lerp(m_vignette_val, 0.44f, m_p1Depression);
    }

}
```

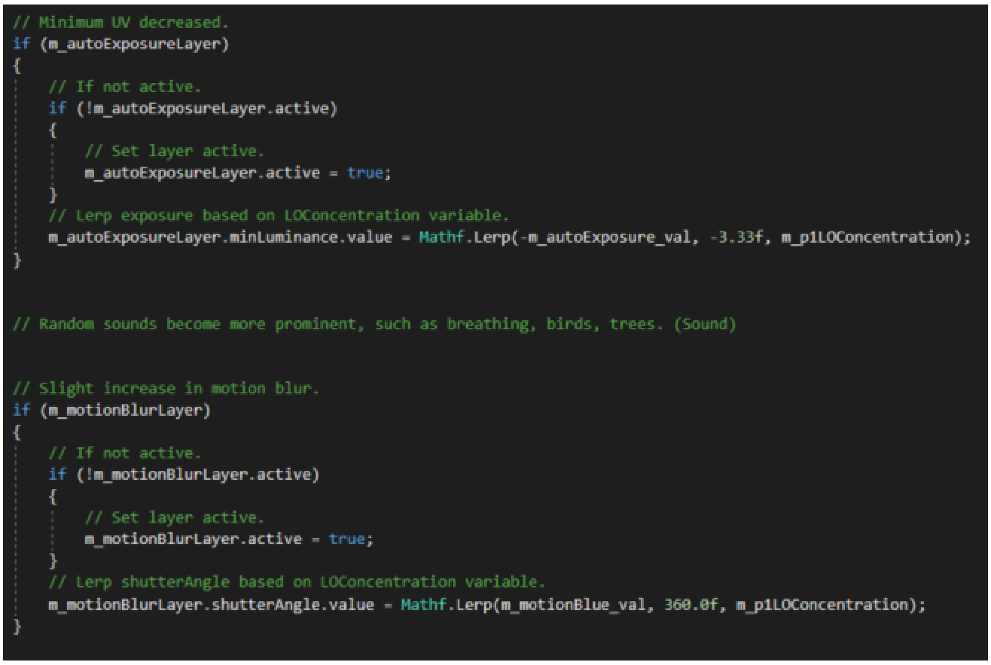
Figure 14 Depression effects taking values from the editor panel to interpolate post processing effects for saturation and temperature.

For lack of concentration, a brighter ultraviolet (UV) range, and the changing of the depth of field at random intervals depending on the slider variable (See Figure 15 and 16).



```
// Depth of Field
// If layer is available, and concentration is not equal to 0.
if (m_depthOfFieldLayer && m_p1LOConcentration != 0.0f)
{
    // If not active, set active.
    if (!m_depthOfFieldLayer.active)
    {
        m_depthOfFieldLayer.active = true;
    }

    // Lerp random chance based on LOConcentration variable.
    float randomChanceMax = Mathf.Lerp(20.0f, 1.0f, m_p1LOConcentration);

    // Get a random number between 0 and lerped variable.
    float randomChance = Random.Range(0.0f, randomChanceMax);

    // If random chance is less or equal to 1.0f and is not already lerping.
    if (randomChance <= 1.0f && !m_isLerping)
    {
        // Now lerping.
        m_isLerping = true;

        // Grab original lerp value.
        m_originalDepthLerp = m_depthOfFieldLayer.focalLength.value;

        // Grab a value to lerp between, also multiplied (effected) by LOConcentration variable.
        m_randomLerp = Random.Range(25.0f, 115.0f * m_p1LOConcentration);
    }

    // If should be lerping.
    if (m_isLerping)
    {
        // Lerp depth of field focal length value depending on the random lerp value generated, previously.
        m_depthOfFieldLayer.focalLength.value = Mathf.Lerp(m_depthOfFieldLayer.focalLength.value, m_randomLerp, Time.deltaTime * 0.85f);

        // If the depth of field is within 2.0f, give or take.
        if ((m_randomLerp + 2) > m_depthOfFieldLayer.focalLength.value && m_depthOfFieldLayer.focalLength.value > (m_randomLerp - 2))
        {
            // Turn lerping off.
            m_isLerping = false;
        }
    }
}
```

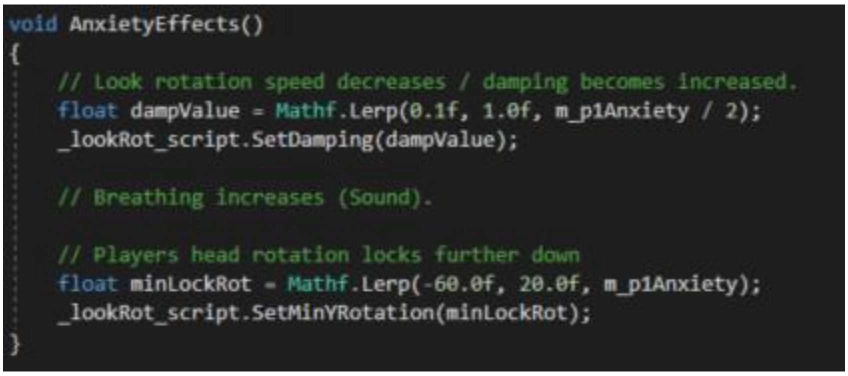
Figure 15 Lack of concentration effects as interpolated by the editor panel, changing the depth of field of the camera.



```
// Minimum UV decreased.
if (m_autoExposureLayer)
{
    // If not active.
    if (!m_autoExposureLayer.active)
    {
        // Set layer active.
        m_autoExposureLayer.active = true;
    }
    // Lerp exposure based on LOConcentration variable.
    m_autoExposureLayer.minLuminance.value = Mathf.Lerp(-m_autoExposure_val, -3.33f, m_p1LOConcentration);
}


// Random sounds become more prominent, such as breathing, birds, trees. (Sound)


// Slight increase in motion blur.
if (m_motionBlurLayer)
{
    // If not active.
    if (!m_motionBlurLayer.active)
    {
        // Set layer active.
        m_motionBlurLayer.active = true;
    }
    // Lerp shutterAngle based on LOConcentration variable.
    m_motionBlurLayer.shutterAngle.value = Mathf.Lerp(m_motionBlue_val, 360.0f, m_p1LOConcentration);
}
```

Figure 16 Additional lack of concentration effects, changing the auto exposure and motion blur of the camera.

Anxiety clamped the head in a downwards position, presenting the feeling of being locked in a situation, as well as slower head movements (See Figure 17). The sound would have drastically improved these effects; however, these were implemented at a later date, with a significant focus to visuals firstly. These effects are set back to their default states once toggled off.



```
void AnxietyEffects()
{
    // Look rotation speed decreases / damping becomes increased.
    float dampValue = Mathf.Lerp(0.1f, 1.0f, m_p1Anxiety / 2);
    _lookRot_script.SetDamping(dampValue);

    // Breathing increases (Sound).

    // Players head rotation locks further down
    float minLockRot = Mathf.Lerp(-60.0f, 20.0f, m_p1Anxiety);
    _lookRot_script.SetMinYRotation(minLockRot);
}
```

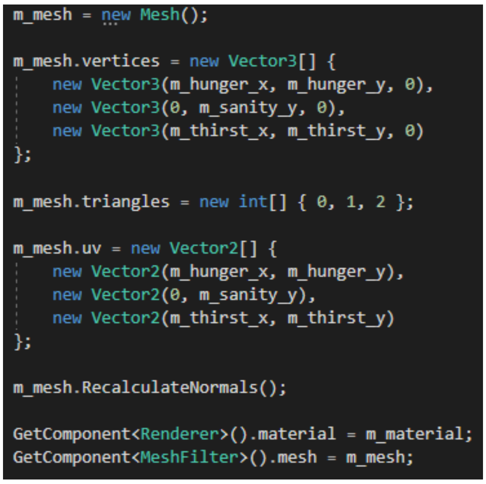
Figure 17 Anxiety effects taking values from the editor panel to interpolate the damping and clamp values of the look rotation script.

Additionally, other scene adjustments were implemented, including the High Definition Render Pipeline (HDRP) procedural skybox and scene settings. This skybox setting was used instead of the High Dynamic Range Imaging (HDRI) skybox as it could be used to transition from day to night without apparent rotation of the image. The HDRP diffuse profile was also set to the Book of the Dead: Environment pack profile as this pertained an immersive diffuse atmosphere. Finally, the field of view for both cameras was linearly interpolated as the player started to sprint, which gave the user valuable visual feedback.

10.5. Sprint 5 – Ending 07/03/2019

This was sprint was to create a way to visualize variables that the player would have to interpret to uphold their sanity. Three variables; sanity, hunger and thirst of the player would invoke the interaction with the environment around them, sustaining these variables to high upkeep, as to not induce any psychosis or changes in the state of mind. Thirst and hunger acted upon sanity, depletion of such decreased the player's sanity at an increased rate.

Two approaches were considered for visualizing these three variables into a radar graph; the utilization of the built-in user interface components, or mesh renderers. The Unity user interface components were not easily modifiable for radar graphs, and so the latter was chosen. Vertices, triangles and UV's and normal were required to be calculated (See Figure 18) to render a triangle mesh. After the creation of the new mesh object, this was applied to an objects mesh filter to be displayed.



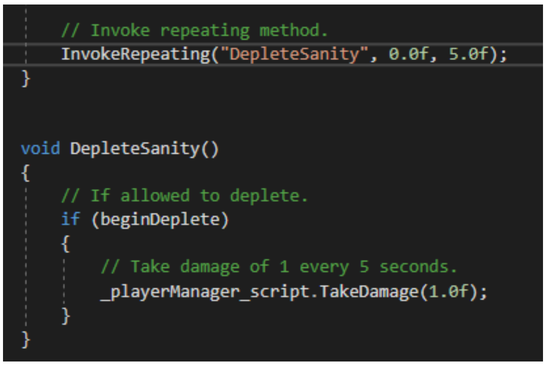
```
m_mesh = new Mesh();

m_mesh.vertices = new Vector3[] {
    new Vector3(m_hunger_x, m_hunger_y, 0),
    new Vector3(0, m_sanity_y, 0),
    new Vector3(m_thirst_x, m_thirst_y, 0)
};

m_mesh.triangles = new int[] { 0, 1, 2 };

m_mesh.uv = new Vector2[] {
    new Vector2(m_hunger_x, m_hunger_y),
    new Vector2(0, m_sanity_y),
    new Vector2(m_thirst_x, m_thirst_y)
};

m_mesh.RecalculateNormals();

GetComponent<Renderer>().material = m_material;
GetComponent<MeshFilter>().mesh = m_mesh;
```

Figure 18 The creation of a custom mesh in the radar graph script.

The invocation of another method calculated new X and Y positions for the triangle's vertices and proceeded to render the triangle iteratively. These new X and Y coordinates were associated with the hunger, sanity, and thirst variables.

The radar graph uses a customize shader, which has a 'Zwriteoff' and render queued overlay so that it draws in front of all other objects on the screen. Atop of this, the radar graph would be displayed on a canvas object; however, due to adverse effects the shader was used as a contingency in-case the canvas does not render the triangle correctly.

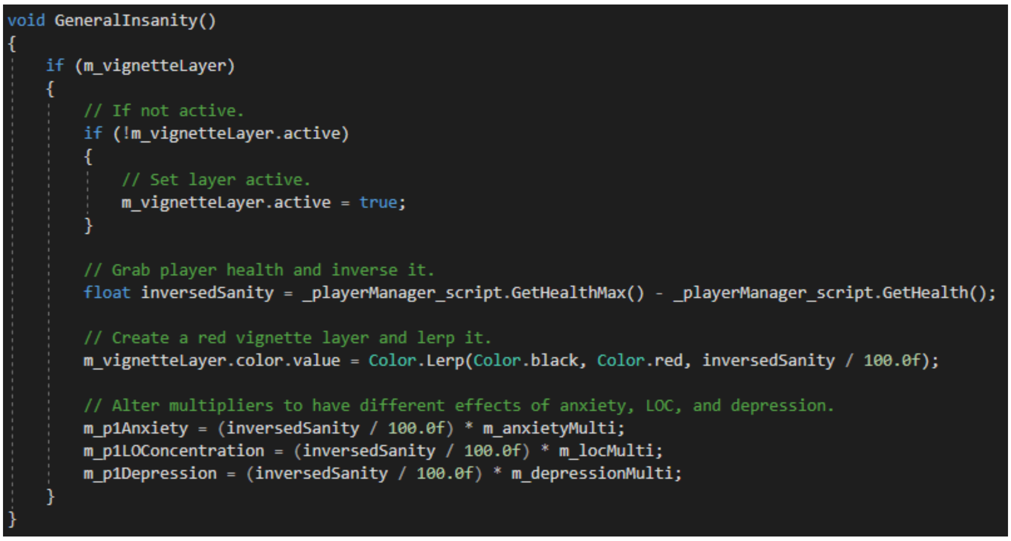
For the radar to take full advantage of its script, an initial decrease in the player's sanity would be required to prompt the player to find resources (food, water, shelter). The sanity depletion script invokes a repeating method, referencing the player-manager script and calls the method take damage. One tick of damage is taken every five seconds for a steady decline (See Figure 19).



```
    // Invoke repeating method.
    InvokeRepeating("DepleteSanity", 0.0f, 5.0f);
}


void DepleteSanity()
{
    // If allowed to deplete.
    if (beginDeplete)
    {
        // Take damage of 1 every 5 seconds.
        _playerManager_script.TakeDamage(1.0f);
    }
}
```

Figure 19 Depletion of sanity overtime, invoked to repeat every five seconds.

The general depletion of sanity now needed to affect the symptoms of psychosis. Within the state behaviour script, the health was inverted so that the values of anxiety, depression and lack of concentration could interpolate correctly. A multiplier was also in effect, which provided the flexibility of the system to decide the rate at which the variable increases or decreases (See Figure 20).



```
void GeneralInsanity()
{
    if (m_vignetteLayer)
    {
        // If not active.
        if (!m_vignetteLayer.active)
        {
            // Set layer active.
            m_vignetteLayer.active = true;
        }

        // Grab player health and inverse it.
        float inversedSanity = _playerManager_script.GetHealthMax() - _playerManager_script.GetHealth();

        // Create a red vignette layer and lerp it.
        m_vignetteLayer.color.value = Color.Lerp(Color.black, Color.red, inversedSanity / 100.0f);

        // Alter multipliers to have different effects of anxiety, LOC, and depression.
        m_p1Anxiety = (inversedSanity / 100.0f) * m_anxietyMulti;
        m_p1LOConcentration = (inversedSanity / 100.0f) * m_locMulti;
        m_p1Depression = (inversedSanity / 100.0f) * m_depressionMulti;
    }
}
```

Figure 20 Changing anxiety, depression and lack of concentration symptoms depending on the sanity of the player within the state behaviour script.

10.6. Sprint 6 – Ending 14/03/2019

The first village scene was appropriately set-up (See Figure 21), with huts from the Boneskin Settlement pack and foliage from the Book of the Dead: Environment pack. With performance in mind, the decision was made to use backdrops (flat planes with textures) for trees at a greater distance, as the details of such would be unidentifiable. Moreover, the performance was improved with the Book of the Dead: Environment pack that contained Level of Detail (LOD) meshes which calculates the camera's distance and reduces the number of triangles if further away (Technologies, 2019) [53].

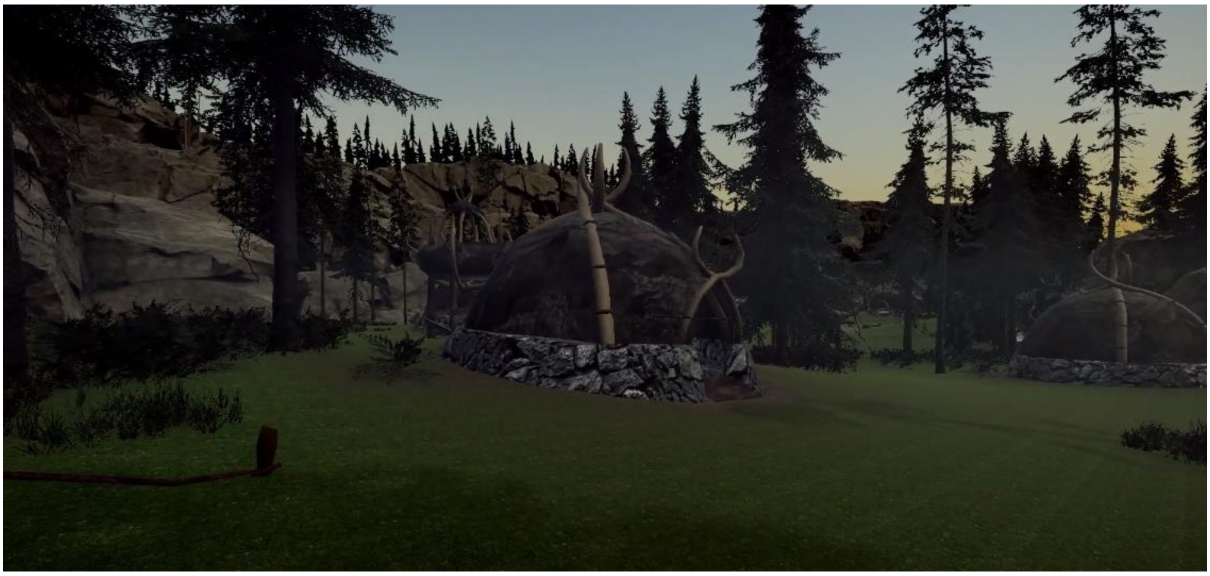
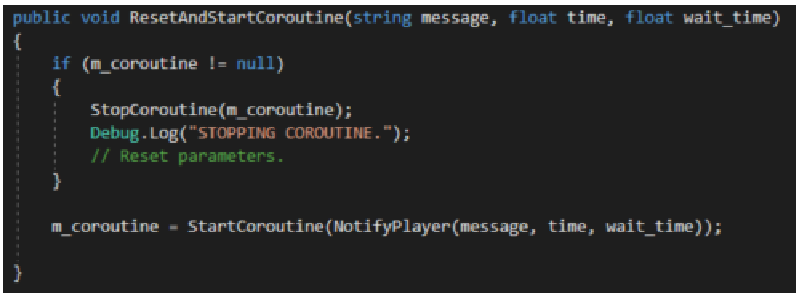


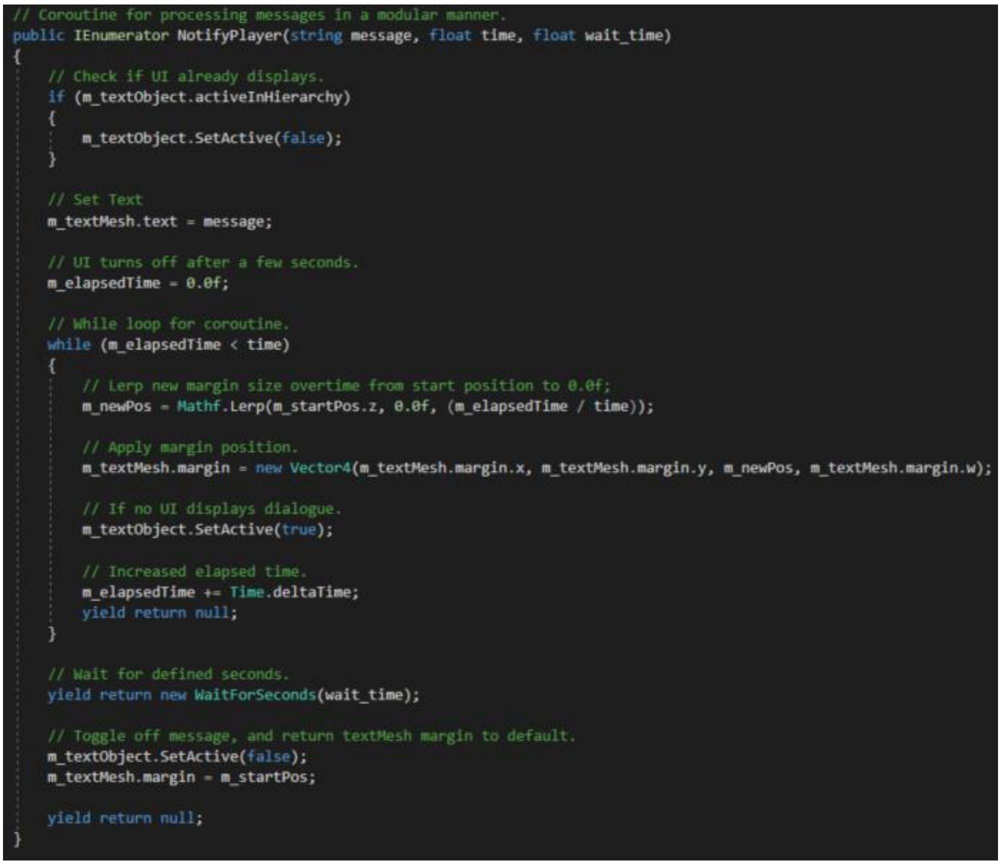
Figure 21 Huts and foliage placed for the village scene.

To prevent the player from exiting the village without the completion of interactions and dialogue prompts with various Non-playable Characters (NPC), a check on exit was implemented which gave appropriate player user interface feedback if tasks were still pending. A separate coroutine could be called from any script to sustain an abstract and modular approach, overwriting any previously on-going routines. The method as seen in Figures 22 and 23, demonstrates how this coroutine behaves, taking in a string, an elapsed time for the stylized slide transition effect, and finally the time it takes to be set inactive.



```
public void ResetAndStartCoroutine(string message, float time, float wait_time)
{
    if (m_coroutine != null)
    {
        StopCoroutine(m_coroutine);
        Debug.Log("STOPPING COROUTINE.");
        // Reset parameters.
    }

    m_coroutine = StartCoroutine(NotifyPlayer(message, time, wait_time));

}
```

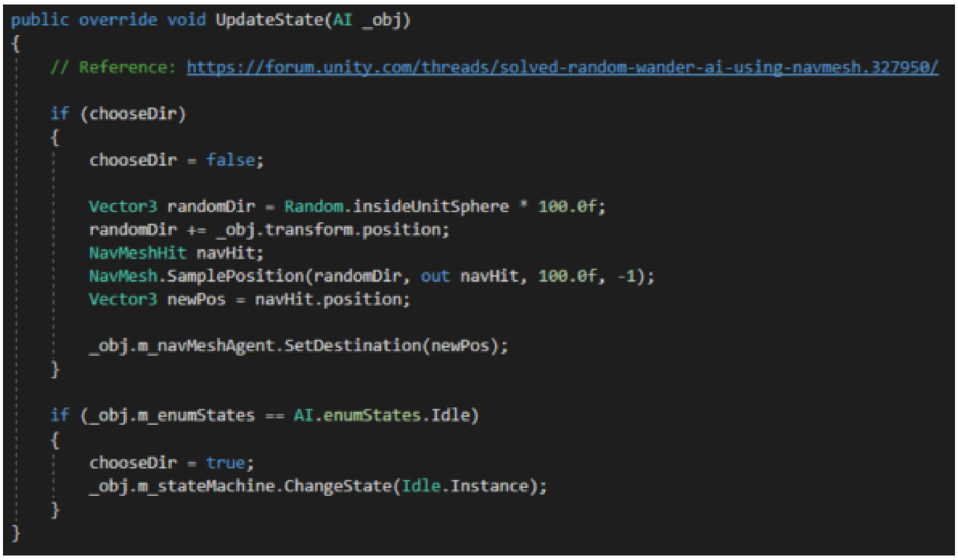
Figure 22 Method to reset and start a coroutine with a new message.



```
// Coroutine for processing messages in a modular manner.
public IEnumerator NotifyPlayer(string message, float time, float wait_time)
{
    // Check if UI already displays.
    if (m_textObject.activeInHierarchy)
    {
        m_textObject.SetActive(false);
    }

    // Set Text
    m_textMesh.text = message;

    // UI turns off after a few seconds.
    m_elapsedTime = 0.0f;

    // While loop for coroutine.
    while (m_elapsedTime < time)
    {
        // Lerp new margin size overtime from start position to 0.0f;
        m_newPos = Mathf.Lerp(m_startPos.z, 0.0f, (m_elapsedTime / time));

        // Apply margin position.
        m_textMesh.margin = new Vector4(m_textMesh.margin.x, m_textMesh.margin.y, m_newPos, m_textMesh.margin.w);

        // If no UI displays dialogue.
        m_textObject.SetActive(true);

        // Increased elapsed time.
        m_elapsedTime += Time.deltaTime;
        yield return null;
    }

    // Wait for defined seconds.
    yield return new WaitForSeconds(wait_time);

    // Toggle off message, and return textMesh margin to default.
    m_textObject.SetActive(false);
    m_textMesh.margin = m_startPos;

    yield return null;
}
```

Figure 23 Coroutine to display any message to the player, can be used to display dialogue between Non-playable Characters, commentary or exposition, and tasks alike.

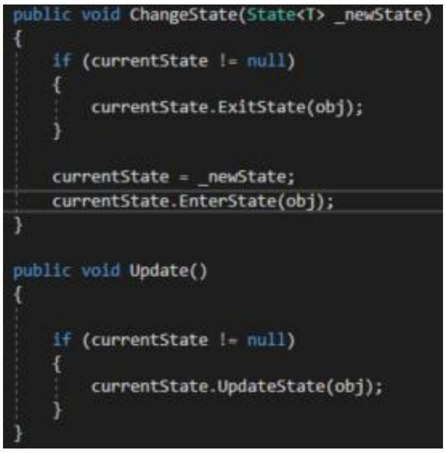
Due to the planned development of Non-playable Characters for the player to interact with, a Finite State Machine (FSM) was essential (See Appendix Item K.4). This implementation made use of three constants; idle, wander and converse. Each of these states was instantiated with the Singleton pattern only to allow one of these to be active at a given time. Each state provided an enter and exit state, but most importantly an update state method (called with Unity's Monobehaviour Update method) which defines what each state processes; for example, the Wander state chooses a direction within a sphere and uses that position to move towards. The enter and exit state are useful for animations to pursuit a smooth transition between states (See Figure 24).



```
public override void UpdateState(AI _obj)
{
    // Reference: https://forum.unity.com/threads/solved-random-wander-ai-using-navmesh.327950/

    if (chooseDir)
    {
        chooseDir = false;

        Vector3 randomDir = Random.insideUnitSphere * 100.0f;
        randomDir += _obj.transform.position;
        NavMeshHit navHit;
        NavMesh.SamplePosition(randomDir, out navHit, 100.0f, -1);
        Vector3 newPos = navHit.position;

        _obj.m_navMeshAgent.SetDestination(newPos);
    }

    if (_obj.m_enumStates == AI.enumStates.Idle)
    {
        chooseDir = true;
        _obj.m_stateMachine.ChangeState(Idle.Instance);
    }
}
```

Figure 24 Enter, exit and update of the wander state, part of the FSM.

To change the state at random intervals between five and ten, the AI script calculated the time elapsed and changed the constant enum. If the enum is changed, the state of the Non-playable Characters transitions to the respective state, firstly by exiting the current state via the ExitState method, entering the new state via the EnterState method, and finally the UpdateState method of the state would be processed (See Figure 25). With the baked navigation mesh of the environment and navigation agents attached to Non-playable Characters, the transitions between states work independently on each, which sustained an unpredictable movement for an expandable approach using appropriate Object Orientated Patterns.



```
public void ChangeState(State<T> _newState)
{
    if (currentState != null)
    {
        currentState.ExitState(obj);
    }

    currentState = _newState;
    currentState.EnterState(obj);
}

public void Update()
{
    if (currentState != null)
    {
        currentState.UpdateState(obj);
    }
}
```

Figure 25 State Machine script, changing state and updating state of an object.

10.7. Sprint 7 – Ending 21/03/2019

This sprint was dedicated towards the creation of a grittier atmosphere for the village, including the implementation of weather effects through various means, and iterations over the village scene to make it look grittier as discussed during the Highlight Report meeting (See Appendix Item I). The volumetric fog was used instead of exponential fog as this sustained a better atmosphere with regards to a grittier scene and took full advantage of the High Definition Render Pipeline features which introduced light rays (See Figure 26).

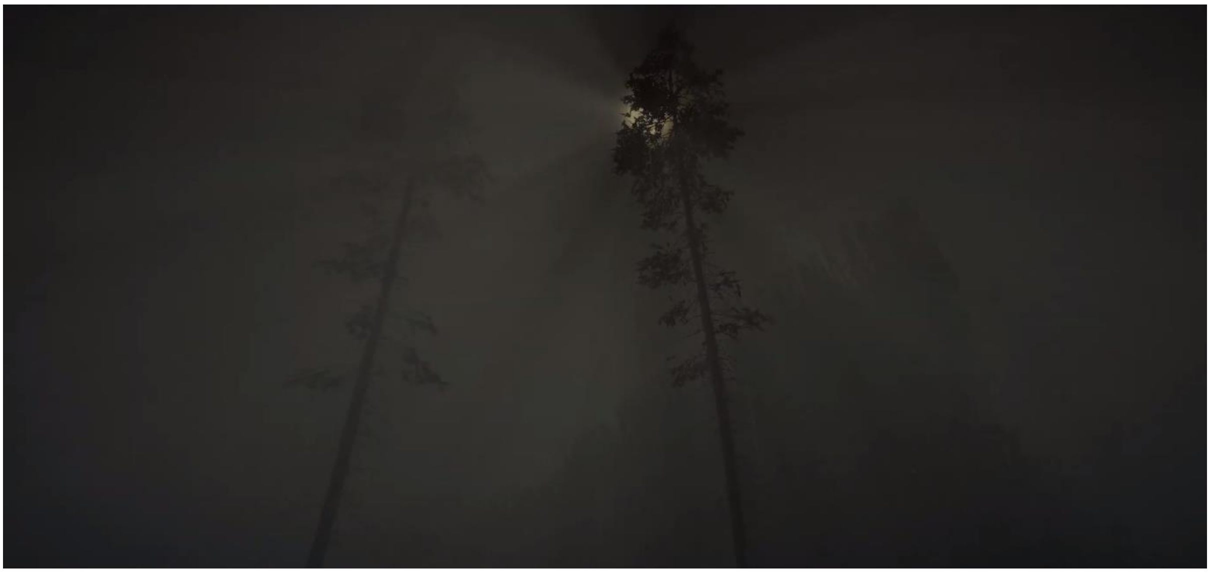


Figure 26 Demonstrates light rays responding to the volumetric fog.

At this stage, a dramatic change in scope and game objective was devised. The analysis of the burndown chart data made it clear that the previously envisioned goal was not going to be met. On the 21st March 2019, there were approximately fifty hours more work to complete, which did not consider usability and functional testing. In reflection of this, the gameplay would only be located in the village, where the weather affected the environment, and the sanity of the player was going to be affected by environment interactions. There was still an underlying narrative to the fictional world; however, this would only be useful for level design, dialogue and exposition, the story would therefore not be driving the gameplay (See Figure 27).

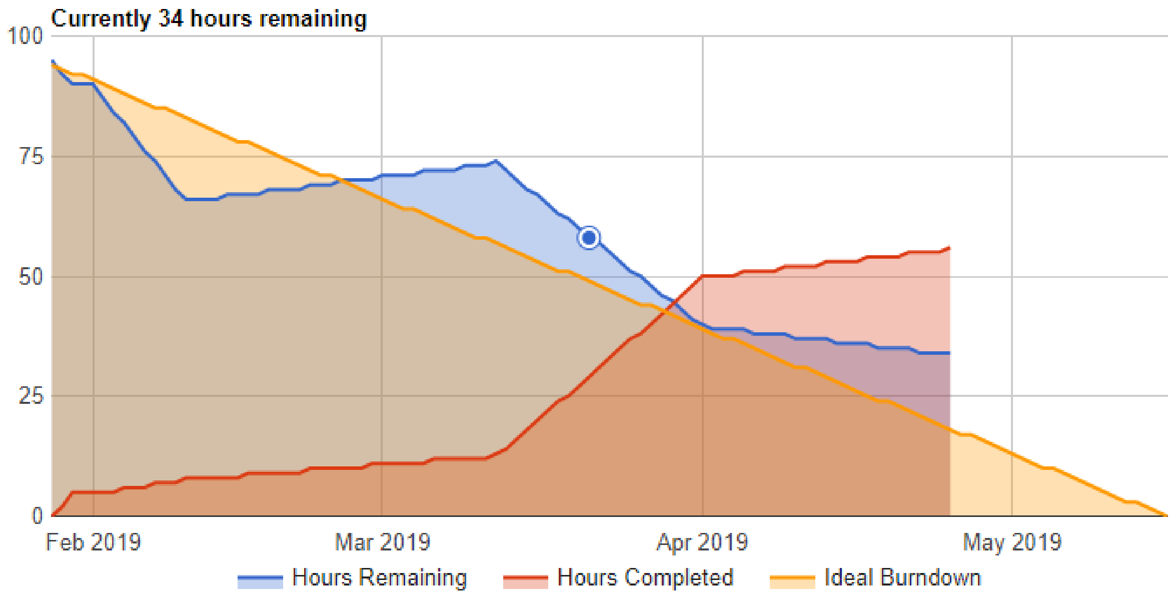


Figure 27 Burndown chart displaying the approximated number of hours left. The vertical (Y) axis represents the hours, while the horizontal axis (X) represents the months towards completion. The 21st March 2019 is displayed as a dot on the graph.

With the new scope of the project in mind, research into weather systems was the next logical step. The author tested with Sam Lord's open source weather system (GitHub, 2019) [54]; however, it was not supported by the High Definition Render Pipeline. Instead, Unity's particle pack (Assetstore.unity.com, 2019) [55] was used for rain particle effects. It seems that with further testing only a few particle effect systems work with Unity's experimental build for the HDRP. The use of Visual Effects Graph proved necessary at later stages of development, which supported GPU particle rendering (GPU simulated particles) for improved performance. A script made it so a reasonably small diameter of rain would follow the player's position to increase the performance of regular particle effects using the CPU, and therefore rain particle effects would not be processed all over the environment. To further enhance the performance the far clipping plane of the camera was decreased so that fewer triangles were rendered at any given moment.

10.8. Sprint 8 – Ending 28/03/2019

To begin this sprint; additions to the environment such as Quixel's Megascans (Quixel, 2019) [56], a part of the Book of the Dead: Environment pack were selectively placed throughout the scene with regards to the scene's imperfections and a less repetitive, well-kept aesthetic (See Figure 28).

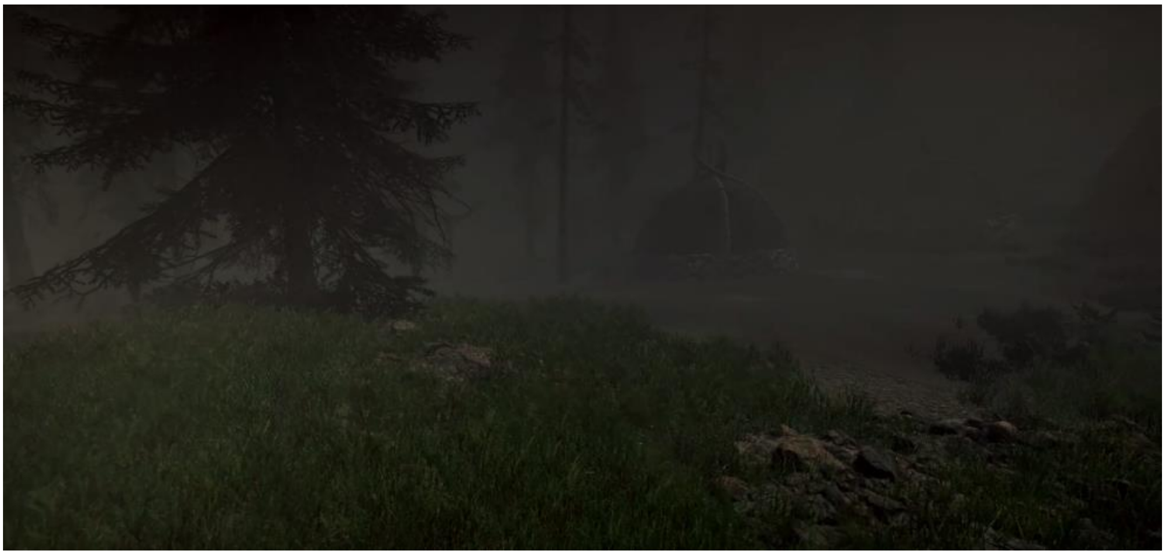
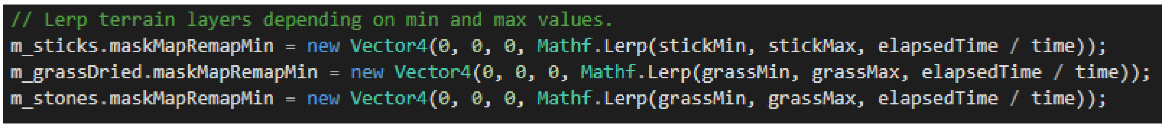


Figure 28 Demonstrates additional foliage and rocks implemented into the environment.

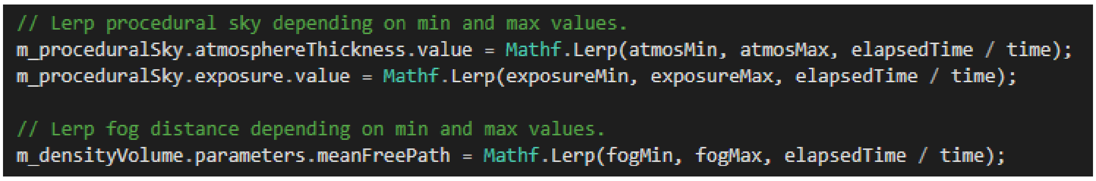
The rain was set to fall at random intervals and switch off after a random amount of time had elapsed. Upon the rain toggling on, terrain textures and environmental objects would dampen to create the illusion of the rain dampening its surroundings. Remapping the minimum mask map Z value resulted in increased smoothing of the terrain textures (sticks, dried grass, and stones) which could be linearly interpolated for a smooth transition (See Figure 29).



```
// Lerp terrain layers depending on min and max values.
m_sticks.maskMapRemapMin = new Vector4(0, 0, 0, Mathf.Lerp(stickMin, stickMax, elapsedTime / time));
m_grassDried.maskMapRemapMin = new Vector4(0, 0, 0, Mathf.Lerp(grassMin, grassMax, elapsedTime / time));
m_stones.maskMapRemapMin = new Vector4(0, 0, 0, Mathf.Lerp(grassMin, grassMax, elapsedTime / time));
```

Figure 29 Terrain textures mask map being linearly interpolated over time.

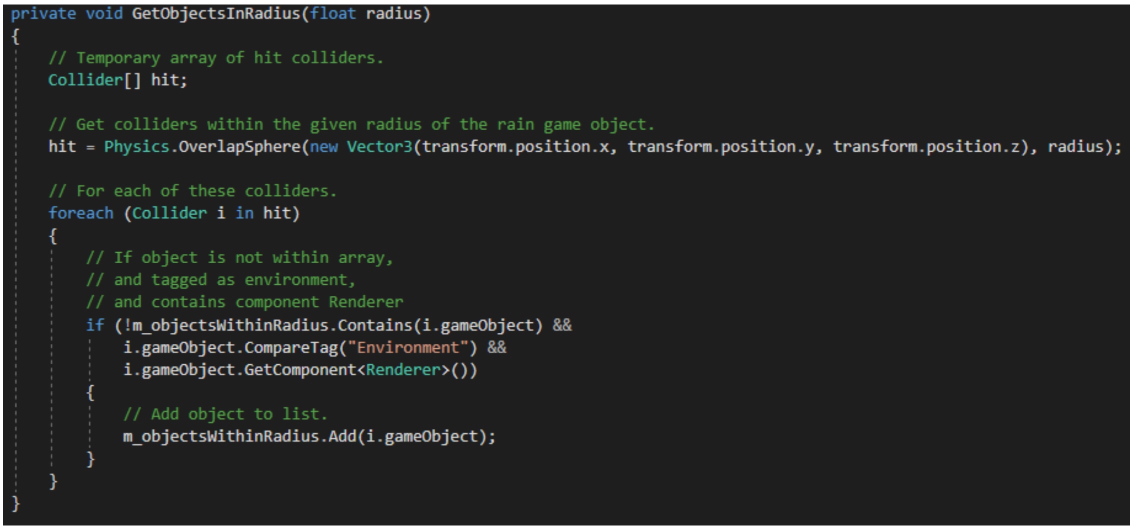
Alongside the changes to these variables when the rain falls, the atmospheric thickness and exposure of the procedural sky were interpolated, and the distance of the fog within the density volume was decreased which was represented as the 'Mean Free Path' (See Figure 30).



```
// Lerp procedural sky depending on min and max values.
m_proceduralSky.atmosphereThickness.value = Mathf.Lerp(atmosMin, atmosMax, elapsedTime / time);
m_proceduralSky.exposure.value = Mathf.Lerp(exposureMin, exposureMax, elapsedTime / time);

// Lerp fog distance depending on min and max values.
m_densityVolume.parameters.meanFreePath = Mathf.Lerp(fogMin, fogMax, elapsedTime / time);
```

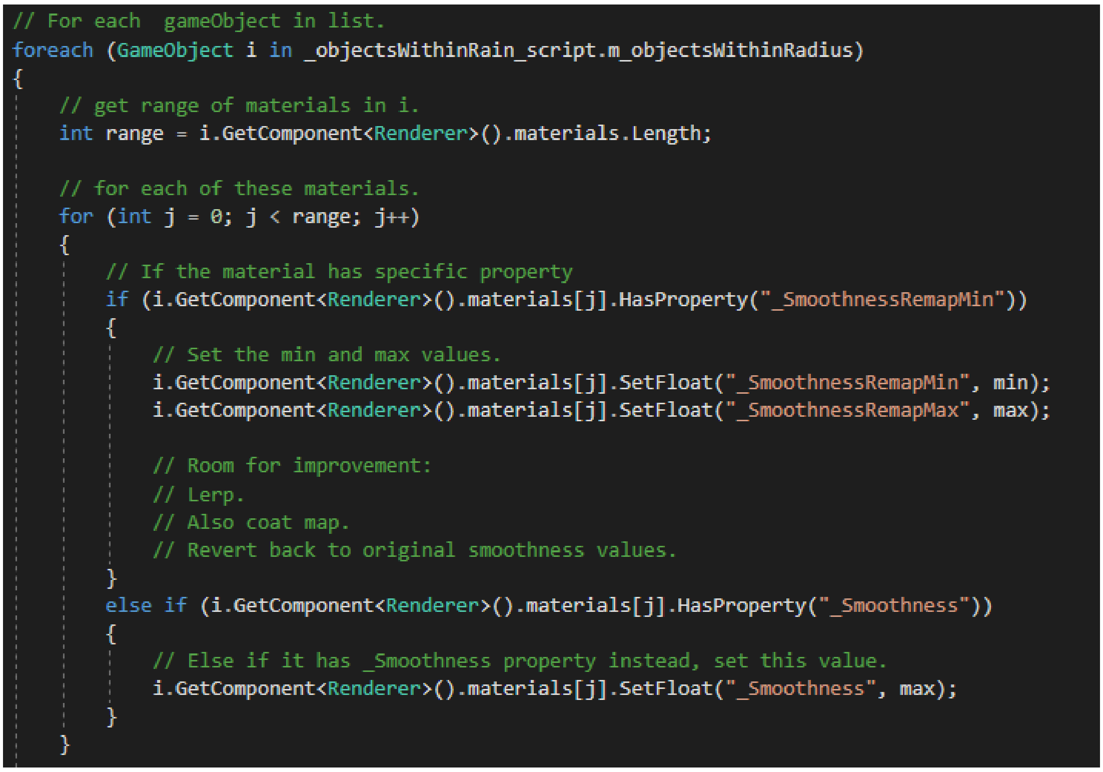
Figure 30 Procedural sky and density volume settings interpolated over time due to rainfall.

Other objects in the scene such as rocks, foliage, and huts also needed their smoothness values remapped too, however interpolating all of these objects at one time resulted in frame rate spikes. Instead, objects within the rain collider using the tag 'Environment,' with a renderer component attached would be added to a game objects list (See Figure 31). When the rain particle effect is toggled, the materials of all objects within the list have their smoothness values modified. Both smoothness value mapping techniques were appropriately accounted for ('_SmoothnessRemapMin (or Max)' and '_Smoothness'), and also a loop to affect all materials of the selected object, rather than just its first material (See Figure 32). Furthermore, if the rain is paused, then the smoothness values of materials are reset, and the objects are removed from the list.



```
private void GetObjectsInRadius(float radius)
{
    // Temporary array of hit colliders.
    Collider[] hit;

    // Get colliders within the given radius of the rain game object.
    hit = Physics.OverlapSphere(new Vector3(transform.position.x, transform.position.y, transform.position.z), radius);

    // For each of these colliders.
    foreach (Collider i in hit)
    {
        // If object is not within array,
        // and tagged as environment,
        // and contains component Renderer
        if (!m_objectsWithinRadius.Contains(i.gameObject) &&
            i.gameObject.CompareTag("Environment") &&
            i.gameObject.GetComponent<Renderer>())
        {
            // Add object to list.
            m_objectsWithinRadius.Add(i.gameObject);
        }
    }
}
```

Figure 31 Detecting objects within a given radius of the rain particle effect object, adding them to a list, if not already.

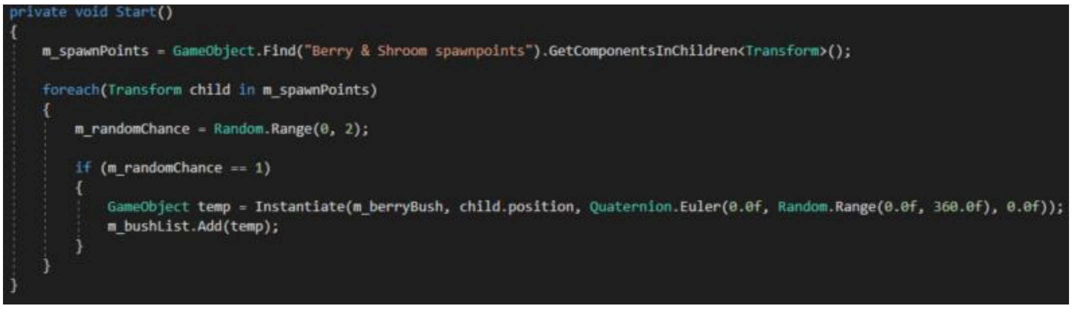


```
// For each  gameObject in list.
foreach (GameObject i in _objectsWithinRain_script.m_objectsWithinRadius)
{
    // get range of materials in i.
    int range = i.GetComponent<Renderer>().materials.Length;

    // for each of these materials.
    for (int j = 0; j < range; j++)
    {
        // If the material has specific property
        if (i.GetComponent<Renderer>().materials[j].HasProperty("_SmoothnessRemapMin"))
        {
            // Set the min and max values.
            i.GetComponent<Renderer>().materials[j].SetFloat("_SmoothnessRemapMin", min);
            i.GetComponent<Renderer>().materials[j].SetFloat("_SmoothnessRemapMax", max);

            // Room for improvement:
            // Lerp.
            // Also coat map.
            // Revert back to original smoothness values.
        }
        else if (i.GetComponent<Renderer>().materials[j].HasProperty("_Smoothness"))
        {
            // Else if it has _Smoothness property instead, set this value.
            i.GetComponent<Renderer>().materials[j].SetFloat("_Smoothness", max);
        }
    }
```

Figure 32 Changing materials smoothness values of each object within the list, using the rain transition script.

10.9. Sprint 9 – Ending 08/04/2019

To engross an audience into the game, it must be replayable with a dynamic environment with randomly occurring events. For example; the current sprint instantiated a random number of berry bushes into the scene, at predetermined spawn locations (See Figure 33).



```
private void Start()
{
    m_spawnPoints = GameObject.Find("Berry & Shroom spawnpoints").GetComponentsInChildren<Transform>();

    foreach(Transform child in m_spawnPoints)
    {
        m_randomChance = Random.Range(0, 2);

        if (m_randomChance == 1)
        {
            GameObject temp = Instantiate(m_berryBush, child.position, Quaternion.Euler(0.0f, Random.Range(0.0f, 360.0f), 0.0f));
            m_bushList.Add(temp);
        }
    }
}
```

Figure 33 Berry spawn script instantiating berry bush prefabs at predefined spawn locations at random, with variations of rotation in the Y axis.

The berry bush will spawn berries if there is rainfall overhead and will proceed to invoke numerous methods over the lifespan of the instantiated prefab. The berries were set to grow old, and then die off after a random amount of time (See Figure 34). Fresh, ripe berries replenished the player's sanity by twenty, while the older berries would decrease the player's sanity by ten; it is up to the player to decide whether the berries are edible.

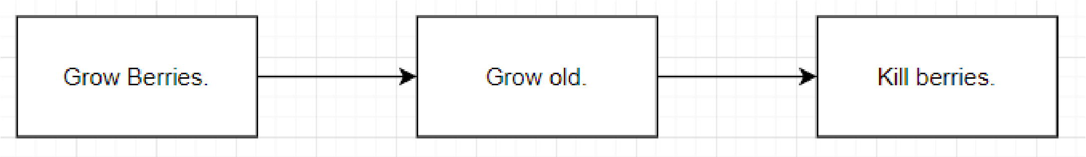
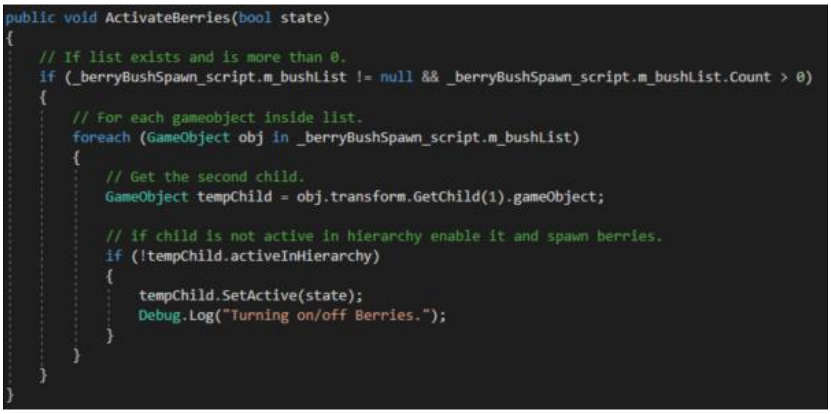


Figure 34 Berry lifetime after being enabled by rainfall.

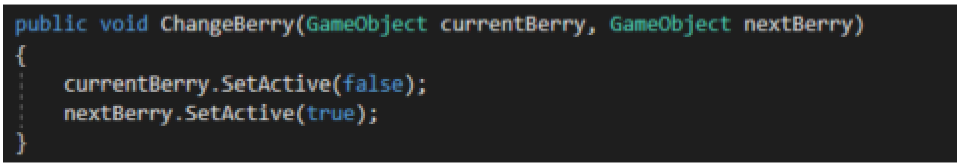
Berries were spawned via the rain transition script, calling the berry state public methods for activating all berries (See Figures 35 and 36).



```
public void ActivateBerries(bool state)
{
    // If list exists and is more than 0.
    if (_berryBushSpawn_script.m_bushList != null && _berryBushSpawn_script.m_bushList.Count > 0)
    {
        // For each gameobject inside list.
        foreach (GameObject obj in _berryBushSpawn_script.m_bushList)
        {
            // Get the second child.
            GameObject tempChild = obj.transform.GetChild(1).gameObject;

            // if child is not active in hierarchy enable it and spawn berries.
            if (!tempChild.activeInHierarchy)
            {
                tempChild.SetActive(state);
                Debug.Log("Turning on/off Berries.");
            }
        }
    }
}
```

Figure 35 Berry states script that contains methods for enabling berries.

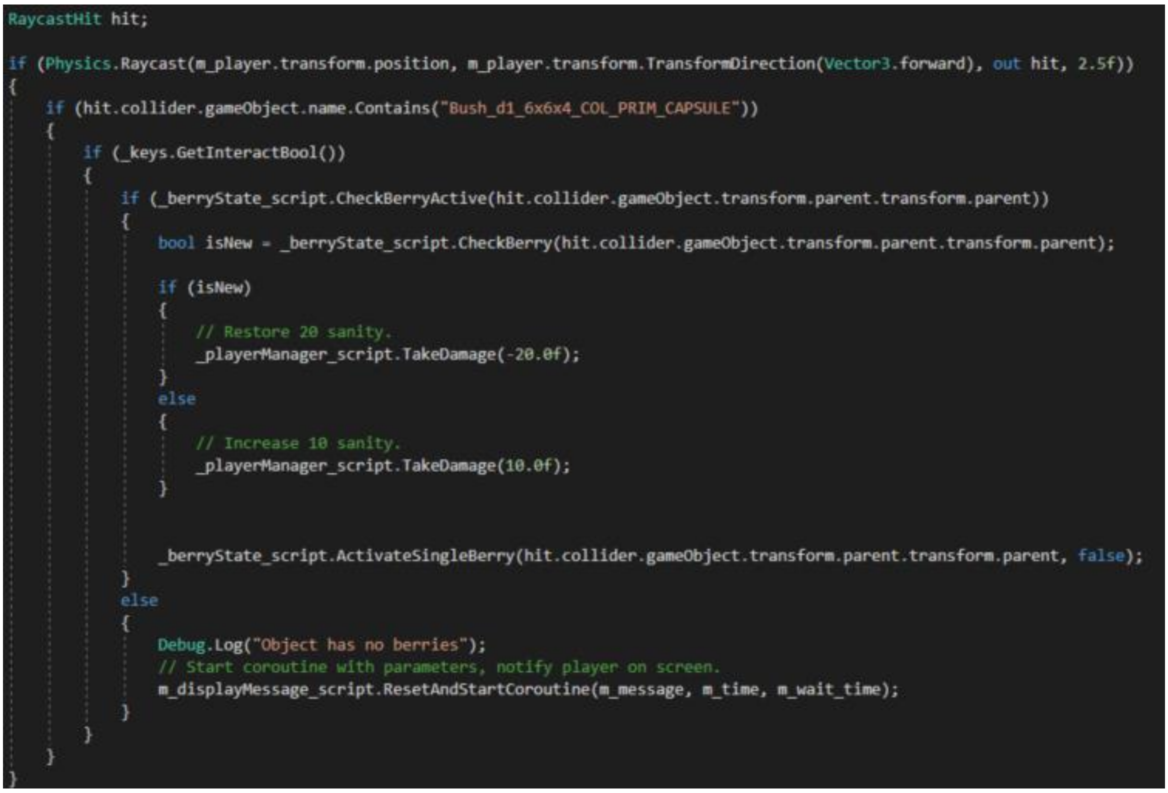
After some time as defined by the berry timer script, the method to change the berries state is invoked, which disables the ripe berry prefab, and enables the unhealthier alternative.



```
public void ChangeBerry(GameObject currentBerry, GameObject nextBerry)
{
    currentBerry.SetActive(false);
    nextBerry.SetActive(true);
}
```

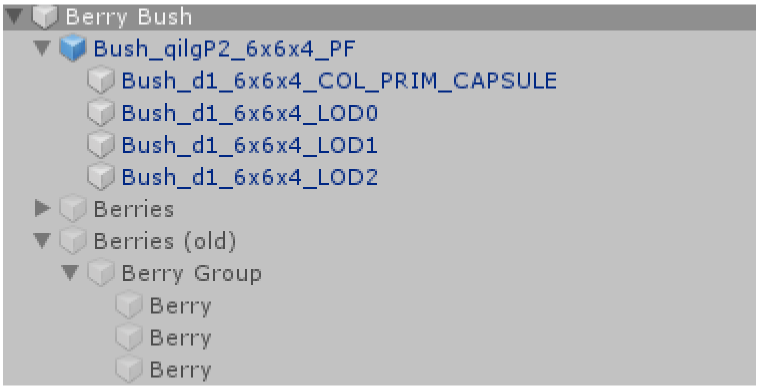
Figure 36 Switch the berry prefabs after time elapsed.

A raycast is deployed to detect a berry bush when the assigned interact key is pressed, which concludes three outcomes. If there are no berries on the bush, the display message script is called, and dialogue will appear in the user interface to tell the player no berries are spawned on the berry bush. The alternative outcomes are the decrease or increase insanity depending on the berry prefab currently enabled (See Figure 37). Due to the complexity of the berry bush prefab, multiple checks are required to determine the correct child prefabs to enable, hence the elongated 'transform. parent' parameter in the 'CheckBerry' method calls (See Figure 38).



```
RaycastHit hit;

if (Physics.Raycast(m_player.transform.position, m_player.transform.TransformDirection(Vector3.forward), out hit, 2.5f))
{
    if (hit.collider.gameObject.name.Contains("Bush_d1_6x6x4_COL_PRIM_CAPSULE"))
    {
        if (_keys.GetInteractBool())
        {
            if (_berryState_script.CheckBerryActive(hit.collider.gameObject.transform.parent.transform.parent))
            {
                bool isNew = _berryState_script.CheckBerry(hit.collider.gameObject.transform.parent.transform.parent);

                if (isNew)
                {
                    // Restore 20 sanity.
                    _playerManager_script.TakeDamage(-20.0f);
                }
                else
                {
                    // Increase 10 sanity.
                    _playerManager_script.TakeDamage(10.0f);
                }


                _berryState_script.ActivateSingleBerry(hit.collider.gameObject.transform.parent.transform.parent, false);
            }
            else
            {
                Debug.Log("Object has no berries");
                // Start coroutine with parameters, notify player on screen.
                m_displayMessage_script.ResetAndStartCoroutine(m_message, m_time, m_wait_time);
            }
        }
    }
}
```

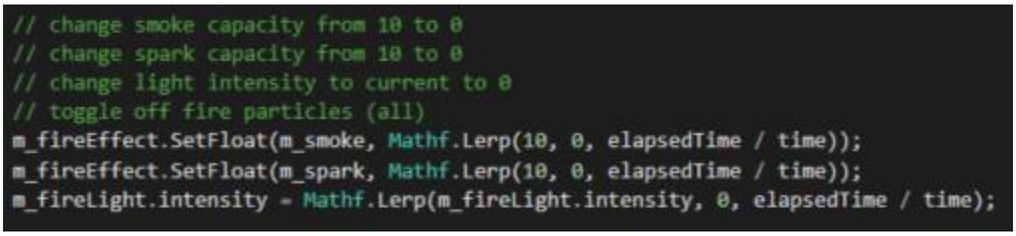
Figure 37 Raycast to check berry bush prefab using the eat berry script.



```
Berry Bush
  Bush_qilgP2_6x6x4_PF
    Bush_d1_6x6x4_COL_PRIM_CAPSULE
    Bush_d1_6x6x4_LOD0
    Bush_d1_6x6x4_LOD1
    Bush_d1_6x6x4_LOD2
  Berries
  Berries (old)
    Berry Group
      Berry
      Berry
      Berry
```

Figure 38 Berry bush prefab child objects. 'Berries' are considered to be the ripe berries, while 'Berries (old)' are the older variation.

A fire effect was created using the Visual Effects Graph (VFX) to finalize the sprint. This fire was planned to regenerate the player's sanity overtime when nearby but will die out in rainfall. These mini objectives should keep the player interested while trying to achieve the main objective, of surviving until dusk. Research into Brackey's tutorial (YouTube, 2019) [57] provided the necessary third-party Wispy smoke assets (Anon, 2019) [58] to create the fire, which consisted of a warm coloured point light, smoke, and sparks (VFX Graph [Lit] Quad Outputs). Smoke and spark capacity variables were exposed so that it transitions based on the current state of rain and its whereabouts, as well as the interpolation of the light intensity (See Figure 39).



```
// change smoke capacity from 10 to 0
// change spark capacity from 10 to 0
// change light intensity to current to 0
// toggle off fire particles (all)
m_fireEffect.SetFloat(m_smoke, Mathf.Lerp(10, 0, elapsedTime / time));
m_fireEffect.SetFloat(m_spark, Mathf.Lerp(10, 0, elapsedTime / time));
m_fireLight.intensity = Mathf.Lerp(m_fireLight.intensity, 0, elapsedTime / time);
```

Figure 39 Linearly interpolating the capacity of the exposed fire parameters and light component parameters.

10.10. Sprint 10 – 15/05/2019

The final sprint was conducted over the final weeks of the project, which entailed the development of finalized fire mechanics, whereby the player could relight the fire by grabbing wood from the log pile and interacting with the stone mound. However due to poor visibility due to the volumetric fog, overlaying world space graphical user interface components were required using a second camera as child to the main camera, that only rendered UI and post processing layers. The UI artwork included wooden logs (DeviantArt, 2019) [59], fire (Anon., 2019) [60], and a bush (UI Ex, 2019) [61] so that all interaction elements could be found.

Secondly, navigation between scenes and a menu system was implemented. The new scenes included; win condition, lose condition, main menu, settings, and finally controls. Buttons used third-party font (1001fonts.com, 2019) [62] and artwork (Anon., 2019) [63].

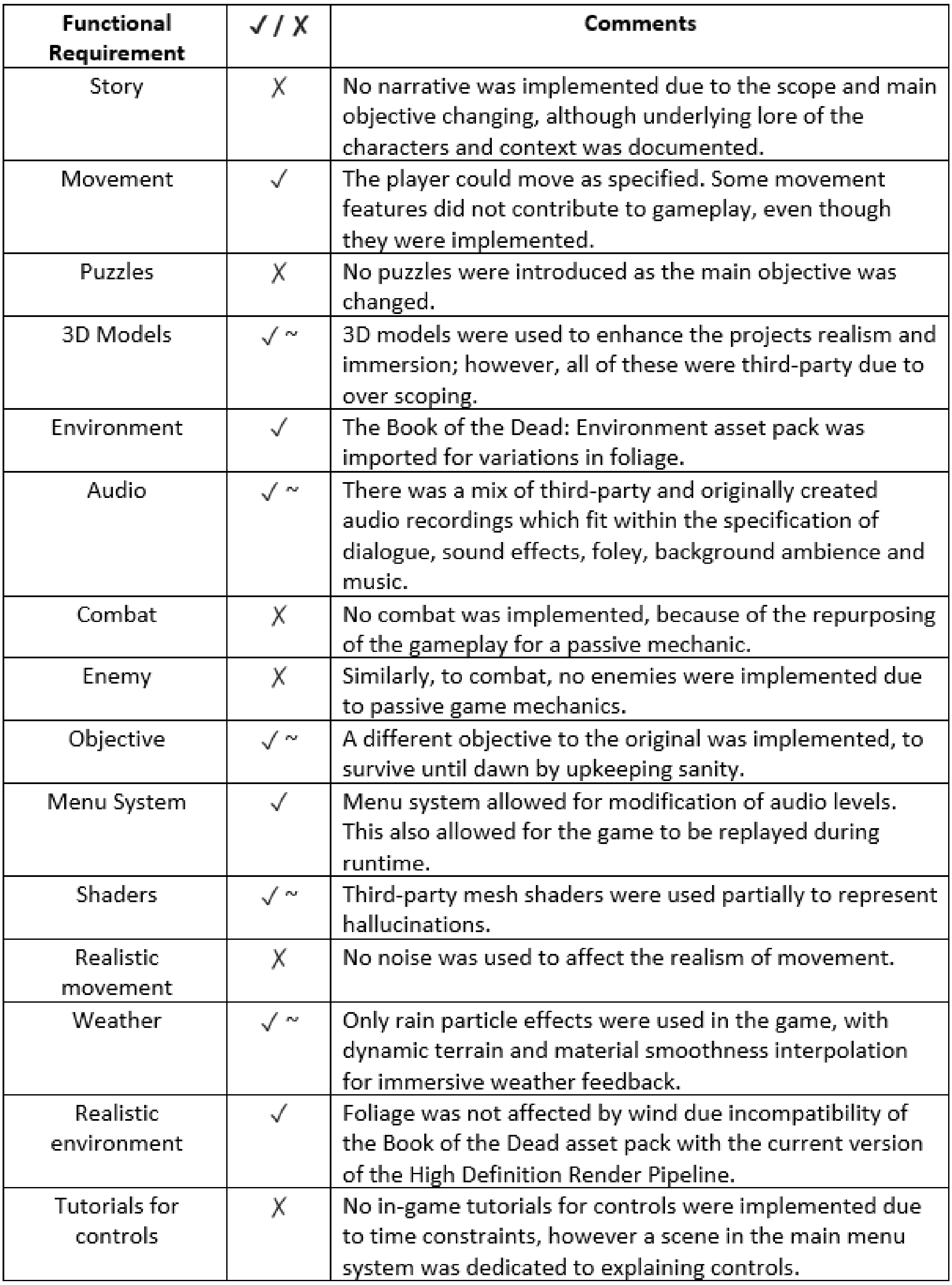
Sound was also designed within this last sprint (Footstep spot effects, fire, rain, distant animal noises and the creaking of trees), using Audacity to overlay and filter originally created spot effects. The initial scope proposed binaural recordings; however, the Unity Game Engine has spatial sound capabilities, and therefore audio was individually instantiated within the player's proximity or orbiting the player's position to mimic delusional thoughts representative of the acute phase of psychosis.

Finally, a day and night cycle brought the project to an end. Once the specified time limit was reached by a coroutine (after dusk) the SceneManager would load the win condition. Lighting had become a problem as the procedural sky would not fulfil significant exposure through the volumetric fog, hence the remove of the night cycle from the final release.

11. Functionality Testing

Tests were carried out to establish if the functional requirements were met at the end of the development, however iterative testing was also carried out throughout with a designated column for bugs, free from release on the Trello board.

Table 2 Assessing functional requirements of the final release.



Functional Requirement	✓ / X	Comments
Story	X	No narrative was implemented due to the scope and main objective changing, although underlying lore of the characters and context was documented.
Movement	✓	The player could move as specified. Some movement features did not contribute to gameplay, even though they were implemented.
Puzzles	X	No puzzles were introduced as the main objective was changed.
3D Models	✓ ~	3D models were used to enhance the projects realism and immersion; however, all of these were third-party due to over scoping.
Environment	✓	The Book of the Dead: Environment asset pack was imported for variations in foliage.
Audio	✓ ~	There was a mix of third-party and originally created audio recordings which fit within the specification of dialogue, sound effects, foley, background ambience and music.
Combat	X	No combat was implemented, because of the repurposing of the gameplay for a passive mechanic.
Enemy	X	Similarly, to combat, no enemies were implemented due to passive game mechanics.
Objective	✓ ~	A different objective to the original was implemented, to survive until dawn by upkeeping sanity.
Menu System	✓	Menu system allowed for modification of audio levels. This also allowed for the game to be replayed during runtime.
Shaders	✓ ~	Third-party mesh shaders were used partially to represent hallucinations.
Realistic movement	X	No noise was used to affect the realism of movement.
Weather	✓ ~	Only rain particle effects were used in the game, with dynamic terrain and material smoothness interpolation for immersive weather feedback.
Realistic environment	✓	Foliage was not affected by wind due incompatibility of the Book of the Dead asset pack with the current version of the High Definition Render Pipeline.
Tutorials for controls	X	No in-game tutorials for controls were implemented due to time constraints, however a scene in the main menu system was dedicated to explaining controls.

12. End-project Report

The project differs mostly from the initial scope and therefore will be analysed regarding the project objectives as listed below. Data has also been gathered from usability testing which discusses specifics of each of the objectives. The initial scope was defined to be a narrative-based first-person action-adventure game representing mental illness; the final result was a first-person survival game with the upkeep of sanity to be the main objective for the player to overcome, a significant difference due to the incremental progression of the ScrumBan methodology.

1. **Create a game that immerses the player in the environment.**

This objective was met concerning the visuals of the game however lacks in auditory immersion. The assets were all using physics-based rendering which was sourced from the Unity Asset Store; however, due to poor time management or over-scoping original assets were not modelled. The author experienced a steep learning curve with the usage of the High Definition Render Pipeline; however, the final results portray realistic lighting and volumetric fog without being computationally heavy. A good portion of the audio was sourced from royalty free websites, although these sounds were still demonstrating spatial sound in Unity which set the tone of the gameplay.

2. **Create a game that represents illness or schizophrenia.**

The representation of illness was achieved through linear interpolation of post-processing effects, implementation of binaural recordings and exposition to signify an unwell mental state, and player mechanics such as head rotation and clamping to induce anxiety. The prevention of a gradual decline of the character's sanity was the finalized game goal for the player. In retrospect, more could have been accomplished to represent the symptoms with regards to physics, and further character development through exposition as the game progresses.

3. **Use the ScrumBan workflow that has been adapted throughout University to manage the project via Trello efficiently.**

Thirdly, the ScrumBan workflow as proposed in Section 6 was partially effective as project management methodology for Trello, although the final release of the project may argue to the contrary. Functional requirements were split between releases and sprints accordingly, which then underwent the development in weekly or bi-weekly sprints concerning the difficulty or enormity of the features. The point system as referenced by the Kanban workflow was not taking into consideration in some portions of the project, whereby the development point limit was not adhered to because numerous sprints relied upon each other for completion. The improvement of such would depend on the correct grouping of coupled features into sprints, rather than features that are similar in mechanics; for example, the states feature relied upon some audio to be produced prior, however, the audio was allocated to a separate sprint.

4. **Keep an updated development vlog or blog, tweets, a press kit to market the game and let the target audience know of its existence before release.**

Finally, this objective was not met as the press kit was only published privately; however, a website with online developer presence was established which upheld a blog (Devliamw.webflow.io, 2019) [64] that the author posted to for the first few weeks of development as it was the cause of distraction, similar to the use of Twitter (Twitter.com) [65] for advertisement of uploaded blogs. The game was finally published to Itch.io, the target platform for release due to its sizeable indie development community.

13. Project Post-mortem

13.1. Product Specification

Objectives, outcomes, functional requirements, and deliverables contribute to product specification. A well-defined specification demonstrates an understanding of the overall goal of the project, however the role of methodology results in different lengths of specification documentation. The waterfall methodology would have resulted in an extended period before the development; however, with the utilization of an Agile approach, the specification can be updated throughout, which benefits the flexibility of scope.

The product specification was altered on numerous occasions throughout the project, which leads to a delay in development. This was due to an unclear specification and poor risk management; for example, falling back to previous ideas. Fewer features within the Minimum Awesome Product were delivered, and lead to a less fulfilling project; in retrospect, a full investigation into plausible deliverables for psychosis would have significantly improved time management.

13.2. Project Management Approach

Despite the uncertainty in the specification, the ScrumBan methodology was the superior choice over Waterfall. While Scrum on its own accord benefits game development teams, the combination with Kanban fulfils the promise of flow within the project for an individual. In return, performance is increased throughout sprints. However, this was not evident as backlog items were not successfully grouped into sprints of similar features; instead, multiple sprints were coupled in features, and the workflow was therefore interrupted and unorganized (usually multiple sprints items in development within a weekly sprint).

The weekly highlight reports and meetings with project supervisor established an unstandardized approach to retrospective meetings. Retrospective meetings were important for discussing what went badly, what went well and what could be improved in later sprints, however supervisory meetings added a layer of justification of features implemented during the week. Explaining the context of features, set-up for a purposeful report and research topics.

The exposure of ScrumBan while working individually differed to previous projects using the same methodology, but within a team. This was due to partial unfamiliarity with solo development projects, although the experience gathered from such will benefit future projects.

14. Conclusions

To conclude, the representation of mental health in video games proves to be a larger scope than anticipated. The failure of accurately portraying symptoms relating to psychosis was due to an unfortunate first few weekly sprints which amounted to a lack of content. The Agile ScrumBan methodology caused iterations onto the functional requirements at early stages due to uncertainty in scope.

Despite this, research and development skills were acquired over the project's course, gaining experience with the use of the upcoming High Definition Render Pipeline on Unity3d for increased realism and exclusive features. After the fundamentals of the HDRP were overcome, visual immersion became a large success for the project in regard to volumetric fog, advanced lighting rendering paths, Post-processing Effects, and Visual Effects Graph.

The project has offered insight into a magnitude of skills from key aspects of methodology, architecture, and furthering my knowledge on the legal, social, ethical and professional implications. Games representative of mental health problems combine both entertainment and educational systems into fun, or therapeutic experiences, and there is room to explore this further in future development endeavours.

15. References

[1] Cullen, M. (2018). *Basics of Sound Design for Video Games*. [PDF] Available at: https://frost.ics.uci.edu/ics62/BasicsofSoundDesignforVideoGames-MichaelCullen.pdf [Accessed 13 Dec. 2018].

[2] Helpguide.org. (2018). *Schizophrenia Symptoms, Signs, and Coping Tips: How to Recognize Schizophrenia and Get the Help You Need*. [online] Available at: https://www.helpguide.org/articles/mental-disorders/schizophrenia-signs-and-symptoms.htm [Accessed 12 Dec. 2018].

[3] Store.steampowered.com. (2018). *Fran Bow on Steam*. [online] Available at: https://store.steampowered.com/agecheck/app/362680/ [Accessed 13 Dec. 2018].

[4] Franbow.com. (2018). *Fran Bow*. [online] Available at: http://www.franbow.com/ [Accessed 12 Dec. 2018].

[5] Drifter, H. (2018). *Hyper Light Drifter on Steam*. [online] Store.steampowered.com. Available at: https://store.steampowered.com/app/257850/Hyper_Light_Drifter/ [Accessed 13 Dec. 2018].

[6] Hellblade.com. (2018). *Hellblade | Ninja Theory*. [online] Available at: https://www.hellblade.com/ [Accessed 13 Dec. 2018].

[7] Technologies, U. (2019). *Unity - Unity*. [online] Unity. Available at: https://unity.com/ [Accessed 30 Apr. 2019].

[8] Unrealengine.com. (2019). *Unreal Engine | What is Unreal Engine 4*. [online] Available at: https://www.unrealengine.com/en-US/ [Accessed 30 Apr. 2019].

[9] E. Dickson, Paul & E. Block, Jeremy & N. Echevarria, Gina & C. Keenan, Kristina. (2017). An Experience-based Comparison of Unity and Unreal for a Stand-alone 3D Game Development Course. 70-75. 10.1145/3059009.3059013.

[10] Epic Games. (2019). *Epic Games | Home*. [online] Available at: https://www.epicgames.com/site/en-US/home [Accessed 30 Apr. 2019].

[11] Pachoulakis, I. and Pontikakis, G. (n.d.). Combing features of the Unreal and Unity Game Engines to hone development skills. p.1.

[12] Assetstore.unity.com. (2019). *Unity Asset Store - The Best Assets for Game Making*. [online] Available at: https://assetstore.unity.com [Accessed 30 Apr. 2019].

[13] Unity. (2019). *Unity Terms of Service - Unity*. [online] Available at: https://unity3d.com/legal/terms-of-service [Accessed 30 Apr. 2019].

[14] Unrealengine.com. (2019). *Marketing Distribution Agreement*. [online] Available at: https://www.unrealengine.com/en-US/marketplace-distribution-agreement [Accessed 30 Apr. 2019].

[15] Unreal Engine. (2019). *Epic Announces Unreal Engine Marketplace 88% / 12% Revenue Share*. [online] Available at: https://www.unrealengine.com/en-US/blog/epic-announces-unreal-engine-marketplace-88-12-revenue-share [Accessed 30 Apr. 2019].

[16] Lagarde, S. (2019). *The High Definition Render Pipeline: Focused on visual quality – Unity Blog*. [online] Unity Technologies Blog. Available at: https://blogs.unity3d.com/2018/03/16/the-high-definition-render-pipeline-focused-on-visual-quality/ [Accessed 9 May 2019].

[17] Cooper, T. (2019). *The Lightweight Render Pipeline: Optimizing Real Time Performance – Unity Blog*. [online] Unity Technologies Blog. Available at: https://blogs.unity3d.com/2018/02/21/the-lightweight-render-pipeline-optimizing-real-time-performance/ [Accessed 9 May 2019].

[18] Visual Studio. (2019). *Visual Studio IDE, Code Editor, Azure DevOps, & App Center - Visual Studio*. [online] Available at: https://visualstudio.microsoft.com/ [Accessed 9 May 2019].

[19] Monodevelop.com. (2019). *MonoDevelop | MonoDevelop*. [online] Available at: https://www.monodevelop.com/ [Accessed 9 May 2019].

[20] GitHub. (2019). *Build software better, together*. [online] Available at: https://github.com/ [Accessed 9 May 2019].

[21] Hu, Y., Zhang, J., Bai, X., Yu, S. and Yang, Z. (2016). Influence analysis of Github repositories. *SpringerPlus*, 5(1).

[22] GitHub Desktop. (2019). *GitHub Desktop*. [online] Available at: https://desktop.github.com/ [Accessed 9 May 2019].

[23] Git Large File Storage. (2019). *Git Large File Storage*. [online] Available at: https://git-lfs.github.com/ [Accessed 9 May 2019].

[24] Help.github.com. (2019). *What is my disk quota? - GitHub Help*. [online] Available at: https://help.github.com/en/articles/what-is-my-disk-quota [Accessed 9 May 2019].

[25] Audacity, (2019). [online] Available at: https://www.audacityteam.org/ [Accessed 10 May 2019].

[26] Maijala, P. (1997). Better binaural recordings using the real human head. *INTERNOISE*.

[27] Lane, J. D., Kasian, S. J., Owens, J. E., & Marsh, G. R. (1998). Binaural Auditory Beats Affect Vigilance Performance and Mood. Physiology & Behavior, 63(2), 249–252. doi:10.1016/s0031-9384(97)00436-8

[28] PreSonus. (2019). *Free PreSonus Studio One 3 Prime Perfect for Beginners and Students | Press Releases | PreSonus*. [online] Available at: https://www.presonus.com/press/press-releases/Studio-One-3-Prime [Accessed 10 May 2019].

[29] Adobe.com. (2019). *Buy Adobe Photoshop | Best photo, image and design editing software*. [online] Available at: https://www.adobe.com/uk/products/photoshop.html [Accessed 10 May 2019].

[30] GIMP. (2019). *GIMP*. [online] Available at: https://www.gimp.org/ [Accessed 10 May 2019].

[31] Balaji, S. and Murugaiyan, M. (2012). Waterfall Vs V-Model Vs Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management*, 2(1), pp.26 - 29.

[32] Winston, D. and Royce, W. (1970). Managing the development of large software systems. *The Institution of Electrical and Electronics Engineers*, pp.328 - 338.

[33] Larman, C., & Basili, V. R. (2003). Iterative and incremental developments. a brief history. Computer, 36(6), 47–56. doi:10.1109/mc.2003.1204375

[34] Cervone, H. F. (2011). Understanding agile project management methods using Scrum. OCLC Systems & Services: International Digital Library Perspectives, 27(1), 18–22. doi:10.1108/10650751111106528

[35] Anderson, D. and Reinertsen, D. (2010). *Kanban*. Sequim: Blue Hole Press.

[36] Reddy, A., Anderson, D. and Benson, J. (2016). *The Scrumban [r]evolution*. New York: Addison-Wesley.

[37] Unity. (2019). *Asset Store Terms of Service and EULA - Unity*. [online] Available at: https://unity3d.com/legal/as_terms [Accessed 12 May 2019].

[38] Unity. (2019). *Unity Companion License ("License") - Unity*. [online] Available at: https://unity3d.com/legal/licenses/Unity_Companion_License [Accessed 12 May 2019].

[39] Creativecommons.org. (2019). *Creative Commons — CC0 1.0 Universal*. [online] Available at: https://creativecommons.org/publicdomain/zero/1.0/ [Accessed 12 May 2019].

[40] Pegi.info. (2019). *Home | Pegi Public Site*. [online] Available at: https://pegi.info/ [Accessed 12 May 2019].

[41] ParentInfo. (2019). *PEGI games ratings explained*. [online] Available at: https://parentinfo.org/article/pegi-games-ratings-explained [Accessed 12 May 2019].

[42] Trello.com. (2019). *Trello*. [online] Available at: https://trello.com/ [Accessed 12 May 2019].

[43] Kanbanflow.com. (2019). *KanbanFlow - Lean project management. Simplified.*. [online] Available at: https://kanbanflow.com/ [Accessed 12 May 2019].

[44] elegantt.com. (2019). *elegantt.com*. [online] Available at: https://elegantt.com/ [Accessed 12 May 2019].

[45] Scrumfortrello.com. (2019). *Scrum for Trello - the free Trello Scrum extension for Chrome, Firefox, and Safari*. [online] Available at: http://scrumfortrello.com/ [Accessed 12 May 2019].

[46] Burndown for Trello. (2019). *Burndown Charts for Trello*. [online] Available at: https://www.burndownfortrello.com/ [Accessed 12 May 2019].

[47] Frictionalgames.blogspot.com. (2019). 4-Layers, A Narrative Design Approach. [online] Available at: https://frictionalgames.blogspot.com/2014/04/4-layers-narrative-design-approach.html [Accessed 20 Apr. 2019].

[48] Snyder, A. (1986). Encapsulation and inheritance in object-oriented programming languages. ACM SIGPLAN Notices, 21(11), 38–45. doi:10.1145/960112.28702

[49] Assetstore.unity.com. (2019). *Book Of The Dead: Environment - Asset Store*. [online] Available at: https://assetstore.unity.com/packages/essentials/tutorial-projects/book-of-the-dead-environment-121175 [Accessed 22 Apr. 2019].

[50] Assetstore.unity.com. (2019). *Boneskin Settlement Pack - Asset Store*. [online] Available at: https://assetstore.unity.com/packages/3d/environments/fantasy/boneskin-settlement-pack-54069 [Accessed 22 Apr. 2019].

[51] Assetstore.unity.com. (2019). *Mesh Effects - Asset Store*. [online] Available at: https://assetstore.unity.com/packages/vfx/particles/spells/mesh-effects-67803 [Accessed 22 Apr. 2019].

[52] Earlypsychosis.ca. (2019). *What is Psychosis? Phases of Psychosis - earlypsychosis.ca*. [online] Available at: https://www.earlypsychosis.ca/pages/curious/phases-of-psychosis [Accessed 21 Apr. 2019].

[53] Technologies, U. (2019). *Unity - Manual: Level of Detail (LOD)*. [online] Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/LevelOfDetail.html [Accessed 25 Apr. 2019].

[54] GitHub. (2019). *Slord6/WeatherSystem*. [online] Available at: https://github.com/Slord6/WeatherSystem [Accessed 25 Apr. 2019].

[55] Assetstore.unity.com. (2019). *Unity Particle Pack 5.x - Asset Store*. [online] Available at: https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-5-x-73777 [Accessed 25 Apr. 2019].

[56] Quixel. (2019). *Quixel Megascans*. [online] Available at: https://quixel.com/megascans/ [Accessed 26 Apr. 2019].

[57] YouTube. (2019). *FIRE AND SMOKE with Unity VFX Graph!* [online] Available at: https://www.youtube.com/watch?v=R6D1b7zZHHA [Accessed 26 Apr. 2019].

[58] Anon, (2019). [online] Available at: https://ole.unity.com/smoketextures [Accessed 26 Apr. 2019].

[59] DeviantArt. (2019). *Wood logs*. [online] Available at: https://www.deviantart.com/abunja/art/Wood-logs-696526292 [Accessed 16 May 2019].

[60] Anon. (2019). *Fire with background download free clipart with a transparent background*. [online] Available at: https://sitejerk.com/explore/fire-with-transparent-background.html [Accessed 16 May 2019].

[61] UI Ex. (2019). *brush drawing animated*. [online] Available at: https://ui-ex.com/explore/brush-drawing-animated/ [Accessed 16 May 2019].

[62] 1001fonts.com. (2019). *Good Foot Font · 1001 Fonts*. [online] Available at: https://www.1001fonts.com/good-foot-font.html [Accessed 16 May 2019].

[63] Anon. (2019). *Bloody hand smear download free clipart with a transparent background*. [online] Available at: https://sitejerk.com/explore/bloody-hand-smear-png.html [Accessed 16 May 2019].

[64] Devliamw.webflow.io. (2019). *Blog*. [online] Available at: https://devliamw.webflow.io/blog [Accessed 16 May 2019].

[65] Twitter.com. (2019). *Liam West (@DevLiamW) on Twitter*. [online] Available at: https://twitter.com/DevLiamW [Accessed 16 May 2019].

16. Appendices

Appendix A. User Guide

Recommended Computer Specification

Processor:	Intel® Core™ i7-7700HQ CPU @2.80GHz
Installed memory (RAM):	8.0GB
Graphics Card (GPU):	Nvidia GeForce GTX 1050 Ti

Unity Editor Specification (Source code)

Unity3d Version:	Unity 2018.3.3f1 (64bit)

Game Instructions

Navigate from the main directory into ‘Astray final build’ folder and run the executable.

Once the executable is running, navigate the menu scene as desired. After becoming knowledgeable of the controls, press ‘Start Game’.

The objective is to keep the sanity of the player above 0 until dusk.

Explore the environment and eat berries on bushes to sustain the player’s sanity. The colour of the berries will determine if they should be eaten.

Standing in close proximity of the fire will replenish your sanity over time, however this will go out when it rains.

The fire can be started again with firewood from the log pile.

Appendix B. Project Initiation Document (PID)

Project Initiation

Liam West

Title & Tagline

ASTRAY.

Tribal Stone-Age, action-adventure, immersive sound game representing mental illness for Microsoft Windows.

Introduction

The project will be a 3D first person, story-driven, action-adventure game, set in the late Stone Age. This will convey split narratives depending on the players choices, with consequences and outcomes that could affect the end narrative, giving a sense of immersion and value to every decision made. Implementation of sound design techniques such as binaural recording, and consideration of different sound families will greatly improve the depth of immersion; Dialogue, Music, Sound Effects, Foley, Backgrounds (ambience) (Cullen, 2018) [1]. These could also develop a psychological response to display symptoms of a looming illness of the player.

Schizophrenia can result in hallucinations, delusions, confused thoughts, and changes in behaviour which will gradually get worse over time (Helpguide.org, 2018) [2]. The idea that the mind could be overwhelmed with voices, delusional hallucinations, and misleading the player with spatial/binaural audio could result in a unique experience. The main goal will be situated towards this illness, whereby the narrative is a metaphor for the player trying to overcome their slowly deteriorating sanity.

Fran Bow (Fran Bow, 2015) [3] is a psychological horror adventure game that tells the story of a '… young girl struggling with a mental disorder and an unfair destiny.' (Franbow.com, 2018) [4]. The character suffers horrifying hallucinations represented through a multitude of techniques. The scenes themselves have 'clues' and interactable objects to both press the narrative forwards, but also sustain an immersive atmosphere. (Frictionalgames.blogspot.com, 2018) points out an arguable 4-Layer framework for narrative design. In terms of narrative background, Fran Bow adds 'Story fragments' which when stumbled upon will add to the main narrative. This is done seamlessly with small interactable objects along a set path; an unmissable part of the gameplay, but also a choice whether to skip idly past them if the player does not want to participate in the dialogue. Also, the developers of Fran Bow have thought about the players 'mental model', creating the sense of something lurking demonstrated in Markiplier's playthrough commentary (YouTube.com, 2018) [5] where it hints at the presence of a clown escaping the picture frame, but it is only to effectively portray the illness, and to keep the player weary of an imminent threat.

Another game that has introduced symptoms/illness/suffering to a game are the developers of Hyper Light Drifter (Hyper Light Drifter, 2016) [6]. The underlying meaning of the game is about heart disease and that the illness can strike you down at any moment, similarly to the developer's representation of his own life. (Rock Paper Shotgun, 2018) [7], which in-turn, if the player has a deeper understanding of this, will feel further the empathy towards the main character and the narrative that unfolds. The main goal is to find a cure for his disease, and this is reinforced throughout the scenes with the player coughing up blood, and visualizing hallucinations.

I plan to use Unity 2018.2.19 (7th December, 2018) (Unity.com, 2018) [8] for development as this includes two important tools for a polished game; The High Definition Render Pipeline provides an upgrade in lighting architecture and physically-based rendering, while Shader Graph provides a user-friendly method of creating shaders for the HDRP. (GitHub, 2018) [9]. Due to my experience with this Game Engine I should be able to easily adapt with a ScrumBan workflow that I have modified over the course of University, using Blender (Blender.org 2018) [10] for art assets. Sourcing assets from the Unity Asset store will also be applicable if I attain the rights to fair use of the product, which is stated in the Unity EULA agreement upon uploading assets to the store (Unity, 2018) [11].

Rationale

Market research suggests that there is only one viable competitor addressing schizophrenia symptoms and suffering. Schizophrenia Simulation by ChoppyPine (Itch.io, 2018) [12] is trying to achieve an accurate representation of the illness to press genuine anxiety onto the audience, utilizing both stereo channels for hearing voices, and creating some abstract visuals, that do not follow suit to real-life physics.

I would like to do something that Hellblade (Hellblade, 2017) [13] achieved with binaural recording to mimic voices moving around the players position or at varying proximity. I argued in my short paper during AINT354 (Spatial Sound and Psychological Response, 2018) that sound is on par with, if not more important than visuals of a game for narrative, gameplay and so forth. There is a gap in the market on Itch.io (Itch.io, 2018) [14], especially with the representation of illness to produce a story-driven experience that focuses on sound design.

However, Hellblade (Hellblade, 2017) [13] is a self-proclaimed AAA title, and therefore the competition will be ideally small Indie Game titles release on Itch.io such as Schizophrenia Simulation by ChoppyPines (Itch.io, 2018) [12], Fran Bow, and Super Light Drifter.

Project Outcomes

1. Effective sound design to create a psychological response. Binaural recordings, spatial sound, and other sound effects taking into consideration all sound families. Research has already been carried out on how spatial sound can create a psychological response; however, there is room for improvement, whereby what else in sound can create a psychological response, for example, binaural beats. This research and project open discussion on the topic.

2. The implementation and abstract representation of illness in an indie game. The author will need to further the author's knowledge of the illness to simulate an accurate depiction.

3. Git Merge workflows will be refined. Instead of the separate roles associated with the Scrum workflow, the author will partake in all activities such as Git repository management and project management. The skills obtained from this will help move forward into the industry.

4. A deeper understanding of how narrative and gameplay intertwine. The gameplay shall fit around the narrative rather than the narrative fitting within the scopes of the gameplay. This will ensure an immersive story experience, but also benefit skills for game design.

Project Objectives

1. Create a game that immerses the player in the environment.
    a. Game Visuals.
        i. Well-designed art assets with realistic materials, normal mapping and physics-based rendering. It can be either sourced on the Unity Asset Store or created with Blender.
        ii. Lighting using the High Definition Render Pipeline for realism.
        iii. Due to the illness affecting the player's mental state, the physics does not need to meet players expectations.
    b. Audio.
        i. Binaural recordings and Foley sounds recording and implemented into Unity 3d.
        ii. Spatial sound, and if possible Ambisonics added to Unity 3d for a 7.1/5.1 surround sound experience when the user wears a suitable headset.
        iii. Auditory cues and background ambience set the tone for scenes.

2. Create a game that represents illness or schizophrenia.

   a. Schizophrenia can be achieved with abstract game visuals, physics that does not meet expectations, and binaural recordings of whispers spinning around the player.
   b. Represent the gradual decline of sanity.

3. Use the ScrumBan workflow that has been adapted throughout University to manage the project via Trello efficiently.

4. Keep an updated development vlog or blog, tweets, a press kit to market the game and let the target audience know of its existence before release.

Initial Scope (Functional Requirements)

Minimum Viable Product

- Story.
    - The story will be a split narrative where the character must make decisions that could alter the ending of the story, therefore having real consequences. Since the story is a large part of the game, this needs to be completed first as this will outline the scenes. Gameplay will be blended in, depending on the story and how it unfolds.

- Movement.
    - The movement of the player shall be to the user's expectations. Looking at the direction of the mouse and W, A, S and D keys to move the position of the player.

- Puzzles.
    - Puzzles will be a part of the story and progress the character development, not just needless tasks in order to reach the primary goal. Research into puzzles needs to be documented before progressing into development.

- 3D Models.
    - 3D Models will enhance the realism of the project. All the models should fit within a similar art style and modular to the scene if possible, to minimize time cost. Research of Blender will need to be documented, due to previous skills with Autodesk 3Ds Max not transferring over. The rationale behind the use of Blender is that the release of assets using Blender comes with no restrictions, the author will benefit from revenue if they so choose to release with payment.

- Environment.
  - Variations of trees, vegetation, cliffs, and rocks will scatter the landscape to immerse the player into the world. Following the workflow and development of the trees in the Book of the Dead environmental asset pack by Unity.

- Audio.
  - Dialogue, Sound Effects, Foley, Backgrounds, and Music all need to be designed to create a fully immersive auditory experience. This all needs to set the tone of the story and the illness as it progresses. Music may be sourced from elsewhere due to inexperience with musical theory.
- Combat.
  - Two types of combat will be implemented. A bow and a melee range weapon. These will be the simple mechanics of the game as they are not the focus. They are to fend off enemies that will attack the player.

- Enemy.
  - The enemies might be real or hallucinations. The author would like to confuse the player and have them question their reality. Shaders will wrap into this, as the hallucinations will eventually linearly interpolate into non-existence, while the real instances of the enemies will be able to harm the player. This also makes the player anxious and weary at every encounter (It portrays what it may have been like in the late Stone Age where they had little understanding of the illness.). Enemies will either be species relevant to the period, or other hostile humans.

- Objective.
  - The main goal will be to find the characters tribe again. This will take the player on a journey while being affected by illness, hindering the chances of finding them. Each scene will have a smaller objective that will move the story forward.

- Menu System.
  - The menu system will enable a full game loop of the game. Including, main menu, settings, pause menu.

- Shaders.
  - Shaders will play a big part in the abstract visuals of the environment and the fading hallucinations. Due to the author’s experience in AINT354 with shaders, the author feels confident with producing shaders for the assets.

Minimum Awesome Product

- Realistic movement.
    - A refined movement that looks free flowing and realistic, with noise to adjust the movement of the camera, not just a static camera.

- Weather.
    - The author would like a dynamic weather system either sourced from the Unity Asset store or created from scratch. The weather will change and therefore reflecting how the illness has progressed to worse. Players anxiety will rise in response to this.

- Realistic environment.
    - The trees in the environment should sway with the wind as if in real life. This can be done with vertex manipulation like they have done so in the Book of the Dead environmental asset pack by Unity. The author would like to bring the whole scene to life instead of being a static environment.

- Tutorial for controls.
    - A dynamic approach for learning the controls the start of the game would be ideal, teaching the player as they play with hints or tips. The controls will be intuitive, so this will only be an extended goal of the project.

Method of Approach

I will be using an agile approach, taking items from the backlog and ordering them into releases and sprints. After doing so, I will go through each of these until completion of the prototype release within a 1 – 2-week(s) sprint. Testing will be incremental and on-going with the development process, constantly referring to the specification and adapting/modifying areas which need to change. This type of workflow falls under a ScrumBan approach, that I modified over the course of University and will be able to adapt it to an individual instead of a team.

The project will be developed using Unity 3d game engine and programmed in C#. Due to the experience and improved knowledge of this programme it seems ideal for development of this game. The HDRP and Shader Graph is specific to Unity, and the Unity asset store provides some 3D models and scripts, free-to-use. Sourcing these assets from the store will save time for the more important tasks such as focusing on the Unique Selling Point(s).

Blender will be used for 3D modelling instead of Autodesk 3Ds Max. 3Ds Max has a restriction on the Student License for releasing products for revenue, for a big expense I would need to purchase the Software License, and then export the assets. Hopefully, my skills will translate from 3Ds Max to Blender (free), however additional research and practise will need to be completed to adjust to the User Interface.

Initial Project Plan

Deadlines

Stage	**Start Date**	**End Date**	**Outcomes**
1. Initiation	7/12/2018	14/12/2018	This document.
2. Highlight report 1	29/01/2019	07/02/2019	Tree Narrative structure, Backlog creation, Git repository set-up.
3. Highlight report 2	08/02/2019	14/02/2019	Player, Movement, Enemy, Combat.
4. Highlight report 3	15/02/2019	21/02/2019	3D modelling, texturing.
5. Highlight report 4	22/02/2019	28/02/2019	Continued 3D modelling, texturing. Set-up in Unity with environment.
6. Highlight report 5	29/02/2019	07/03/2019	Audio recorded & added to Unity.
7. Highlight report 6	08/03/2019	14/03/2019	Shaders.
8. Highlight report 7	15/03/2019	21/03/2019	Puzzles (referring to Narrative)
9. Highlight report 8	22/03/2019	28/03/2019	Objectives. (referring to Narrative)
10. Poster	29/03/2019	04/04/2019	Poster, refined objective, game loop, game menus.
11. Draft report	05/04/2019	02/05/2019	Draft report, refinement of project, build, documentation finalised.
12. Final deliverable	03/05/2019	16/05/2019	Final report.

Control & Communication Plan

Sprints will be conducted on a weekly basis with a review with the supervisor, generating reports and goals for upcoming sprints.

Initial Risk List

Risks	Description
Over scoping.	The project is ambitious, with multiple elements needing to come together in order to achieve a full game loop. If necessary, assets may be sourced in order to achieve faster playable releases, such as sourcing 3D assets for prototyping the narrative structure and game mechanics.
Lack of narrative structure.	Due to inexperience in writing narratives for game design, there is a possibility that character development will not progress as envisioned, or that the narrative will not tie in with the gameplay. I will need a clear vision for the story before-hand so that the gameplay elements will sprout from this initial documentation.
Illness.	Illness may slow down progress. Supervisors will be notified and then I will need to account for catching up into my upcoming sprints.
Technology failure.	Code will be uploaded to a Git Repository and important files will also be stored externally so if the PC fails or data is lost, there will be a backup for these types of cases.

Quality Plan

Requirements	The Minimum Viable Product will be the focus of the project. This contains all requirements necessary for a fully immersive play through of the game, however, to increase the immersion and enjoyment of the game I proposed a Minimum Awesome Product, which identifies certain assets that could be improved if ahead of schedule.
Sprint Review	Tests will be performed at the end of each sprint, as well as throughout. If the output has failed to meet the requirements of the sprint, the features will be carried forth into the upcoming sprint and accounted for in the overall project development cycle.
Release	After completion of all releases, the project will be built and released onto Itch.io, and uploaded to SPMS with all deliverables included.

Legal, Ethical and Social Issues

Usability testing carried that will ask for the user to perform certain tasks on a computer. An interview will be conducted as well as a questionnaire. The user will remain strictly confidential and at no point will the users name or identification be used. The user will also be able to withdraw from consent and opt out from the participation at any time.

The PEGI-12 age rating is suitable here as there is non-realistic violence towards human-like characters, showing violence in a slightly more graphic nature.

Credit will be given the assets used in development. The assets sourced will be from the Unity Asset Store that states in their EULA that assets on the store are free-to-use unless specified otherwise. (Unity, 2018) [11].

Within Itch.io's terms-of-use they refer to the Digital Millennium Copyright act with addresses copyright of the product but only of the publishers on the website, so additional copyright should be specified within the repository and description of the game. (Itch.io, 2018) [15].

Outline GDD

1. **Title & Author**
   Astray by Liam West

2. **Target Platform(s)**
   Microsoft Windows

3. **Development Software**
   3.1. Engine Software
   Game Development - Unity 3d (https://unity3d.com/)
   3.2. Programming Software
   Coding - Microsoft Visual Studio (https://visualstudio.microsoft.com/)
   3.3. Artwork Software
   3D modelling - Blender (https://www.blender.org/)
   Texturing - Substance Painter (https://www.allegorithmic.com/products/substance-painter)
   Editing - Photoshop (https://www.adobe.com/uk/products/photoshop.html)

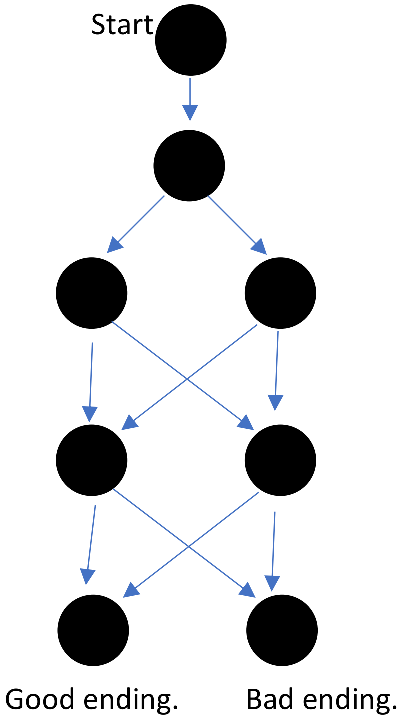
4. **Specification**
   4.1. Concept
   A first-person action-adventure story-driven game set in the Late Palaeolithic era (stone age). The player must find their way back to their tribe by travelling alone, showing symptoms of Schizophrenia. To do so they will be interrupted by hallucinations and puzzles mixed in with the narrative.
   4.2. Story
   The story will need to be developed further. However, the high-level overview of the narrative is of a male at the age of 18 situated in the Late Palaeolithic era (stone age) separated from his tribe after a hunt. He is already affected with Schizophrenia but comes more apparent as his journey progresses alone, slowly his sanity declines. He must choose his path carefully and find his tribe. His trails will split depending on this and will alter the outcome.

   2 endings so far;
   - Good ending: Find tribe.
   - Bad ending: Death. (Even though this ending will not be rewarding, I will make the emotional connection with the character worthwhile. The illness can make those affected suicidal and therefore the consequences should be so.)

   4.3. Setting
   Late Stone-Age, the wild, woods, vegetation caves.

4.4. Game Structure

Scene structure. Arrows represent path/choice taken. (Not revised yet)



4.5. Players

Single player.

4.6. Actions

- Explore.
- Solve puzzles.
- Experience story.
- Fight enemies.
- Immerse in the world.

4.7. Objectives

End goal: Find your tribe.

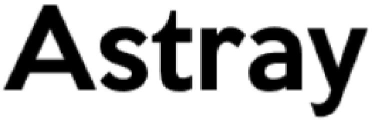
5. Graphics

5.1. Styles

Starts of light, however as the game progresses the atmosphere turns darker. Realism is a key style here to immerse the player.

5.2. Fonts

Keep Calm (https://www.dafont.com/keep-calm)


Astray

Clean, easy to read font. A font that is without a theme as themed font may look cartoonish.

5.3. Colours

Minimalistic colours. Perhaps these could change throughout the game.

5.4. Influences

Hellblade, 2017 [13]

Details: Art style, voices, binaural audio, illness, schizophrenia.

Fran Bow, 2015 [4]

Details: Illness, symptoms and visualization, narrative framework ideas

Hyper Light Drifter, 2016 [6]

Details: Representation of different diseases in games, symptoms. Loosely connected.

Schizophrenia Simulation, 2018 [12]
Details: Direct competitor for release, morphs physics, representation of illness. Field of view, distortion of reality.

6. **Look & Feel**

Realism and immersion will be a key part of the game so high-resolution assets will be required.

7. **Data Storage**

7.1. Local Data

Saving & Loading game data might be an option after the minimum viable product to further continuity in the game after closing it.

8. **Gameplay**

8.1. Object Types

Objects for combat, a melee weapon (spear), bow, arrows.
Will update when story is refined.

8.2. Controls

Keyboard
Mouse
(Xbox 360 controller)

8.3. Direct Control

W, A, S, D key(s) – Movement
E key – Interact
Space bar – Jump
Shift (hold) – Run
Left Mouse Button – Attack/Fire
Right Mouse Button (hold) – Block/Aim

8.4. Indirect Control

AI of enemies will usually attack player on sight or within a certain range of them. They may roam with the use of the Unity Nav Mesh.

9. **Supported Hardware**
Hopes of binding controls for Xbox 360 controller as refinement and additions to the minimum viable product.

References

[1] Cullen, M. (2018). *Basics of Sound Design for Video Games*. [PDF] Available at: https://frost.ics.uci.edu/ics62/BasicsofSoundDesignforVideoGames-MichaelCullen.pdf [Accessed 13 Dec. 2018].

[2] Helpguide.org. (2018). *Schizophrenia Symptoms, Signs, and Coping Tips: How to Recognize Schizophrenia and Get the Help You Need*. [online] Available at: https://www.helpguide.org/articles/mental-disorders/schizophrenia-signs-and-symptoms.htm [Accessed 12 Dec. 2018].

[3] Store.steampowered.com. (2018). *Fran Bow on Steam*. [online] Available at: https://store.steampowered.com/agecheck/app/362680/ [Accessed 13 Dec. 2018].

[4] Franbow.com. (2018). *Fran Bow*. [online] Available at: http://www.franbow.com/ [Accessed 12 Dec. 2018].

[5] YouTube. (2018). *Fran Bow | Part 1 | HORRORS BEYOND IMAGINATION*. [online] Available at: https://www.youtube.com/watch?v=seT7wHorSQg [Accessed 12 Dec. 2018].

[6] Drifter, H. (2018). *Hyper Light Drifter on Steam*. [online] Store.steampowered.com. Available at: https://store.steampowered.com/app/257850/Hyper_Light_Drifter/ [Accessed 13 Dec. 2018].

[7] Rock Paper Shotgun. (2018). *How A Lifetime Of Heart Disease Birthed Hyper Light Drifter*. [online] Available at: https://www.rockpapershotgun.com/2013/11/07/how-a-lifetime-of-heart-disease-birthed-hyper-light-drifter/ [Accessed 12 Dec. 2018].

[8] Unity. (2018). *Unity*. [online] Available at: https://unity3d.com/ [Accessed 13 Dec. 2018].

[9] GitHub. (2018). *Unity-Technologies/ScriptableRenderPipeline*. [online] Available at: https://github.com/Unity-Technologies/ScriptableRenderPipeline/wiki/High-Definition-Render-Pipeline-overview [Accessed 12 Dec. 2018].

[10] https://www.blender.org/ (2018). *Blender | Open Source 3D creation. Free to use for any purpose, forever* [online] Available at: https://www.blender.org/ [Accessed 13 Dec. 2018].

[11] Unity. (2018). *Asset Store Terms of Service and EULA - Unity*. [online] Available at: https://unity3d.com/legal/as_terms [Accessed 13 Dec. 2018].

[12] Simulation, S. (2018). *Schizophrenia Simulation by ChoppyPine*. [online] itch.io. Available at: https://choppypine.itch.io/schizophrenia-simulation [Accessed 13 Dec. 2018].

[13] Hellblade.com. (2018). *Hellblade | Ninja Theory*. [online] Available at: https://www.hellblade.com/ [Accessed 13 Dec. 2018].

[14] itch.io. (2018). *Download the latest indie games*. [online] Available at: https://itch.io/ [Accessed 13 Dec. 2018].

[15] itch.io. (2018). *itch.io Terms of Service*. [online] Available at: https://itch.io/docs/legal/terms#2-users-and-publishers [Accessed 13 Dec. 2018].

[16] Assetstore.unity.com. (2018). *Book Of The Dead: Environment - Asset Store*. [online] Available at: https://assetstore.unity.com/packages/essentials/tutorial-projects/book-of-the-dead-environment-121175 [Accessed 13 Dec. 2018].

Appendix C. Highlight Report 1

PRCO304: Highlight Report

Name: Liam West

Date: 07/02/2019

Review of work undertaken

Installations of 3rd party assets such as ProBuilder, and ProGrid are utilized for prototyping a scene where I can test various mechanics. The High Definition Render Pipeline is also used as this provides good quality lighting / shader graph support for later development.

I began work on the Rigidbody Controller for the player, however due to flaws in some physics, and the player not feeling in control, I changed my approach for a Character Controller instead. I added a smooth rotation towards the mouse, a jump mechanic, movement at different speeds, and a modular input script so I can adapt it for a key binding's menu later into development.

I also worked on player management; for example, getters and setters for health. There is also regeneration of health over time if the player has not taken damage recently.

Also, using elapsed time I have detected when to damage the player after falling, using time as a multiplier.

I have researched character rigs for prototyping such as the UMA 2 asset pack, however, will be using this in later development as it requires a lot of set-up.

A crouch mechanic will allow the players to duck under small confined spaces. To fix bugs withstanding up in confined spaces, I rely on a SphereCast to project upwards.

Alongside this document I have also outlined a narrative.

Plan of work for the next week

I have added the Bush Craft asset pack for prototyping weapons. Next week I will be taking care of Bow and Sword/Spear mechanics as well as animations. I want to spend a good amount of time on these mechanics, giving them enough basic visual feedback for prototyping, such as the pull-back of the bow.

I will be adjusting my roadmap, as modelling will not take up a large space in early development. In doing so, I have extended this week's sprint to 2 weeks to allow enough time to develop the player mechanics. If the sprint goes to plan, I shall be moving onto gathering all necessary 3rd party assets for level building and designing the layout of the map based on the narrative.

Date(s) of supervisory meeting(s) since last Highlight

Brief notes from supervisory meeting(s) since last Highlight

Appendix D. Highlight Report 2

PRCO304: Highlight Report

Name: Liam West

Date: 14/02/2019

Review of work undertaken

Installed assets:
Book of the Dead: Environment
Boneskin Settlement Pack
Mesh Effects

After some careful reconsideration of the purpose of my game, I have decided on a new approach to allow the player to take control over the anxiety / hallucinations / voices, and to switch to a less anxious world, or the same location with other dangers. This could result in a more thorough report, with a purpose to decrease anxiety, or give people that they can have control over it, while keeping story-driven gameplay.

The first attempt at this mechanic was by adding 2 relatively identical scenes, with 2 player models based on each. This would act as a way to differentiate between the psychosis and the ability to control it. For example; hallucinations of enemies would in-turn become less of a threat, and pathways may be unveiled. This feature also changed my mindset about the player having a number of components. I instead created a player manager to hold important scripts relevant to the player. To create the seamless transition for the player between the 2 cameras, I took the vector, relative from an empty object in the scene that was identically placed in the second 'scene', and then applied that vector to the other player prefab. This would position the player exactly in the position of the other, but in a different area. I also took the camera rotational transform and set it for camera 1 or 2 (depends which camera is toggled).

Even though this was a seamless solution, having 2 levels within 1 scene would cost way too much processing power. There was a second solution, that was more time efficient and still had layers of potential. At this point in time I am only using 1 player prefab, with 2 cameras that mask only appropriate layers (Normal and Psychosis). This means that objects that were visible before, are now no longer rendered. Although the colliders for these objects are still in use, and therefore I had to create a list of all objects with the relevant layers and toggle their colliders if and when they are toggled.

I have also adjusted smoothness values of the Boneskin asset pack, and some normal values that were not correct for the type of material.

Plan of work for the next week

In the upcoming weeks, further attention will be given to level design, implementing the tribal village and forest landscapes. These landscapes will go hand-in-hand with puzzle

designs but keeping the puzzles relevant to the main narrative, so the player does not lose interest. I also would like to make some progress on the introduction of the Final Report, and gather references to features in my game, especially to explore different level design patterns for the upcoming sprint.
Date(s) of supervisory meeting(s) since last Highlight: 08/02/2019
Brief notes from supervisory meeting(s) since last Highlight A detailed highlight with good progress. In future include a view link to project management tool, any interim or working documents can be mentioned in the highlight and a view link provided, it is not necessary to upload them separately to the weekly highlight. In future provide direct links to 3rd party tools, addons, services used - pay attention to usage/licensing constraints and discuss in the main report accordingly.

Appendix E. Highlight Report 3

PRCO304: Highlight Report

Name: Liam West

Date: 21/02/2019

Review of work undertaken

Risk assessment and measures taken;

- Deadlines -
  At the beginning of this week's sprint there were priorities of a report due on the 18th for AINT308. To combat this risk, I will attempt to catchup during the coming weeks by pushing for significant progress in the level design.
- Internet Provider -
  Virgin Media has had an outage / problem throughout the entirety of Plymouth for the last few days, so progress on the level design was ground to a halt due to the inability to push changes / research effectively. I have taken initiative to talk to the provider, however they have not yet stated a date in which it will be fixed. I will not be able to traverse my work onto the University computers due to the large amount files required to transfer over, and the router problems preventing me from doing so. I will, however, attempt smaller sprints on a separate project folder at the University using a seperate branch / repository, so that once the service is running again, I will be able to merge, and not lose significant progress.

For what I can do, I have been researching psychosis, with the condensed and relevant points of the findings, bulletpointed in the miiscellaneous deliverables. This gives me a clearer understanding of the effects, and how I can incorporate those into my game.

Plan of work for the next week

Next week I shall be continuing with the sprint, trying to catchup on lost progress. It is untimely that these problems have occured during the same period.

I will also be working on further research papers about puzzle design in games, for a greater understanding on how I can incoporate those with the narrative.

Date(s) of supervisory meeting(s) since last Highlight: 14/02/2019

Brief notes from supervisory meeting(s) since last Highlight

Appropriate strategy in light of service outage - useful resource gathering and investigation (document added to miscellaneous) in light use full references - see Mendeley or LaTeX etcetera.

- Support the originality of the project with Psychosis/Mental-illness research.
- Think about the transitioning of normal
  self to a psychotic state. Slow transitioning?

• Editor panel- sliders or toggles the effect of the psychosis. Will work great for testing user experience and the project review.

Appendix F. Highlight Report 4

PRCO304: Highlight Report

Name: Liam West

Date: 27/02/2019

Review of work undertaken

Project Management: https://trello.com/b/V6J16IUo
GitHub: https://github.com/Plymouth-University/prco304-final-year-project-lwest2

- Custom editor inspector for blurring the lines between states of psychosis. Prodrome symptoms (Anxiety, lack of concentration, depression) and the Psychotic phase (Hallucinations, delusions.).
- So far implemented visuals for Anxiety, lack of concentration and depression through the use of post processing behaviours, lerping dependent on the slider in the inspector. This is purely to present the effects and will later be adjusted dynamically by events.
- Gathered further research on this symptoms, and how long they last for in relation to eachother. For example, the psychotic state often peaks and then falls slowly towards a normal state (In the long-term), however for this medium it will be represented in the short-term.
- HDRP procedural skybox instead of the HDRI skybox.
- Scene settings and (diffuse profile from The Book of the Dead environment pack.).

Plan of work for the next week

- I have a clearer idea of what I am pushing towards at this point. Different interactions will increase the severity of the prodrome symptoms, and after a certain threshold, the player will go into their psychotic state (Hallucinations, delusions.)
- Weather adjustments, symbolic of mood, and sound design will further increase the immersion of these symptoms.
- Visual adjustments for hallucinations and delusions need to be implemented.

Date(s) of supervisory meeting(s) since last Highlight: 22/02/2019

Brief notes from supervisory meeting(s) since last Highlight

Strong start, identifying core elements of the UX to deliver a credible representation/engagement with elements of psychosis

- Work on the states of psychosis (Example; insomnia, anxiety disorder, psychotic episode). Drafting out what happens in each (Script, visuals), for example for a psychotic episode the camera field of view may be hyper-focused, while if sleep deprived there may be other post processing or spot effects that may startle the player. After doing so, pull these states of psychosis together into a playable experience.

	• Possible interactions of environment (Lack of sleep, food eaten) or events (location) that trigger the state to change. • Blur between each of these states, not just a toggle between each of them. • Use LaTeX for bibliography / referencing.

Appendix G. Highlight Report 5

PRCO304: Highlight Report

Name: Liam West

Date: 07/03/2019

Review of work undertaken

Project Management: https://trello.com/b/V6J16IUo
GitHub: https://github.com/Plymouth-University/prco304-final-year-project-lwest2

- Attempted to make a 3-way 'health-bar' (Radar chart), which will display the sanity, hunger and thirst of the player, however there was difficulties with displaying this in the UI so a simpler approach may be necessary to begin with. Hunger and thirst increase over time, insanity will be equal to the average of these two.
- Made progress on tribe village scene. Still needs foliage.

Plan of work for the next week

- Finish the sanity mechanic. (Post processing and UI affected by the sanity of the player.)
- Finish the tribe village scene.

Date(s) of supervisory meeting(s) since last Highlight: 28/02/2019

Brief notes from supervisory meeting(s) since last Highlight

attempted complex UI solution, reverting to non-UI based psychosis effects as discussed

- Flesh out what the player will do.
- Interactions/paths.
- Object Orientated structure? Survive the night?
- User stories for objects.

Appendix H. Highlight Report 6

PRCO304: Highlight Report

Name: Liam West

Date: 14/03/2019

Review of work undertaken

Project Management: https://trello.com/b/V6J16IUo
GitHub: https://github.com/Plymouth-University/prco304-final-year-project-lwest2

- Village:
  - o Tribe village exit, check if ready to exit village, if not then displays a message to the player.
  - o Modular coroutine for displaying messages with a stylized transition. This will be useful for all dialogue in the game, and will act as subtitles, along-side audio voice-overs.
  - o Foliage in the scene and backdrops.
  - o Started Finite State Machine. Currently has many 2 states, wander and idle, and switches between these states on random time intervals. Baked NavMesh and included basic agent.
  - o Testing Occlusion maps and light maps.
  - o See graph is misc. deliverables of FSM.

Plan of work for the next week

- Create converse states, whereby the nav agent will talk in groups of other NavMesh for a period if in close perimeter of them. All states of FSM should be implemented.
- Look towards player when in close contact. (Could also act as a paranoia state, where all the NPC's look towards the player.)
- Interactions and Dialogue with NPC's.

Date(s) of supervisory meeting(s) since last Highlight: 07/03/2019

Brief notes from supervisory meeting(s) since last Highlight

Good progress as discussed

- Draft structure.
- Write up drafts for sprints.
- Work on proposed idea. (continue)

Appendix I. Highlight Report 7

PRCO304: Highlight Report

Name: Liam West

Date: 21/03/2019

Review of work undertaken

Project Management: https://trello.com/b/V6J16IUo
GitHub: https://github.com/Plymouth-University/prco304-final-year-project-lwest2
Video link: https://www.youtube.com/watch?v=s1h9ScrGwn0

- Added volumetric fog instead of exponential fog. This really sustained a better atmosphere with god rays/light rays.
- Tested with Sam Lord's weather system however it is not supported with the HDRP as far as I can tell. Tested with various other weather systems however have resorted to particle effects from Unity instead (Such as rain), which follows the player slowly instead of spreading over the whole scene which could cause performance issues.
- Decreased far clipping plane as I try to find the balance between the aesthetics and performance.
- Draft structure added to misc.

Plan of work for the next week

- Fade in / out rain solution. (Wetness of terrain through smoothness values, the rain itself, and the atmosphere of the skybox)
- Use VFX graph to create fire pit. (This will react to the rain and keep the players sanity from depleting.)
- Add various rocks, uneven terrain, water puddles, ponds to the village scene.

Date(s) of supervisory meeting(s) since last Highlight: 14/03/2019

Brief notes from supervisory meeting(s) since last Highlight

Positive progress, the environment is far more convincing, a useful investigation into weather system to improve the environment immersion

- Continue with project. (Environment is affected by weather events; sanity is affected by the environment interactions.)
- Make the tribe village grimier and more realistic of nature.
- Test with Sam Lord's weather system.
- Add draft structure to 'miscellaneous deliverables' on SPMS
- Add a draft chapter or section to 'miscellaneous deliverables' on SPMS for review/guidance

Appendix J. Highlight Report 8

PRCO304: Highlight Report

Name: Liam West

Date: 28/03/2019

Review of work undertaken

Project Management: https://trello.com/b/V6J16IUo
GitHub: https://github.com/Plymouth-University/prco304-final-year-project-lwest2
Video link: https://www.youtube.com/watch?v=9B2a9bVoP0E

- Rain particle effect by Unity.
- Rains at random intervals, and lerps terrain textures to look wet.
- Checks for objects within radius of the rain and changes their smoothness values, accordingly, depending on what the Renderer specifies. Originally did not use the radius, but the performance needed optimizing, and this approach is sufficed.
- Changed atmospheric parameters, and volumetric fog distance while it rains.
- Added more rocks and foliage, updated environment.

Plan of work for the next week

- Still need to make firepit for the player to keep alight.
- Will need to add objects / interactions for those objects affected by the rain.

Date(s) of supervisory meeting(s) since last Highlight: 21/03/2019

Brief notes from supervisory meeting(s) since last Highlight

- Pragmatic update with logical progression.

Appendix K. Miscellaneous Deliverables

Appendix K.1. Psychosis Research 01

Psychosis

(Healthline, 2019) [1] [2]

Difficulty Concentrating	• Lack of short-term memory. • Difficulty sitting still. • Difficulty thinking clearly. • Frequently losing things. • Inability to make decisions. • Inability to perform complicated tasks. • Lack of focus. • Careless mistakes.
Depressed mood	
Sleeping too much or not enough	
Suspiciousness	
Withdrawal	
Delusions	A false belief or impression that is firmly held even though contradicted by reality. (Paranoia etc.) Might think they are being followed, or that secret messages are being sent. There are other types of delusions such as somatic and grandiose.
Hallucinations	Sensory perception. Seeing, hearing, feeling or smelling something that is not real.
Switching topics erratically	
Depression	
Suicidal thoughts	

(Taylor, 2008) [3]

Hallucinations –

- May command harm to self or others.

Delusions –

- Feeling that one’s mind is dominated by forces beyond one’s control.
- Feeling of outside thoughts that are not their own.
- Feeling that there are people that wish to do one harm.

(Birchwood et al., 2007) [4]

Social Anxiety Disorder in schizophrenia –

Triggered by anticipation of catastrophic loss of social status. Shame of psychosis causes psychotic symptoms and depression. Socially marginalizes them.

Video Case Studies

(YouTube, 2019) [5]

Causes:

- Stress.
- Drugs/Alcohol.
- Risk of medications (Steroids)
- Mental illness (Mood disorder/schizophrenia)
- Brain damage.

(YouTube, 2019) [6]

- We all have the illusion of reality.
- An active imagination/impressions/perceptions. "Back project to the sensations that we do have".

References

[1] Healthline. (2019). *Psychosis: Symptoms, Causes, and Risk Factors*. [online] Available at: https://www.healthline.com/health/psychosis#symptoms [Accessed 21 Apr. 2019].

[2] Healthline.com. (2019). *Unable to concentrate: Causes, Symptoms and Diagnosis*. [online] Available at: https://www.healthline.com/symptom/unable-to-concentrate [Accessed 21 Apr. 2019].

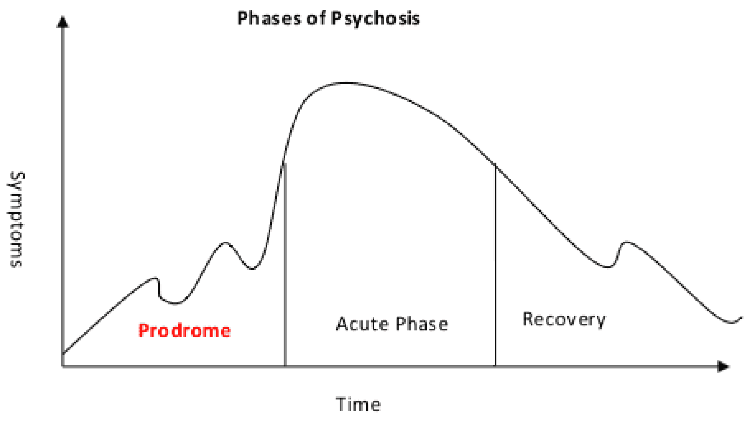
[3] Taylor, P. (2008). Psychosis and Violence: Stories, Fears, and Reality. *The Canadian Journal of Psychiatry*, 53(10), pp.647-659.

[4] Birchwood, M., Trower, P., Brunet, K., Gilbert, P., Iqbal, Z. and Jackson, C. (2007). Social anxiety and the shame of psychosis: A study in first episode psychosis. *Behaviour Research and Therapy*, 45(5), pp.1025-1037.

[5] YouTube. (2019). *Psychosis - causes, symptoms, and treatment explained*. [online] Available at: https://www.youtube.com/watch?v=UK813F8ZoPg [Accessed 21 Apr. 2019].

[6] YouTube. (2019). *Psychosis: Bending Reality to See Around the Corners | Paul Fletcher | TEDxCambridgeUniversity*. [online] Available at: https://www.youtube.com/watch?v=tV2RLLtOgL4 [Accessed 21 Apr. 2019].

Appendix K.2. Psychosis Research 02



(Earlypsychosis.ca, 2019) [1]

Quick rise towards phase. Fluctuates a bit. Recovery phase is slower.

First phase.

1. Disconnected.
2. Alone.
3. Suspicious.
4. Brighter or sounds become louder.
5. Difficulty on focusing on what they are hearing.
6. Difficulty screening out distracting information and sensation.
7. Difficult to keep track of things.

Second phase.

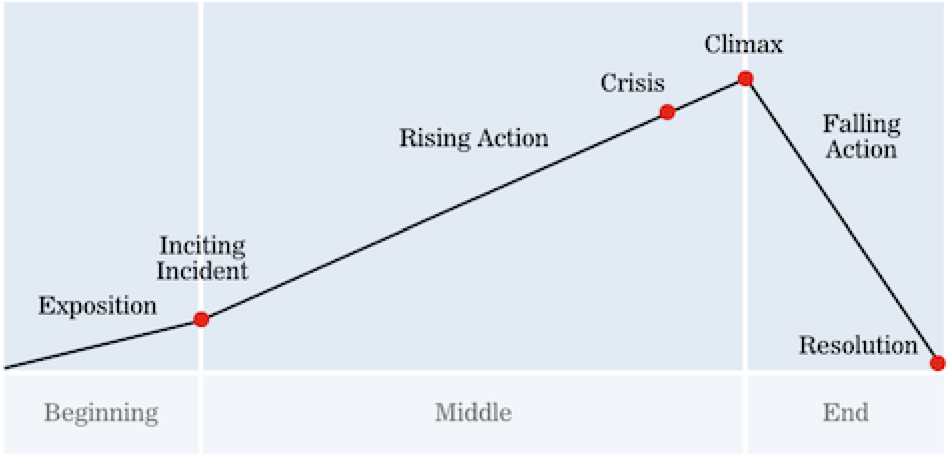
1. Hallucinations.
2. Delusions.
3. Very odd speech behaviours.

References

[1] Earlypsychosis.ca. (2019). *What is Psychosis? Phases of Psychosis - earlypsychosis.ca*. [online] Available at: https://www.earlypsychosis.ca/pages/curious/phases-of-psychosis [Accessed 21 Apr. 2019].

Appendix K.3. Narrative Plan

High Level Narrative Summary



(Kiransureshblog.wordpress.com, 2019) [1]

Major Locations

[Flash back]

- Tribe village – Exposition
  - o Your character looks around and sees a small village nestled in the welcoming arms of a mountain range.
  - o Smell of smoke and general chatter brings you back to your thoughts.
  - o Your eyes scanning the forest that is just out of reach from where your temporary settlement lays; secluded and safe.
  - o You decide to explore your home, wandering throughout the small area and come upon a place to trade, weapons, and huts of various shapes and sizes.
  - o Able to speak to some of the villagers, however pain overcomes you and your vision warps slightly causing you to grip your head in confusion and pain.
  - o You head back to your home where your mother awaits, greeting her, before laying down some furs and falling asleep.
- Inside hut – Inciting Incident
  - o You wake up, the hut afire and puddled in blood. You scream, shouting for help. However, when the others arrive everything is fine, your mind was hallucinating.

[Continues after flash back]

- Around a camp fire.
  - Facing your mother, and the shaman, who says;
    - Your heart is haunted, and your mind is in a dark place. You are being exiled from your home but only until you defeat your demons.
    - You are handed a spear and a bow, as you walk out towards the forest. The sun setting as the forest darkens, its mouth stretching open in an ominous manner as the shadows consume you.
- Middle area.
  - Here you can choose one of three directions. Each direction leads you on a short journey to defeat a certain type of demon.
  - Each direction is equipped to a different element.
- Path 1 – Darkness/Loneliness
  - Puzzles.
  - Boss.
- Path 2 – Winter/Cold
  - Puzzles.
  - Boss.
- Path 3 – Fire/Evil
  - Puzzles.
  - Boss.
- Tribe village – Resolution.
  - Arrives back at the village in welcoming arms.
  - The sun glimmers in your eyes as it rises.
  - Fades to black

Characters

Nzir –

- Your character.
- Young male.
- Naïve/scared.

Shaman –

- Elder male.

- Position of knowledge and all knowing.

Nzir’s mother –

- Middle-aged female.
- Loving/thoughtful/worried.

Villagers (Small interactions dialogue) –

- Man making a speer tip.
- Woman cooking food.

Boss 1 –

- Darkness boss.

Boss 2 –

- Ice boss.

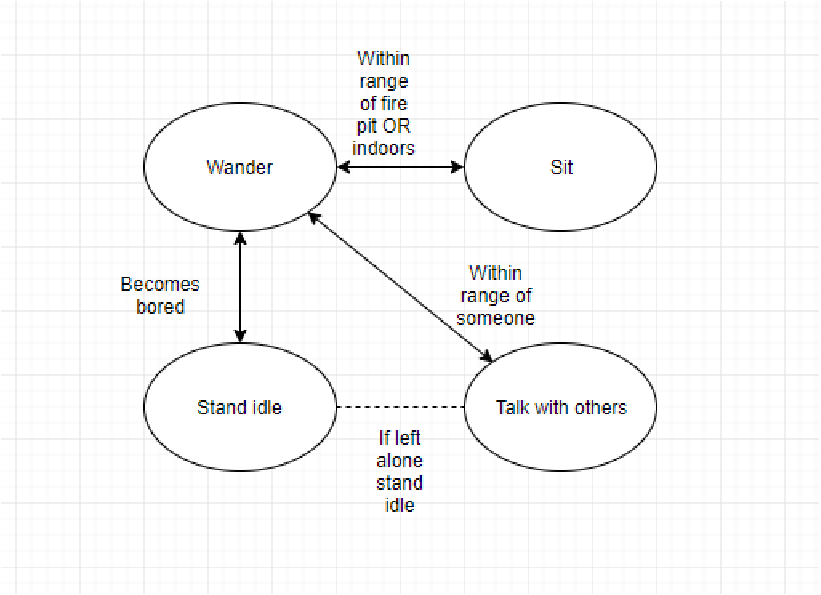
Boss 3 –

- Fire boss.

References

[1] Kiransureshblog.wordpress.com. (2019). *narrative arc – A2 Media Coursework*. [online] Available at: https://kiransureshblog.wordpress.com/tag/narrative-arc/ [Accessed 21 Apr. 2019].

Appendix K.4. Finite State Machine Diagram



Logic defining the Finite State Machine of neutral AI, able to sit, wander, stand ide and talk with others depending on a number of variables.

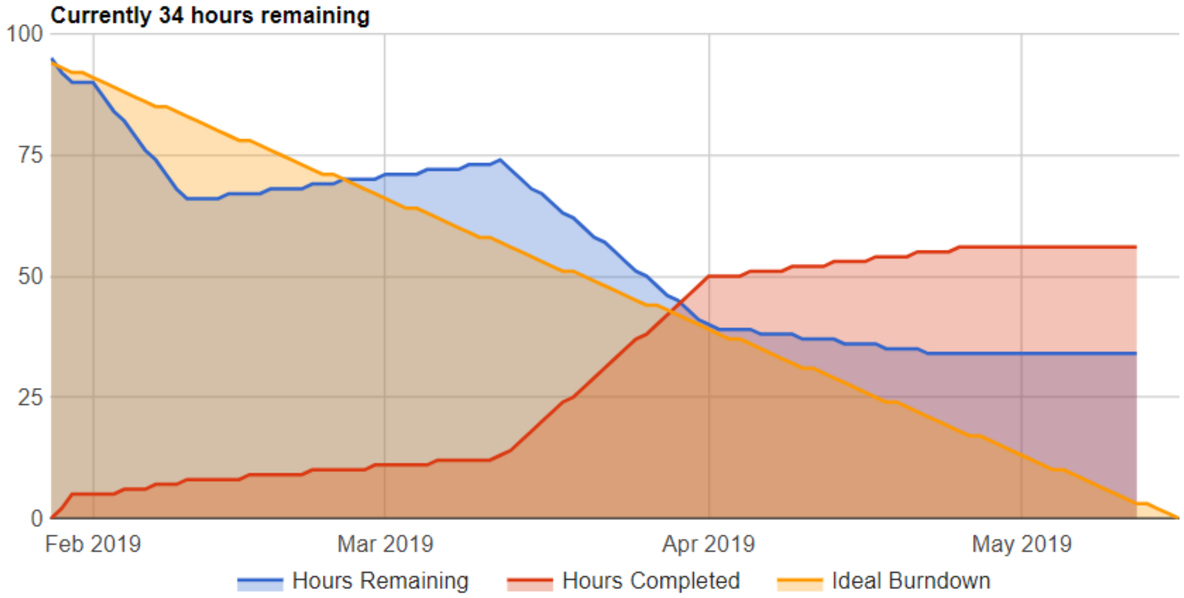
Appendix L. Project Management Artefacts

Appendix L.1. Trello Board

https://trello.com/b/V6J16IUo/astray

Appendix L.2. Burndown Chart

https://BurndownForTrello.com/share/x3el333nwe



Appendix M. Unity's Book of the Dead: Environment License

This documentation excludes licenses relating to software, or assets that were not used in the project 'Astray'.

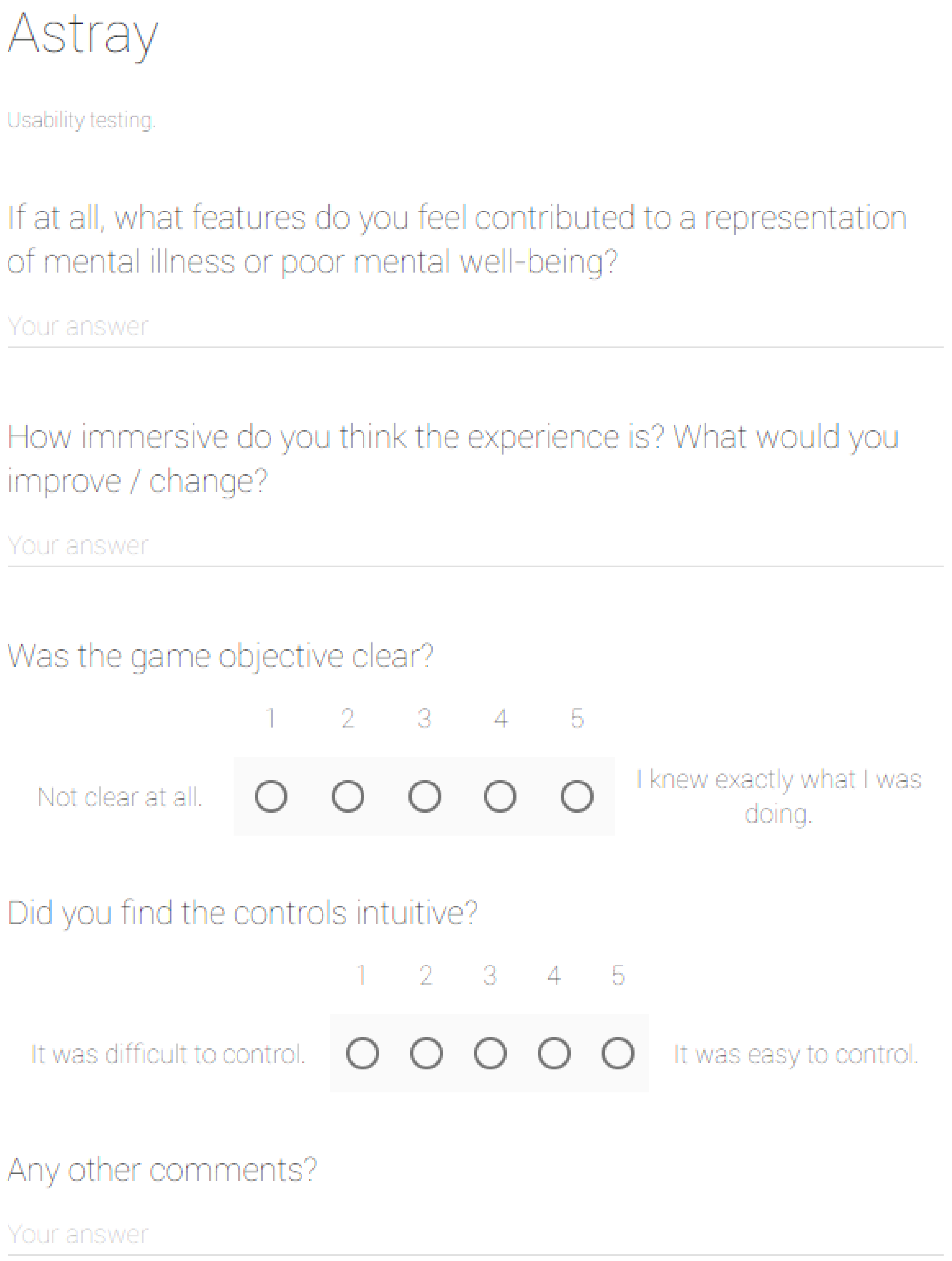
Appendix M.1. Book of the Dead: Environment

The Book of the Dead: Environment project and all its art assets are governed by the standard Unity Asset Store EULA; however, the following components are governed by custom licenses.

Appendix M.2. Post Processing, Shader Graph, Render Pipeline Core, High Definition Render Pipeline, ProBuilder

Appendix N. Usability Testing

Appendix N.1. Questionnaire



Astray

Usability testing.

If at all, what features do you feel contributed to a representation of mental illness or poor mental well-being?

Your answer

How immersive do you think the experience is? What would you improve / change?

Your answer

Was the game objective clear?

	1	2	3	4	5	
Not clear at all.	○	○	○	○	○	I knew exactly what I was doing.

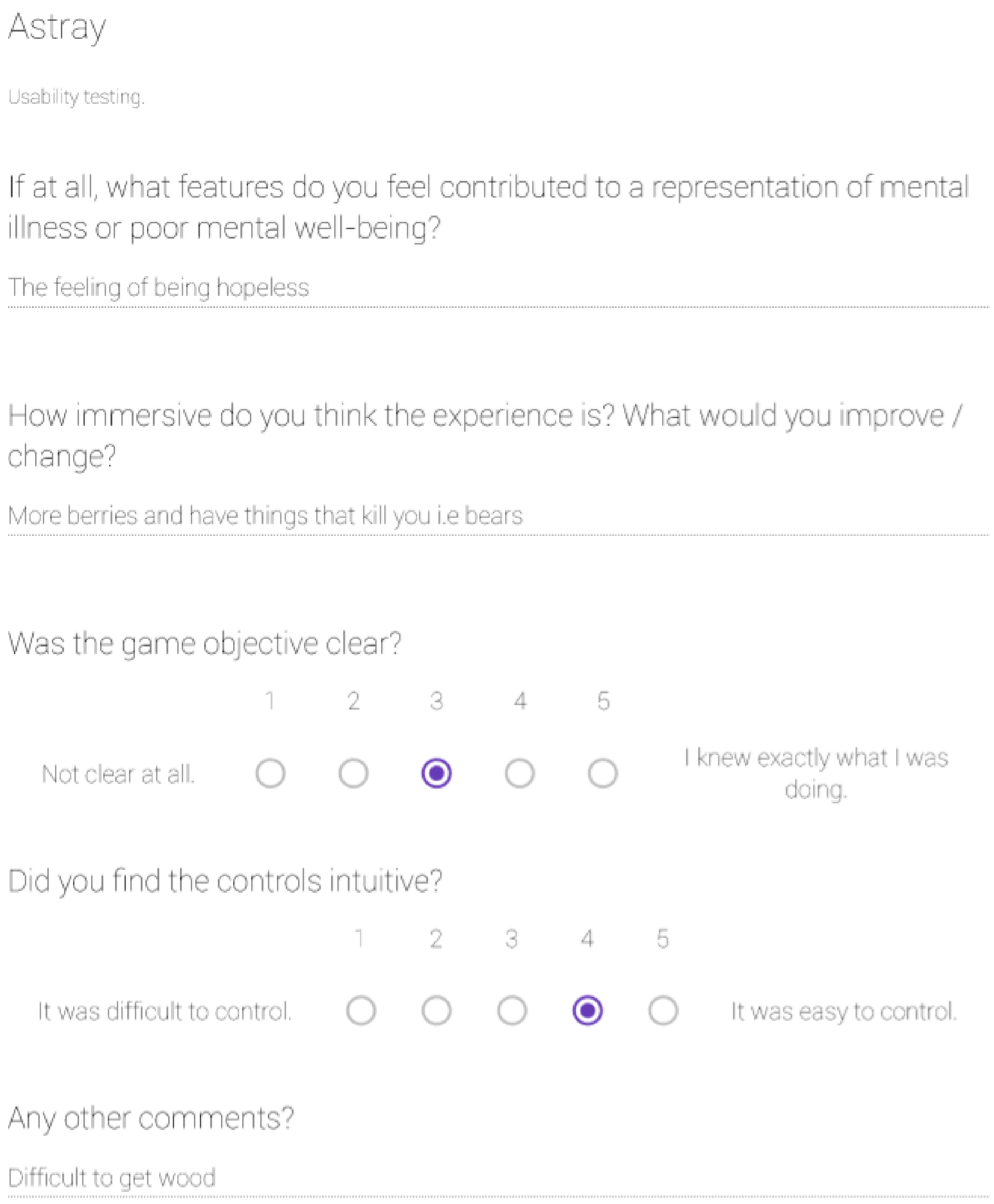
Did you find the controls intuitive?

	1	2	3	4	5	
It was difficult to control.	○	○	○	○	○	It was easy to control.

Any other comments?

Your answer

Appendix N.2. Questionnaire Responses



Astray

Usability testing.

If at all, what features do you feel contributed to a representation of mental illness or poor mental well-being?

The feeling of being hopeless

How immersive do you think the experience is? What would you improve / change?

More berries and have things that kill you i.e bears

Was the game objective clear?

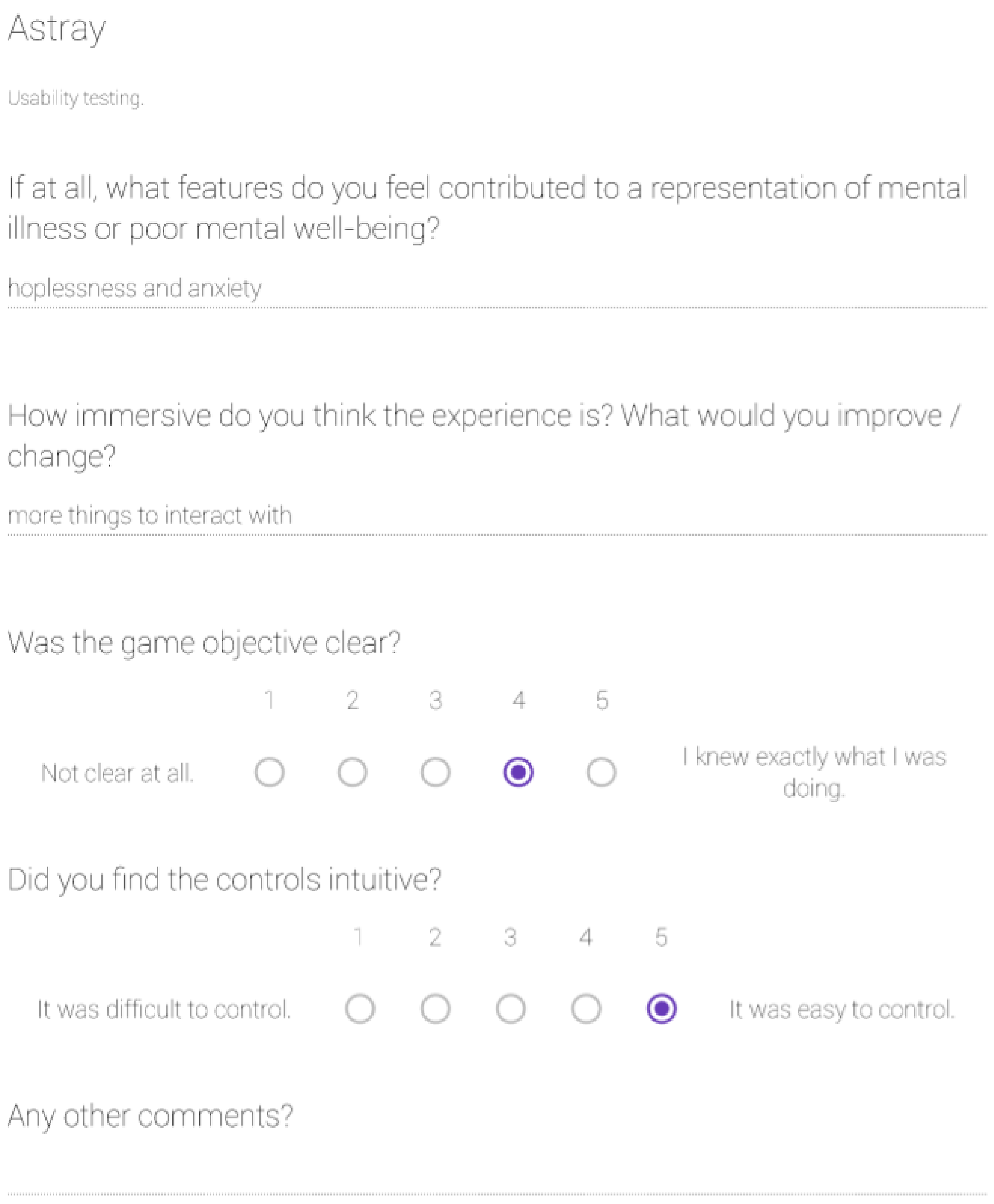
	1	2	3	4	5	
Not clear at all.	○	○	●	○	○	I knew exactly what I was doing.

Did you find the controls intuitive?

	1	2	3	4	5	
It was difficult to control.	○	○	○	●	○	It was easy to control.

Any other comments?

Difficult to get wood



Astray

Usability testing.

If at all, what features do you feel contributed to a representation of mental illness or poor mental well-being?

hoplessness and anxiety

How immersive do you think the experience is? What would you improve / change?

more things to interact with

Was the game objective clear?

	1	2	3	4	5	
Not clear at all.	○	○	○	●	○	I knew exactly what I was doing.

Did you find the controls intuitive?

	1	2	3	4	5	
It was difficult to control.	○	○	○	○	●	It was easy to control.

Any other comments?

Appendix N.3. Information Sheet

FACULTY OF SCIENCE AND ENGINEERING

RESEARCH INFORMATION SHEET

Name of Principal Investigator

Liam West

Title of Research

Astray – Usability Testing

Aim of research

Research into how games can represent mental wellbeing and psychosis through various mediums; including visuals and auditory sources. The research will also gatherer necessary feedback on the mechanics of the game, and subjective views on the approach towards this topic.

Description of procedure

Approximately 15 minutes per individual. Firstly, the participant will play the game while being asked various questions in response to their actions. Secondly, the participants will be asked to fill out a questionnaire

Description of risks

Possible sensitive topics of discussion involving mental health.

Benefits of proposed research

Knowledgeable insight into how games can represent an individual's mental state through visuals and sound. Necessary usability experience feedback to improve the games mechanics.

Right to withdraw

You have the right to withdraw at any moment before, during or after the research. Any data that you have submitted may be removed at any time upon request. You will remain anonymous as part of this research.

If you are dissatisfied with the way the research is conducted, please contact the principal investigator in the first instance: telephone number *01621 855 814.* If you feel the problem has not been resolved, please contact the secretary to the Faculty of Science and Engineering Research Ethics & Integrity Committee: Mrs Paula Simson 01752 584503.

Appendix N.4. Consent Forms

FACULTY OF SCIENCE AND ENGINEERING

Human Ethics Committee Sample Consent Form

CONSENT TO PARICIPATE IN RESEARCH PROJECT / PRACTICAL STUDY

___

Name of Principal Investigator

Liam West

___

Title of Research

Astray – Usability Testing

___

Brief statement of purpose of work

Research into how games can represent mental wellbeing and psychosis through various mediums; including visuals and auditory sources. The research will also gatherer necessary feedback on the mechanics of the game, and subjective views on the approach towards this topic.

___

The objectives of this research have been explained to me.

I understand that I am free to withdraw from the research at any stage and ask for my data to be destroyed if I wish.

I understand that my anonymity is guaranteed, unless I expressly state otherwise.

I understand that the Principal Investigator of this work will have attempted, as far

as possible, to avoid any risks, and that safety and health risks will have been

separately assessed by appropriate authorities (e.g. under COSHH regulations)

Under these circumstances, I agree to participate in the research.

Name: Aleef Naseem

Signature: ANaseem Date: 15/05/2019

FACULTY OF SCIENCE AND ENGINEERING

Human Ethics Committee Sample Consent Form

CONSENT TO PARICIPATE IN RESEARCH PROJECT / PRACTICAL STUDY

Name of Principal Investigator

Liam West

Title of Research

Astray – Usability Testing

Brief statement of purpose of work

Research into how games can represent mental wellbeing and psychosis through various mediums; including visuals and auditory sources. The research will also gatherer necessary feedback on the mechanics of the game, and subjective views on the approach towards this topic.

The objectives of this research have been explained to me.

I understand that I am free to withdraw from the research at any stage and ask for my data to be destroyed if I wish.

I understand that my anonymity is guaranteed, unless I expressly state otherwise.

I understand that the Principal Investigator of this work will have attempted, as far

as possible, to avoid any risks, and that safety and health risks will have been

separately assessed by appropriate authorities (e.g. under COSHH regulations)

Under these circumstances, I agree to participate in the research.

Name: Harry Garfield

Signature: HGarfield Date: 15/05/2019

Appendix O. GNU GENERAL PUBLIC LICENSE V3.0

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too. When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps:

(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.