

PR6001 - Professional Studio Project

'Project UMBRAE'

Production Log

James Marland - 23100010

Table of contents:

Introduction	Page 01
Week 01: Initial Idea (24/11)	Page 02
Week 02: Planning and Concepts (01/12)	Page 05
Week 03: Presentation (08/12)	Page 08
Week 04: Wrapping up (15/12)	Page 11
Holiday: Organisation (22/12)	Page 15
Holiday: Tools (29/12)	Page 19
Reading: Production Prep (05/01)	Page 21
Week 05: Production Start (12/01)	Page 26
Week 06: Planning Content (19/01)	Page 30
Week 07: Creating Content (26/01)	Page 35
Week 08: Catchup and Cleanup (02/02)	Page 39
Week 09: Pivoting development (09/02)	Page 43
Week 10: Player Revamp (16/02)	Page 47
Reading: Break (23/02)	Page 50
Week 11: Refinements (02/03)	Page 50
Week 12: External Profiling (09/03)	Page 52
Week 13: Cutscenes and Showcase (16/03)	Page 55
Week 14: Immersion (23/03)	Page 57
Holiday: Development (30/03)	Page 61
Holiday: EPC2026 (06/04)	Page 61
Week 15: Event Overhaul (13/04)	Page 61
Week 16: Polish (20/04)	Page 64
Week 17: Final Week (27/04)	Page 65
Conclusion	Page 68

Introduction

Over the course of my final group project module 6001, I worked with 14 other students to develop a game called Project UMBRAE. Each week of production is covered in the following chapters, which present my thoughts and contributions to the project, along with the resources used. I also reflect on the week and discuss the problems faced.

WEEK 1: Initial Idea

Objective:

The objectives for week 1 are to first meet and organise with the team, get to know all the teammates and what they want to make, generate ideas of what the game could be and expand the original pitch, then work towards consolidating them into one core idea that everyone agrees on. Alongside, and especially once the idea is locked in, working on the pre-production deliverables. I also plan to work on an asset list tool for in-engine use during pre-production.

Research:

This week's resources used/researched:

Sites:

- **Lewandowski, M.** (2012). *Floor Plan Creator*.

Articles/Documentation:

- **Epic Games** (2025). *Level Streaming Volumes Reference in Unreal Engine | Unreal Engine 5.7 Documentation | Epic Developer Community*.
- **Muir, M.** (2025). *Technical Guide to Linear Content Creation: Pipeline Development | Course*.

Games:

- **CAPCOM Co., Ltd.** (2021). *Resident Evil Village*.
- **CAPCOM Co., Ltd.** (2017). *Resident Evil 7 Biohazard*.
- **SadSquare Studio** (2020). *Visage*.
- **Tarsier Studios** (2017). *Little Nightmares*.
- **Kinetic Games** (2020). *Phasmophobia*.

Videos/Talks:

- **Argyriou, A.** (2025). *Lighting in UE5: Scaling for Quality & Performance | Unreal Fest Bali 2025*.
- **Condon, J.** (2025). *Technical Pre-Production Fundamentals for Unreal Engine Projects | Unreal Fest Bali 2025*.
- **Georg, L., Huffa, J. and Aikman, A.** (2024). *Exploring Technical Art in Game Development with Riot Games*.
- **Matt Oztalay** (2025a). *A Tech Artist's Guide to Automated Performance Testing | Unreal Fest Bali 2025*.
- **Oztalay, M.** (2025b). *Optimising UE5: Advanced Rendering, Graphics Performance, and Memory Management | Unreal Fest 2024*.
- **Reuter, A.** (2025). *Technical Artist's Guide to Profiling in Unreal Engine | Unreal Fest Orlando 2025*.

Research Analysis:

Through my research, I learned the context and considerations needed for proper profiling, problem-solving, and tool development. While I don't need to set up full-automation profiling, I will use the lessons learned to build and set up scripts that produce data that can be referenced across the project, making it clear when bottlenecks are forming and what is causing them. Running this in a weekly scrum format also helps make sure problems are caught early and handled quickly. I also refreshed my game design and art direction experience in horror by researching games I had played and by looking at new titles with a vision very similar to the brief.

Weekly Summary:

Organising the team

Over the course of this week, I made a lot of progress. I was elected producer of my team's project because I had prior production experience and had created the original brief. I was able to explain the original vision to the rest of my team, immediately clarify what they had envisioned, and work collectively to generate ideas and build something the entire team was happy to make, not just mine. Many early steps toward the game's direction and scale were established. While similar in concept to the brief, this was achieved by experimenting with different ideas and ultimately reinforcing certain aspects of it.

Involving the team

I also worked on getting an idea of what everyone could do and wanted to make over the project, so they could work on their role and specialism as much as possible and be involved in the game. While I created the art bibles' first template, I trusted the team to contribute to the preprod sections, so the GDD, Art bible, and Asset list are the work of others.

Name	Discipline	Job Title (Self described)	Wanted role(s) in project	Skills (Add relevant skills from the drop down)
Jamie Marland	Tech Art	Producer / Tech Artist	Producer / Tech Artist	Tools Optimisation VFX Shader Set Dressing Lighting Games design Blueprints Programming Unreal Blender Perforce Houdini Photoshop-like
Kanye Wallace	Games Design	Puzzle Designer	Puzzle Designer	Games design UI/UX Blueprints Level Design Puzzle Design AI VFX Unreal
Joel Oybo	Games Design	Gameplay / Level Designer	Gameplay / Level Designer	Games design UI/UX Blueprints Level Design Puzzle Design Unreal
Amy Aules-Ortiz	Character Creation		Character Artist	
Charlotte Knight	Game Art			
Sonamah (Sam) Malik	Game Art	Prop / Environment Artist	Prop/Environment/Narrative Design	Hard Surface Modelling Set Dressing Maya Photoshop-like Substance Painter
Sam Tiplady	Game Art	Prop / 3D Artist		Hard Surface Modelling Texturing UV 2D Drawing Maya Substance Painter
Adam Honey	Game Art	Environment Artist		Hard Surface Modelling Sculpt (Organic) Modelling Set Dressing Lighting Texturing UV VFX Photoshop-like ZBrush
Daniel Curley	Game Art	Environment Artist		Hard Surface Modelling Set Dressing Lighting Texturing UV Blueprints Level Design Maya Unreal ZBrush Substance Painter Photoshop-like Perforce
Matthew Bellamy	Game Art	Lighting / VFX Artist	Lighting/VFX Artist, Set Dressing	VFX Set Dressing Lighting Unreal Maya
Max Ebdon	Game Art	CG Generalist		Hard Surface Modelling Set Dressing Texturing UV Lighting Games design 3D Animation Maya Blender
Shanti Hamilton Sandhu	Animation	3D Animator	Player Animator	Hard Surface Modelling Sculpt (Organic) Modelling Texturing 3D Animation Maya ZBrush
Lauren Chan	Animation	3D Animator	Creature Animator	Hard Surface Modelling Sculpt (Organic) Modelling Texturing 2D Drawing UI/UX 3D Animation 2D Animation Maya Houdini ZBrush Substance Painter
Summer Rimmer-Riley	Animation			
Danielus Silek	Animation	3D Animator	Events Animator	Sculpt (Organic) Modelling Texturing 2D Drawing UV 3D Animation Maya Houdini

This is a self described skillset list for people to fill in, those that put them originally on discord have been carried over.
Remember more skills does not equate to better. Add what you feel are your strengths.
This should hopefully allow people to know who they can talk to about related issues.
This will also start to serve as guidance for the roles people will play in production.
Think I'm missing a relevant skills? msg me and ill add it - Jamie

Skill set list for people to know what everyone else can do, self-assigned by them.

Project setup, style guide and standards

I went about setting up the Unreal project itself with the correct project settings and outliner/content browser/sublevel structure and organisation, as well as explaining this in person with my team and having a written version so that we get people on the same page for assets standardisation, which is a crucial aspect to maintain with so many members and time. I also needed to fix a problem with the Perforce P4V Ignore setup.

★ Project Structure Guide

Important Perforce Warnings:

- Perforce does not commit empty folders; hence, you will see a fair number of placeholder textures to ensure they are committed. Use them if necessary.
- Perforce does not like deleting/renaming folders that have been committed to the main try, so avoid doing so at all costs, as it can create ghost folders and files. Make sure all folders are properly made and named before committing them.
- If you move/rename a folder, right-click the parent folder and select Update Redirector References. Unreal may keep the older folder for old references if they're committed, which can lead to bad asset references.

Content Browser Structure:

Assets = Concept Approach

Assets should be stored by 'concept', so if we had a sink asset, we would first consider the sink's use. It is an asset that appears in the backroom, so we would put this in 'Content/UMBRAE/Environment/Bathroom/Sink'.

This concept contains anything related to it, for the sink, this might be the Mesh, PBR Textures and materials. If it uses more, they also get added here. The exception is for content used across multiple concepts. If we had a texture used by multiple concepts, we would place it at the highest level. So, if a texture is used by both bathroom and kitchen assets, we would put it just inside the environment folder. If, however, it is intended for generic/common/global use, we should put it in the _Global folder under a related subsection.

★ Asset Guide - Style & Targets

All Artists should aim to respect these guidelines set out by the technical artist (Jamie).

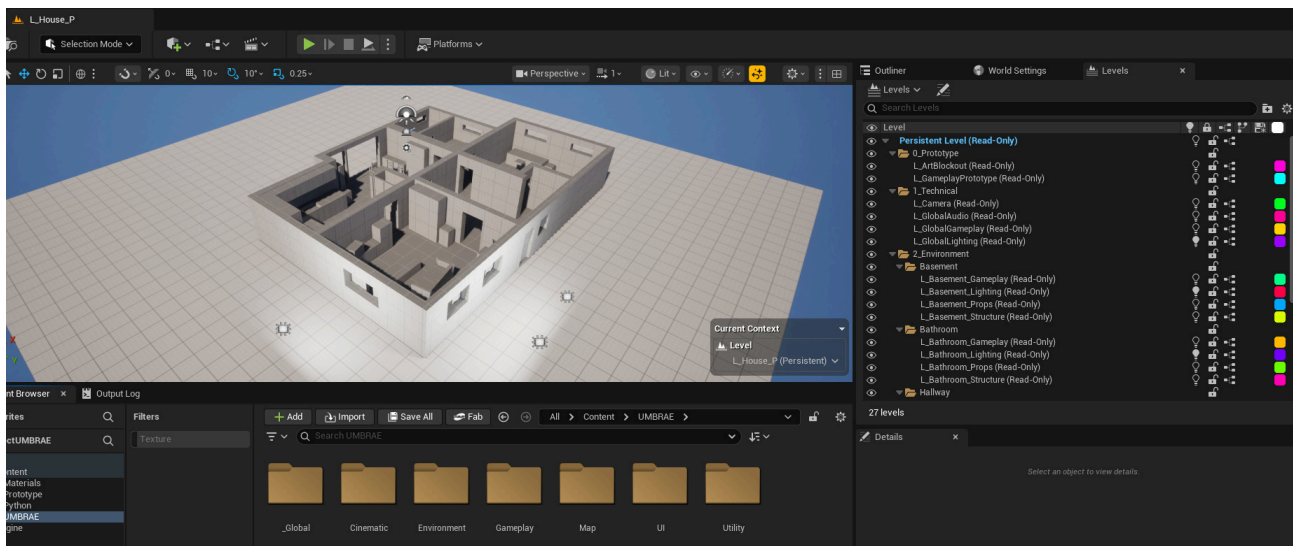
General:

- All assets must follow the naming convention set out by the UES Style Guide: <https://github.com/Allar/ue5-style-guide>
- Naming Convention: (Prefix){AssetName}{Variant}_{Suffix} For example: 'T_WoodPlank_01.D' for a wood plank's diffuse is a variant. Assets without variants would omit the _{Variant}.
- Meshes must be Nanite-enabled. (Minus translucent meshes, which will be handled separately)
- All textures should be a power of 2, e.g., 128, 256, 512, 1024, 2048, 4096.
- Textures that should be made for an asset: Diffuse (D), Normal (N), Occlusion Roughness Metallic (ORM) Packed PBR
- PBR Values should aim to follow real-world values: <https://physicallybased.info/>
- All materials, unless necessary, should inherit the master material.
- One Material slot per mesh unless a real use is required.
- The player is 1.8m tall, and the Asset must be scaled accordingly. (Find real-world measurements)
- Baked details from high poly to low poly workflow.
- All Assets should be in their retrospective folders: [Folder Structure Guide]

Small Prop:

- Props Size < Player Size
- <= 1k Texture Size
- <= 3k Poly count

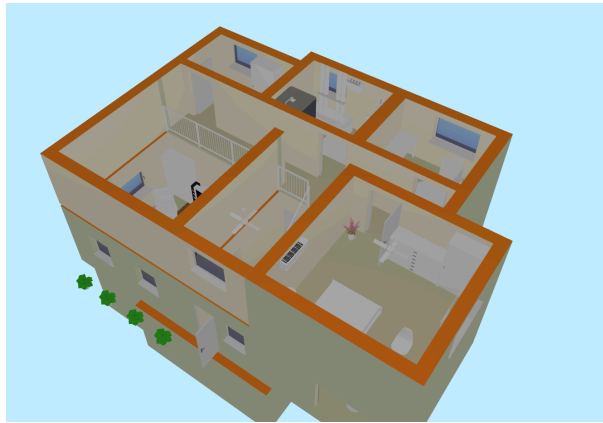
Project structure guide and asset guide tech documentation.



In-engine showcase of sublevels and content browser organisation.

Blockout

Based on Adam's parents' old house, which fit the environment we wanted to create and Kanye's level design plan, I made an early 3D layout in a house-building tool, which Charlotte later used as the first draft of the actual model.



3D Layout plan model made with Floor Plan Creator.

Weekly Reflection:

Reflection

This week, I made a strong start on getting people organised and involved in the project, and I feel I incorporated many of their ideas to improve the game and make it a product of everyone. I have also focused on gathering as much documentation as possible and on pre-prod setup, so I can shift to tool

development in preparation for production and to proof-of-concept testing for some of the more technical ideas planned. I think I'm slightly ahead of schedule.

Problems

I didn't really face any major problems. However, I realised I have limited access to the engine. I wanted to link to the on-import event to execute a renamer, but C++ isn't supported in a blueprint project, and this behaviour isn't exposed in Python or blueprints.

WEEK 2: Planning and Concepts

Objective:

The objectives for week 2 were to flesh out and plan what we wanted to make. This included the story as well as working on proof-of-concept mechanics, which, in my case as a technical artist, will involve implementing seamless portals and establishing project guidelines for folder structure and asset standards. As well as keeping on top of general pre-production deliverables.

Research:

This week's resources used/researched:

Articles:

- **marcauby** (2024). *UE5 & Perforce Best Practices | Community Tutorial*.
- **Muir, M.** (2024). *Technical Guide to Linear Content Creation: Pipeline Development | Course*.

Videos/Talks:

- **Cavanagh, S. and Michaelides, M.** (2025). *Integrated Insights: Automating Performance Capture in Unreal Projects | Unreal Fest Orlando 2025*.
- **Cloward, B.** (2025). *Game Optimisation - Introduction & General Principles - Episode 1*.
- **Dev Squared** (2023). *Seamless Portals made in Unreal Engine 5*.
- **Dev Squared** (2024). *How to Create Third Person Portals | Unreal Engine 5*.
- **Hoey, J.** (2025). *The State of the Art: What ArtStation Data Tells Us about the Industry | Unreal Fest Stockholm 2025*.
- **Tong, G.** (2025). *LEGO® Horizon Adventures: Bringing Terrain to Switch | Unreal Fest Stockholm 2025*.

Research Analysis:

This week, my primary research focused on the portal, as the team aimed to replicate non-Euclidean spaces. After finding Dev Squared videos, I was able to understand exactly what a seamless portal setup looks like in UE5, primarily through camera reprojection and render targets. I also looked into some general tool development and more automated performance tests.

Weekly Summary:

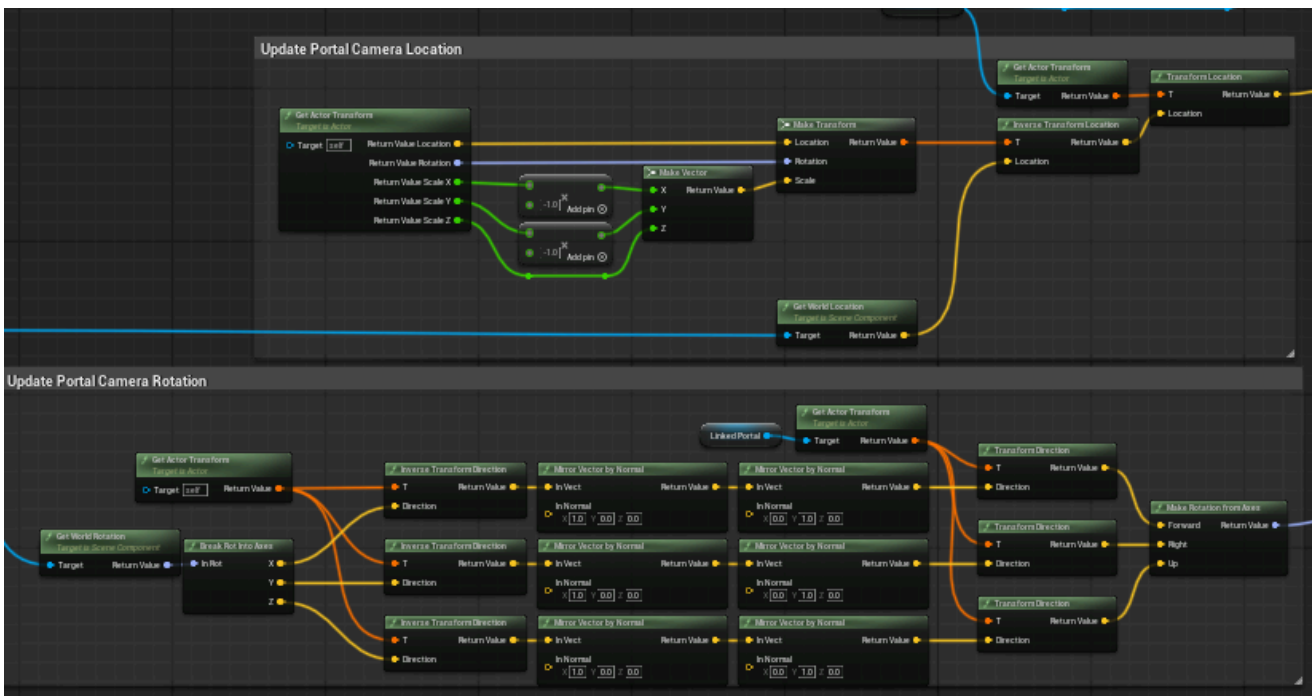
Producer

Throughout the week, I stayed on top of producer roles by organising the SCRUM Weekly and Daily meetings and made the team aware of their responsibilities and the week-long goal. The online sessions are recorded, and Weekly SCRUM Meetings are summarised into posts as well.

Proof of concepting the portal

Over this week, my focus was mainly on setting up an MVP of the portal to show the team how it would look and work. It is not the most performant, but that was not the focus at this point.

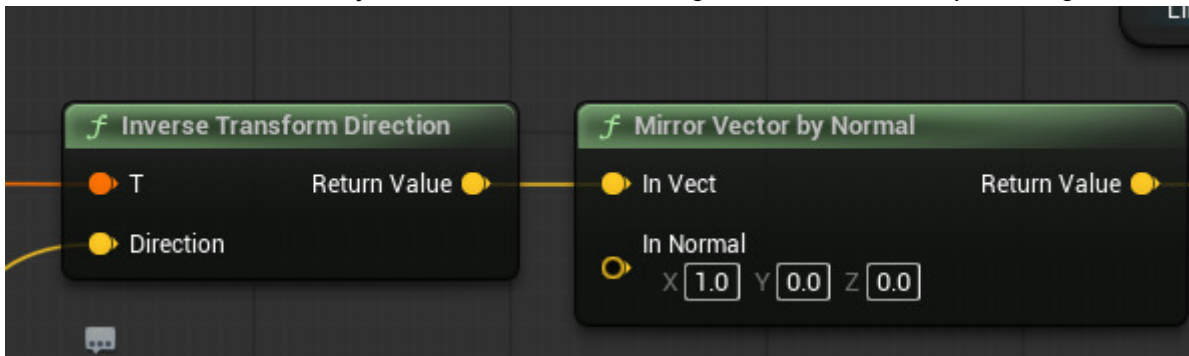
The portal works as follows: it first creates a dynamic resolution render target that captures from another portal's linked scene capture, then sets the plane's mesh to that target. If the player is detected nearby, it will set the scene capture to reflect the player's camera's position and rotation relative to the other portal's location, mimicking their movement.



The portal's scene capture reflection logic.

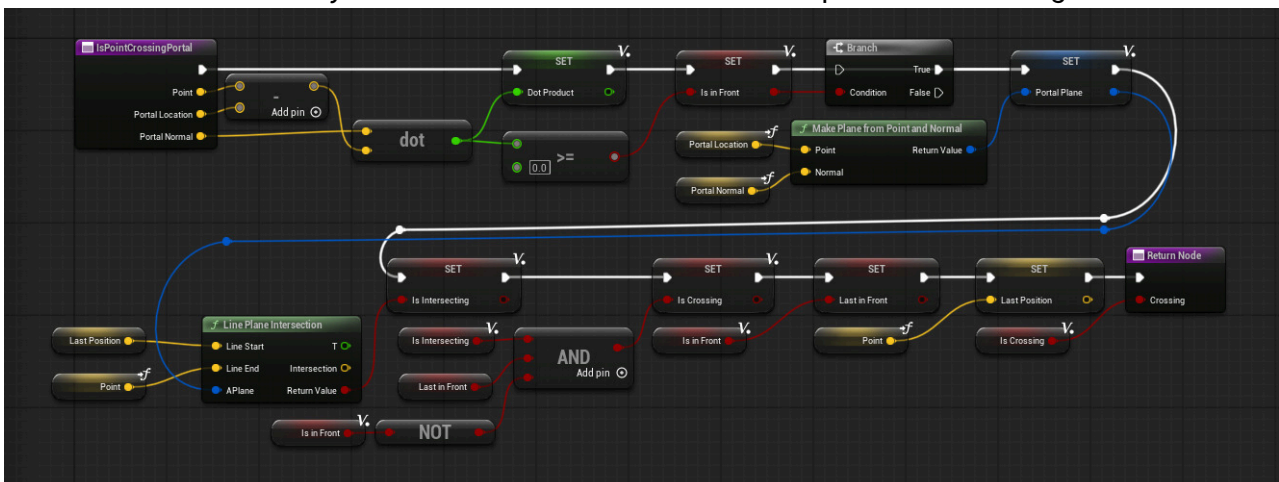
If the player enters the teleport collision, we check if they overlap the plane and then if they intersect exactly the plane (So we don't teleport them backwards and forwards or have an awkward middle zone between teleports)

Teleporting the player is actually very similar, as we again reflect the player's location, the camera's location and rotation, relative to the portal. I also store their current velocity so I can set it again after teleporting.



Inverse Transform and Mirror Vector functions are the backbone of this entire system.

This does require temporarily disabling the FPS movement damping, so it happens instantly, but it wasn't too much of a hassle. I did face some problems with point-crossing detection, but I was able to fix them, as the line-plane intersection would always fail because it was set up in the wrong direction.



Correctly set up the point crossing portal with the line plane intersection.

Bottlenecking

Upon entering class on Thursday, the project experienced significant FPS drops and high GPU usage. For most of the day, I worked on figuring out what this problem was and why it happened. The PCs in that room, in particular, were designed for multi-threading. Still, the Unreal editor uses only one of these threads, which was heavily bottlenecking the entire GPU dispatch pipeline, making it appear as if the GPU was heavily used. In other classrooms, especially on high-performance single-core-focused CPUs, this problem disappears. Talking with Gregor and Niccolo about this, I hope a solution can be found. My knowledge on this topic was mainly through Cloward, B. (2025), which I had previously watched before production.

GPU 0 Graphics Queue 0 Timing [STATGROUP_GPU0_Graphics0]			
Counters	Avg	Busy Max	Min
Queue Total	88.71	92.87	84.69
VolumetricCloud	24.41	30.83	15.86
RenderDeferredLighting	19.67	22.89	16.54
Postprocessing	14.24	14.66	13.78
TemporalSuperResolution	13.49	13.92	13.04
Shadow Depths	9.11	10.49	7.33
Lights	7.68	10.02	5.37
LumenScreenProbeGather	7.50	8.41	6.77
Shadow Projection	3.99	5.29	3.35

GPU stats showed processes well above their real costs.



The high draw time indicates CPU bottlenecking.

Weekly Reflection:

Reflection

This week, I did okay. I got bogged down a lot with producer responsibilities and wasn't able to dedicate as much time as I would have liked to my technical art role, but I still made decent progress. Next week, I plan to finish up my roles in pre-production and push the team towards the presentation in good shape. I'm on schedule.

Problems

This week was marked by various hitches that slowed development, as discussed above regarding optimisation, portal setup, and being spread thin helping others.

WEEK 3: Presentation

Objective:

The objective for week 3 was straightforward: to prepare for the end-of-week presentation to our Industry Mentor, Karl Wooley, who is the Global Head of Immersive at Framestore.

Research:

This week's resources used/researched:

Forums/Blogs:

- **Apgar426** (2022). *Improving Virtual Shadow Map Distance/Quality*.
- **kiokko89** (2024). *Lumen and Light Mobility*.
- **NicoNiiChan** (2019). *What's this X Mark on the light?*
- **noswahj** (2014). *Shadows Disappear with Increased Camera Distance*.

Articles/Documentation:

- **Epic Games** (2025a). *Lumen Global Illumination and Reflections in Unreal Engine*.
- **Epic Games** (2025b). *Set Tracing Max Distance | Unreal Engine 5.7 Documentation | Epic Developer Community*.

Games:

- **Fireproof Games** (2014). *The Room*.

Video/Talks:

- **Reid, C.** (2016). *Practical Law 101 for Indie Developers: Not Scary Edition*.
- **Guerrette, K.** (2018). *Moving the Heavens: an Artistic and Technical Look at the Skies of The Last of Us*.
- **Virtual Production Insider** (2024). *Tips for Optimising Your Lighting in Unreal Engine*.

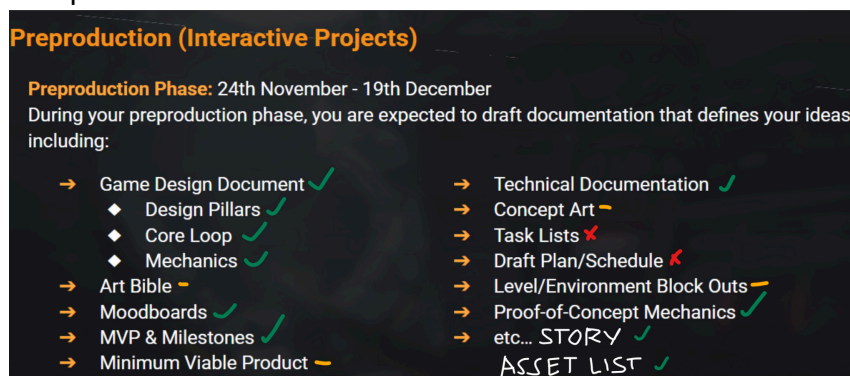
Research Analysis:

Through my research this week, I primarily focused on solving case-by-case, project-specific problems as we set up the project and researched topic areas, both related and more general TA resources.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and made the team aware of their responsibilities and the week-long goal. The online sessions are recorded, and Weekly SCRUM Meetings are summarised into posts as well.



A quick progress list I made for the weekly scrum post.

The rest of the post focused on breaking down those areas that still needed work. For most of the team, this would be the art bible, blockout or proof of concepts. As the producer, I also wanted to focus on ticking off the task list and the project schedule, and until I could practice for my presentation, the schedule and helping artists and game designers set up the blockout would be my focus.

Schedule

I worked on outlining the product's development stages and how it should look, along with milestone targets and key dates. This was created in the YouTrack sprints, along with an infographic added to the knowledge base. Curious about the recent online discussion of Nano Banana Pro's ability to create infographics, I decided to give it a try. It failed miserably and couldn't contain any information or be visually appealing, so I resorted to just generating an HTML file and taking a screenshot. Once it was made, I would fact-check and proofread it to ensure what I presented was correct. So while I credit Gemini for the formatting and presentation, the rest was created by me.

Project Timeline					
★ = Milestone					
WK	DATE RANGE	TYPE	STAGE	PRODUCT VERSION	IMPORTANT DATES
01	24 Nov – 1 Dec	Uni	Preproduction	Planning	
02	2 Dec – 8 Dec	Uni	Preproduction	MVP	
03	9 Dec – 15 Dec	Uni	Preproduction	MVP	Mentors: Fri 12 Dec
04	16 Dec – 22 Dec	Uni	Preproduction	MVP ★	
05	23 Dec – 29 Dec	Break	Production	Alpha	
06	30 Dec – 5 Jan	Break	Production	Alpha	
07	6 Jan – 12 Jan	Reading Week	Production	Alpha	
08	13 Jan – 19 Jan	Uni	Production	Alpha	
09	20 Jan – 26 Jan	Uni	Production	Alpha	
10	27 Jan – 2 Feb	Uni	Production	Alpha ★	
11	3 Feb – 9 Feb	Uni	Production	Beta	
12	10 Feb – 16 Feb	Uni	Production	Beta	
13	17 Feb – 23 Feb	Uni	Production	Beta	
14	24 Feb – 2 Mar	Reading Week	Production	Beta ★	
15	3 Mar – 9 Mar	Uni	Production	Gamma	Mentors: Fri 6 Mar
16	10 Mar – 16 Mar	Uni	Production	Gamma	
17	17 Mar – 23 Mar	Uni	Production	Gamma	
18	24 Mar – 30 Mar	Uni	Production	Gamma	
19	31 Mar – 7 Apr	Uni	Production	Gamma ★	
20	7 Apr – 13 Apr	Break	Postproduction	Gold	
21	14 Apr – 20 Apr	Break	Postproduction	Gold	
22	21 Apr – 27 Apr	Uni	Postproduction	Gold	
23	28 Apr – 4 May	Uni	Postproduction	Gold	
24	5 May – 11 May	Uni	Postproduction	Gold ★	Project Sub: Thu 7 May
25	12 May – 18 May	Uni	Postproduction	Retrospective	
26	19 May – 22 May	Uni	Postproduction	Retrospective ★	Retro Sub: Fri 22 May

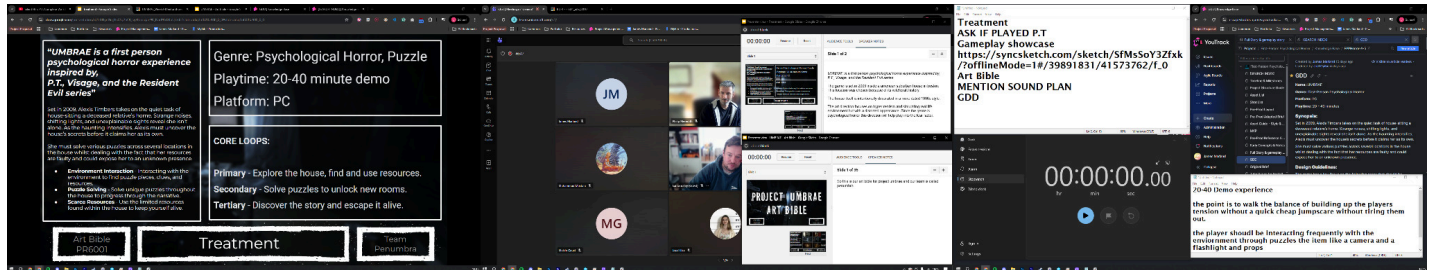
Project Timeline, HTML formatting code generated by (Google, 2025).

Hopefully, it lines up the team well with knowing where we should be without me guiding. Each milestone will be defined later, along with precisely what we want made at that point in development and the following priorities.

Presentation Time

Preparing for the presentation took a fair amount of time. I created a gameplay showcase video and polished rough areas of the blockout for the showcase, focusing on gameplay, props, lighting, and collision.

I also had to make sure I knew what to say and what to show. I outlined the overall presentation plan and key points to highlight, and, in less clear sections, wrote speaker notes for what I would say on the slides. I also rehearsed and timed this to get the flow right. Below is how my desktop would look during the presentation.

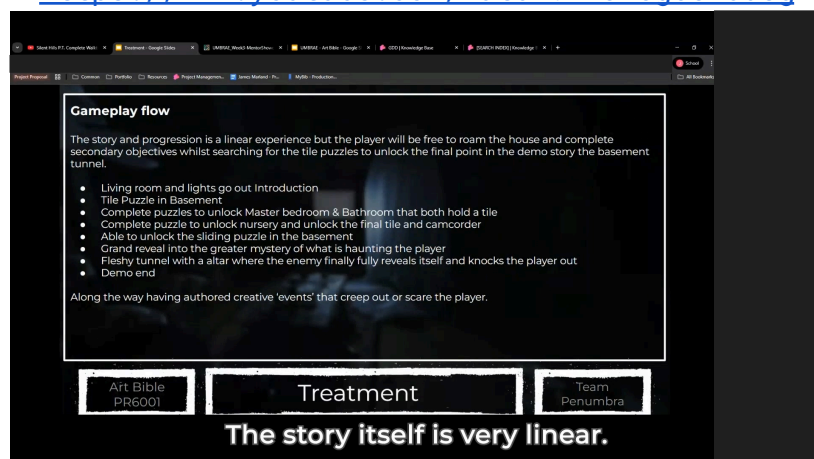


Desktop during presentation.

The presentation was well received, and both Karl and the tutors liked it. Gathering feedback later in the day and turning it into action plans next week would be part of week 4.

Over the weekend, I also created a supercut version for team members to rewatch easily. The presentation is available online here:

<https://www.youtube.com/watch?v=3N0gc8WUdsg>



Project UMBRAE - PREPROD - Mentor Showcase (Marland, 2025).

Weekly Reflection:

Reflection

This week, I did pretty well. Again, I wasn't able to do much in the TA role, which was understandable given that this week's focus was on my being heavily involved as a producer. I was proud of the presentation I did, and I hope the team felt I did them well.

Next week, I plan to integrate the feedback into the team's weekly goals and into my TA duties by working on the event system. I feel I'm on schedule.

Problems

Luckily, this week I didn't face any issues.

WEEK 4: Wrapping up

Objective:

The objective for Week 4 was to integrate the feedback we received from the presentation into our plans and ensure the team understood what we would be working on and how we would proceed after the holidays.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Shields, B.** (2025). *Optimising Performance in Unreal.*
- **Świerad, O.** (2025). *Tech Art Aid - Unreal Engine Graphics Tutorials.*

Articles/Documentation:

- **Epic Games** (2024). *MegaLights in Unreal Engine | Unreal Engine 5.5 Documentation | Epic Developer Community.*

Videos/Talks:

- **Albert, C.** (2025). *Creating Debug Tools with SlateIM | Unreal Fest Stockholm 2025.*
- **Barle, S.** and Khan, H. (2025). *The Future of Nanite Foliage | Unreal Fest Stockholm 2025.*
- **Ben Cloward** (2025). *Better Tile Breakup - Terrain Shaders - Episode 7.*
- **camwing** (2025). *How Puzzle Games Hack Your Brain.*
- **Drapeau, D.** (2025). *Why Most Game Artists Fail at Freelancing (And How to Fix It).*
- **Tyroller, J.** and Pillosu, R. (2025). *Writing Clean Code for Games (Industry Veteran).*

Research Analysis:

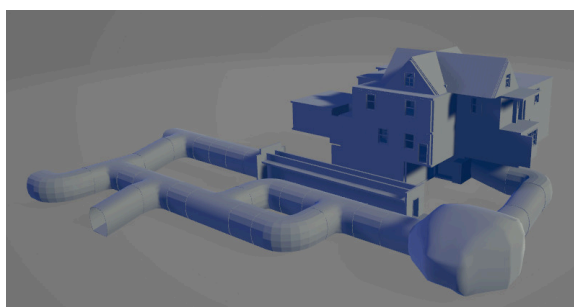
This week's research primarily consisted of a general-purpose TA study and performance specifications for the mirrors and portal implementations.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and made the team aware of their responsibilities and the week-long goal. The online sessions are recorded, and Weekly SCRUM Meetings are summarised into posts as well. This also involved clarifying the production milestones and weekly builds, performance tests, and feedback reviews.

I also helped Danielius set up Perforce to showcase the scene and, at the end, provided a complete model of the house prototype for all animators to use as references for their scenes.



Pre-production house blockout.

Over the week, I clarified with everyone that they were happy with what they would mainly be working on, that they would take responsibility for it throughout the project, and that they knew what the first thing they would be working on was.

Created by James Marland 9 days ago Updated by Lauren Chan 9 days ago Visible to article readers

★ Team Roles

This role does not mean you only have to do that thing, but it should reflect what you do the **majority** of the time during the project. And remember, it's more important that you actually mention what you're doing on your production log! If there are any issues, **please** let me know. - Jamie

THE CURRENT FOCUS AND WORKING OVER BREAK IS TEMPORARY FOR THE BREAK TO PREVENT CONFLICTING WORK

Name	Role	Current Focus	Working over break
Jamie	Producer / Tech	Events / Asset Validator	Yes
Kanye	Puzzle Design	Puzzle Mechanics	Yes
Joel	Gameplay	Item Mechanics	No
Amy	Character	Blockout Characters	Yes
Charlotte	Prop / Structure	Modular Structure Models / Living Room / Puzzles Props	Yes
Sami	Prop	Bathroom Props	No
Sam	Prop	Nursery / General Props	No
Adam	Prop	Basement / Tunnel / Interactive Props	No
Daniel	Prop / Structure / Material	Modular Structure Textures	Yes
Matty	Lighting / VFX	VFX	Yes
Max	Prop	Kitchen / Dining Room / Hallway	No
Shanti	Player Anim	Player Tests	Yes
Lauren	Creature Anim	Creature Tests	Yes
Summer	General Anim	Player Tests	No
Danielelius	Scene Anim / Tech	Scene Tests	Yes

Team roles and current focus.

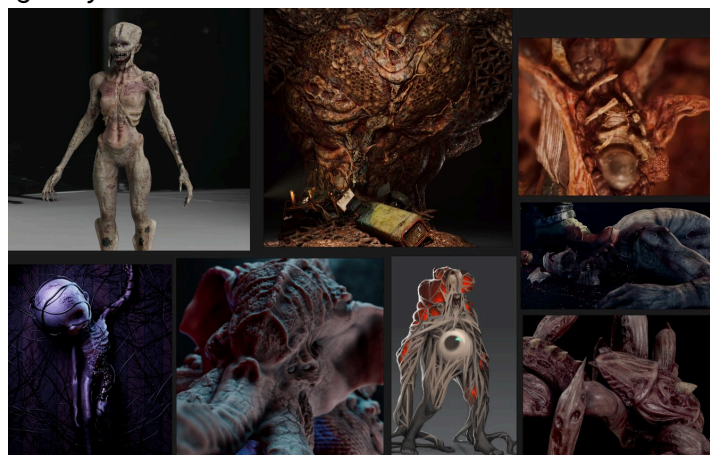
Feedback Changes

Based on the feedback from last week, we decided to remove the gameplay sanity component and move toward a stretch goal. This gives us more time to focus on the player's experience.

They also recommended having a definitive list of puzzles, specifying which rooms they would be in, the order in which they would appear, and how to solve them. Kanye would primarily handle this as the Puzzle Designer. The same was also said about the events; having some proof-of-concept examples to show would be helpful. The focus across the week was on creating these.

A suggestion brought up was that we could use photogrammetry. The team discussed the potential use cases for photogrammetry. We decided that, for large props, we wouldn't, as it would be challenging to create unique ones with photogrammetry. However, to help fill out the scene with small, low-priority props, especially organic ones, it could be an efficient way to speed up the workflow.

Another thing discussed afterwards was the level of gore the rookies were allowed. We have to be careful to stay within the rookie's guidelines. I looked at past rookie finalists to see what has been previously accepted, and our project plans to include a similar level of gore, or slightly less. We aim to stay away from showing any traumatic events.



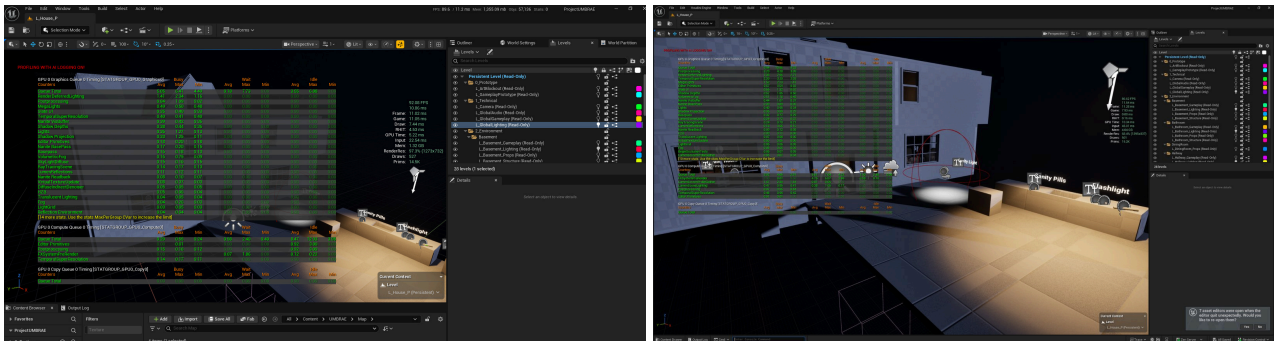
Mirror and Portal Implementation

On Thursday, I looked more closely into balancing high-fidelity mirrors with a portal that accurately shows the scene without significantly impacting the budget. I looked into several methods, including Screen Space Reflections, Lumen Cache, Planar, and Cubemap. The issue was that all of these were either high-cost or low-quality.

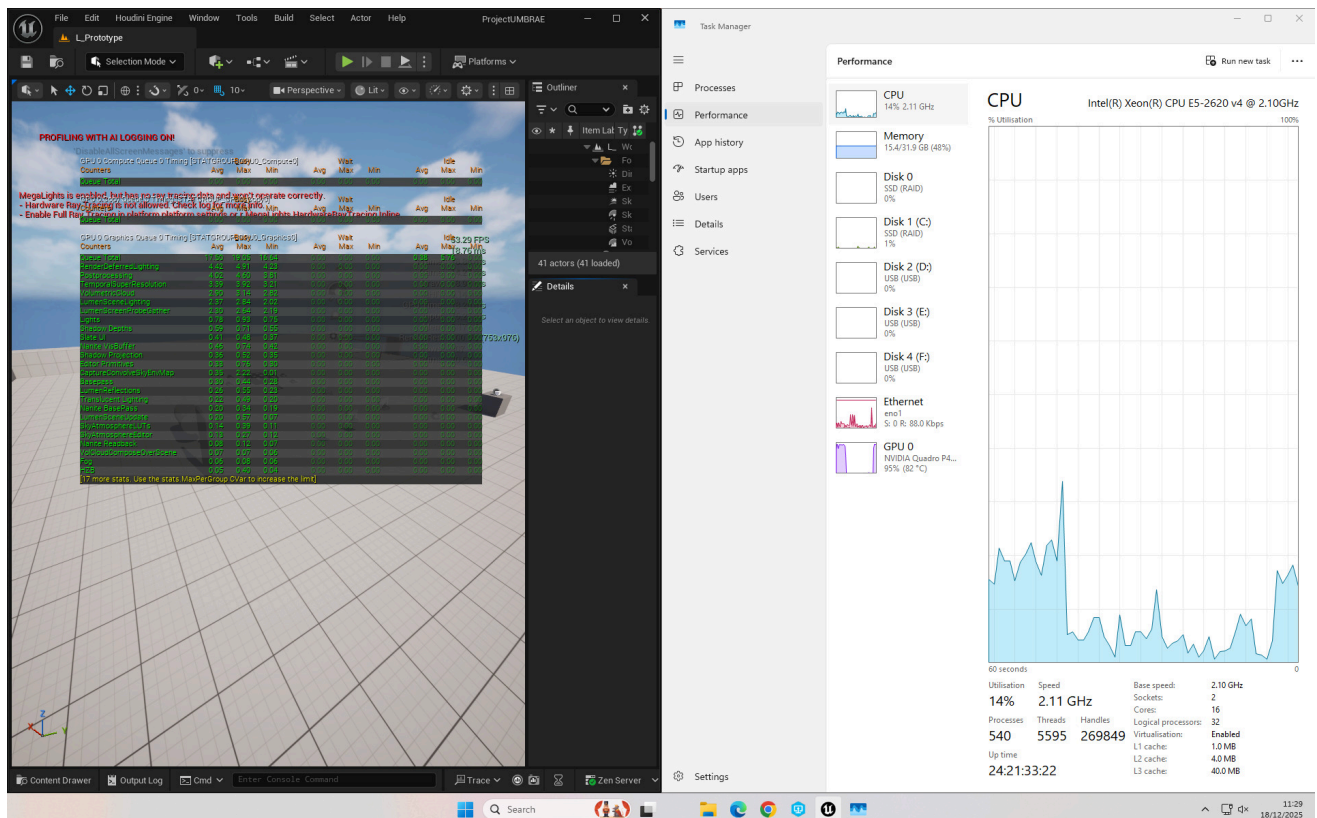
On Friday, I discussed these options with the team. We weighed the options and ultimately decided on a mix. Mirrors use planar and Screen Space Reflections to deliver somewhat performant but high-quality results. In contrast, Portals would use duplicate rooms, with doors serving as teleportation points. We would also use triggers that justify not using mirrors, such as breaking after the event that uses it.

Performance tests

On Thursday, Gregor asked me to gather the performance data to send over that explains the issues with that room. I also investigated the issue more.



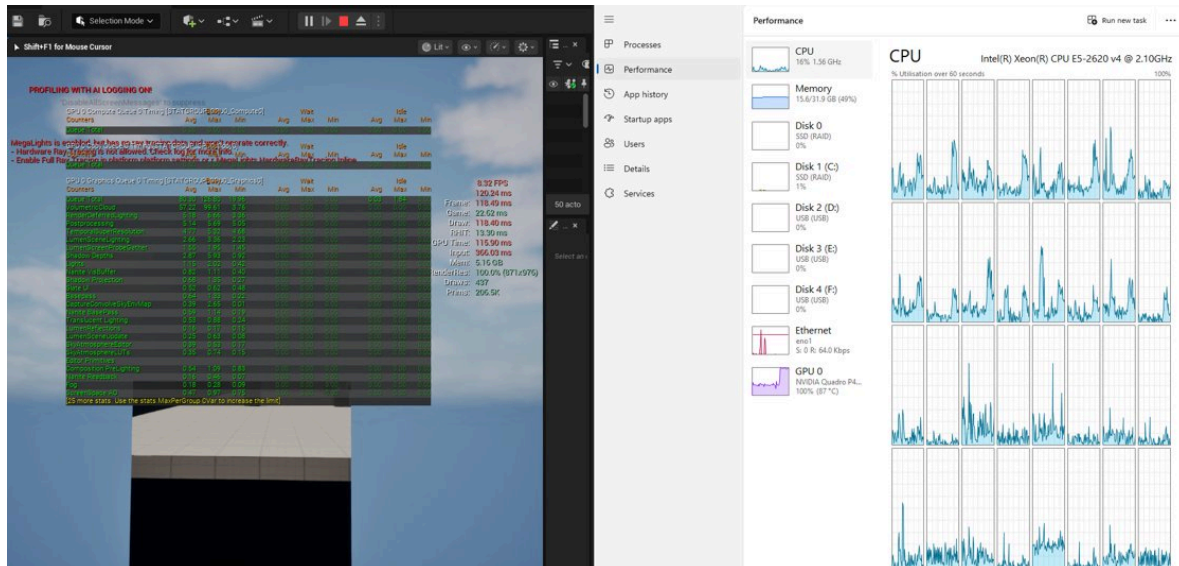
Left is my home PC, and Right is the Arrendale PC (in another room).



Winterfell (Thursday room) performance tests.

As you can see, the scenes are not costly at all, and they run fine in another room. There is a clear hardware bottleneck issue in Winterfell. The GPU also

doesn't support hardware ray tracing, which is an issue because our lighting artist, Matty, wants to use megalights.



Low CPU utilisation across threads and maxed-out GPU utilisation in a simple scene show the hardware is being dragged to its knees.

Event System

I also started on the event system, which would need to handle a variety of events we planned. To get this in motion, I set up the parent and child categories as blueprints and outlined them to the team.

James Marland 18/12 10:58

Event System

Currently have planned four types of events: (From Right to Left)

- **Auditory Hallucination:** When triggered or randomly, and when its conditions are met to activate, it will play its audio cue.
 - E.g., a sound of a baby crying when outside the nursery, a window tap randomly
- **Visual Hallucination:** When triggered or randomly, and meets its conditions to activate, it will play its visual effect for a set time.
 - E.g., objects looking unlit, Objects moving slightly, etc.
- **Environment Change:** When triggered, it will permanently change the scene visually.
 - E.g. After finishing the nursery arc, the room looks significantly worse from then on.
- **Creature Encounter:** When in trigger, it will play an encounter with the creature. This may involve creature movement, props changing, audio playing, and lights flickering. This also includes the potential for the player to 'fail' it and be reset.
 - E.g. Looking in a mirror, you see it over your shoulder looking back, it disappears, it twitches in the corner, and when the player approaches, it lashes at them quickly.

The event system is outlined on Teams.

Weekly Reflection:

Reflection

This week went well, but again was very producer-focused. This will hopefully change as people are self-guided and working through tasks in production, allowing me to take on my tech artist role more effectively. I plan to work over the holiday on several systems in preparation for production, including automated development screenshots, the asset validator, and project organisation, as well as fleshing out systems such as events.

Problems

Minus the performance problems, there was nothing of concern.

Holiday: Organisation

Objective:

The objective for the first week of the holiday was to work on aspects that would be harder to add while others are working, such as organising the project files and naming conventions to conform to the standard, setting up automatic development progress screenshots, and creating an asset validator plugin.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Mróz, J.** (2025). *This Will Rewire How You Think about Rendering optimization. This is a Deep dive.* - Blog.

Articles/Documentation:

- **Cane, M.** (2025). *Texel Density Checker Tool V2.0 | UE5.*
- **Lei, X.** (2022). *Unreal Rendering Workflow with Python Automation.*
- **Lindgren, J.** (2025). *How to Install, Secure, and Maintain a Perforce P4 Server on Windows | Community Tutorial.*
- **Nistor, A.** (2025). *Asset Status Overlay.*
- **Shinsoj, I.** (2025). *PCG Knowledge Base.* | Notion.
- **Thiessen, H.** (2025). *GitHub - HaukeThiessen/Mip-Explorer: a Tool to Display How Texture Information Is Distributed between mips, Using a Laplacian Pyramid.*
- **Unreal Engine** (2025a). *Cinematic Level Visibility Track in Unreal Engine | Unreal Engine 5.7 Documentation | Epic Developer Community.*
- **Unreal Engine** (2025b). *Level Streaming Volumes Reference in Unreal Engine | Unreal Engine 5.7 Documentation | Epic Developer Community.*
- **Unreal Engine** (2025c). *unreal.MoviePipelineQueueEngineSubsystem - Unreal Python 5.6 (Experimental) documentation.*
- **Weiss, M.** (2022). *Command Line Rendering with Unreal Engine Movie Render Queue | Community Tutorial.*

Videos/Talks:

- **Božić, T.** (2025). *Technical Content Grimoire: Volume 2 | Unreal Fest Stockholm 2025*
- **Chew, H.** (2025). *Physically Based Rendering Explained.*
- **Game Maker's Toolkit** (2025). *The World Design of Hollow Knight: Silksong.*

Research Analysis:

As always, a fair amount is my daily general TA research, but I also looked at plugins or tools that could be useful to recreate and have ready for the team in preproduction. I also looked into creating an automatic process for the Movie Render Queue via a Python startup script during this week, so a few sources are related to that.

Weekly Summary:

Development Progress Log

My first focus for the week was on creating a set of cameras to capture development progress throughout the development process. I had created something similar in the previous project, but there were some key lessons learned along the way.

Firstly, too few cameras; it's impossible to go back and take photos of early progress retroactively, and I felt the final collection missed out on some areas I would have liked to show off more. Secondly, there is a lack of or inconsistency in the snapshots taken. Since it was a manual process I was solely in charge of, I tended to forget when I was quite busy, which meant the 'weekly progress' could vary and jump in time quite drastically. To fix this, I decided to have as many cameras as feasible and to set up a process to take snapshots as frequently as possible, with the option to exclude them later as I saw fit.

In this project, this was achieved by setting up over 30 cameras, covering each room, floor overview, and exterior shot, with large rooms containing two cameras.



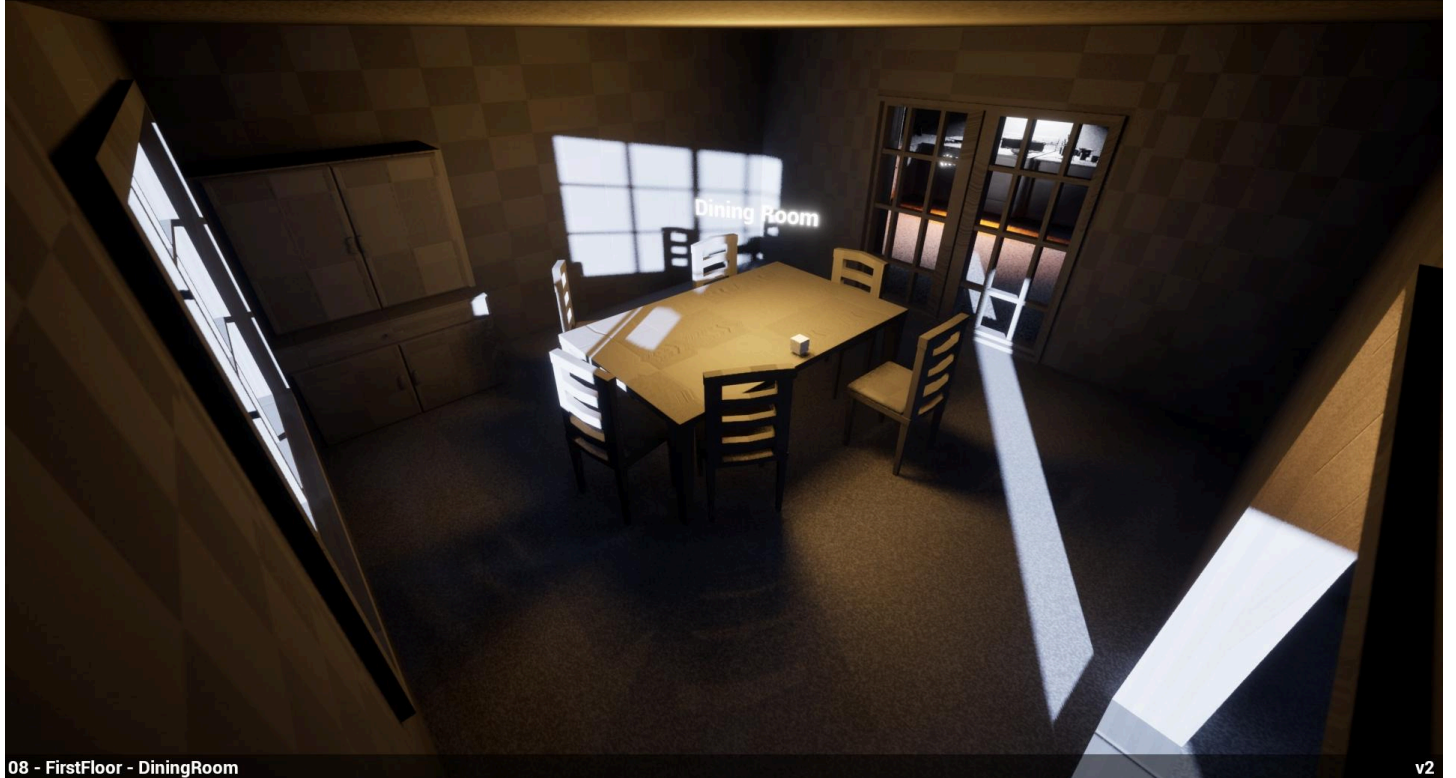
30 cameras across every room on the level.

I placed these into a dedicated sublevel and created a movie render queue, with each camera set to a single frame, and included warm-up, resolution and a burn-in overlay configuration, and more. It exports with the location's ID, Floor, Room,

date, and progress version attached to the image, which is a big improvement.

Project UMBRAE: Development Progress Log

30 Dec 2025



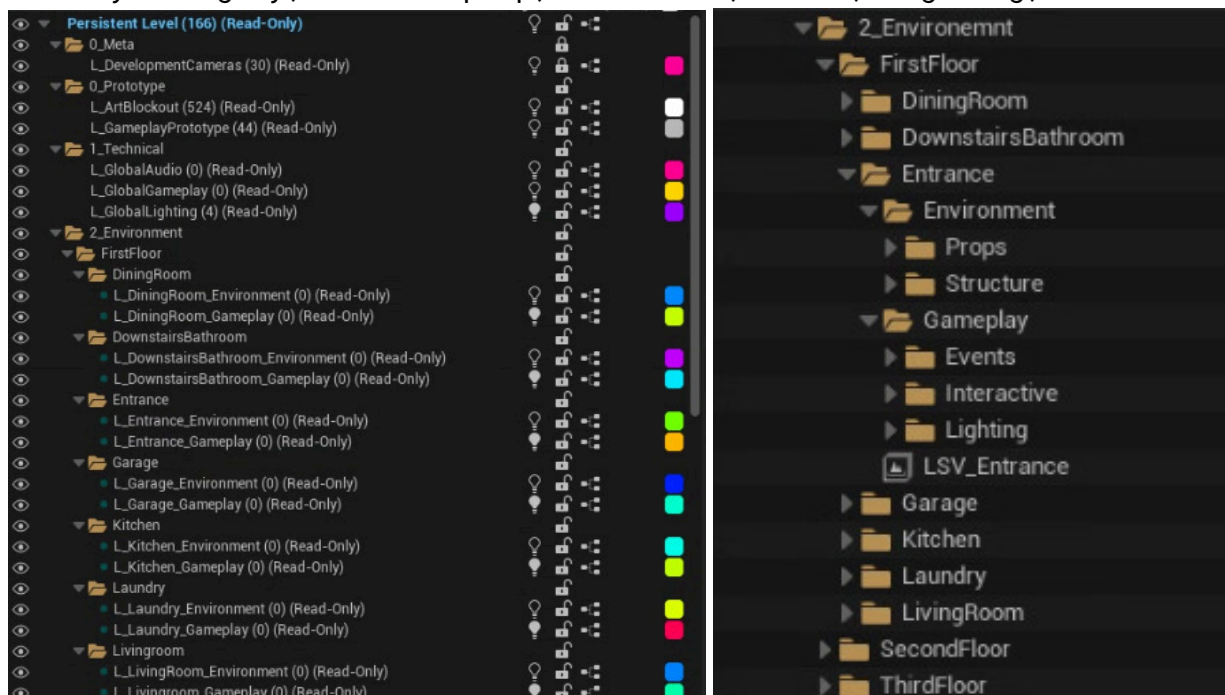
An example capture of the living room.

Project Organisation

With preproduction coming to an end and many of my team members no longer working on the project over the break, it was a great opportunity to clean up and prepare the project itself for production.

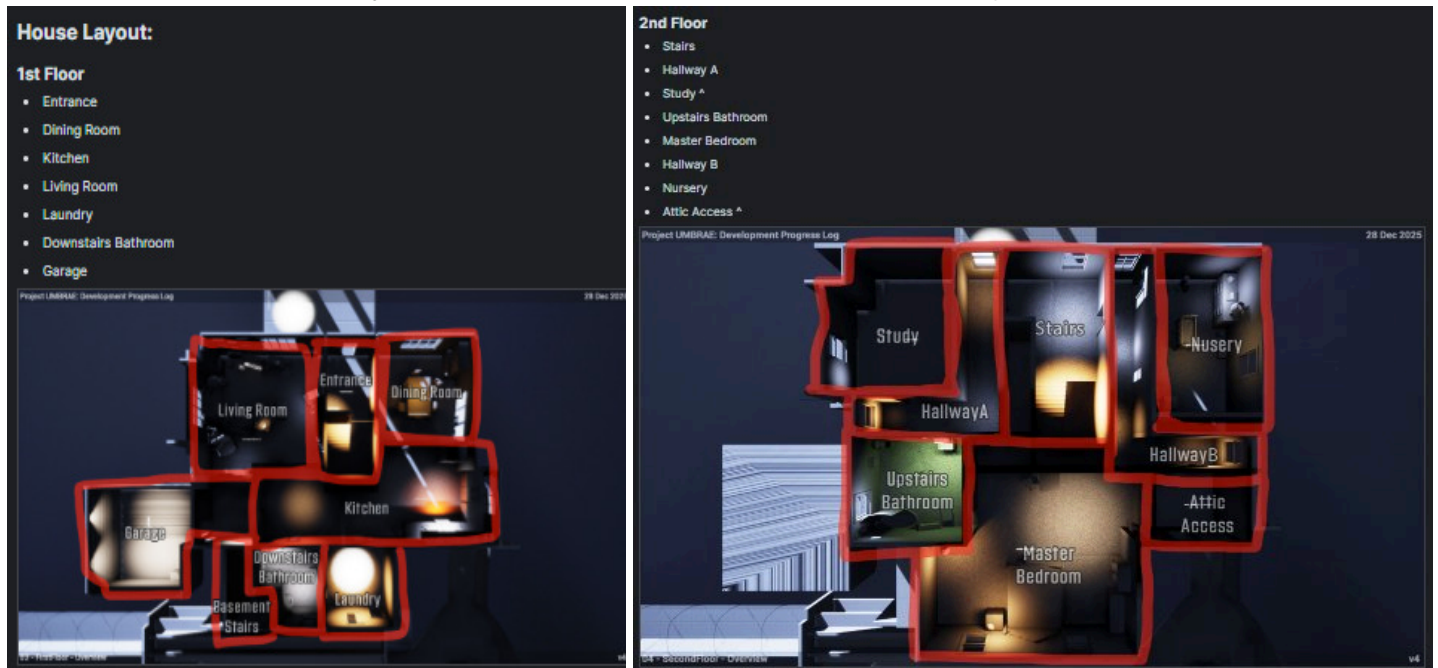
This involved me renaming assets in the content browser to conform to the correct naming conventions and folder structure, and reorganising the level outliner and sublevels to better suit the project, so each room would have only two sublevels: Gameplay - Anything dynamic or interactable, and Environment - Anything static.

These have a mirrored structure in the outliner, then are further split into subfolders by category, such as prop, structure, event, lighting, etc.



Left is the sublevel structure; right is the outline structure.

As part of this, I also created a clear final pre-production layout for each room, so everyone would know which areas counted as which rooms and how they should be referred to, importantly, with a unique name to avoid any confusion. This is broken down by floor and included in the knowledge base.

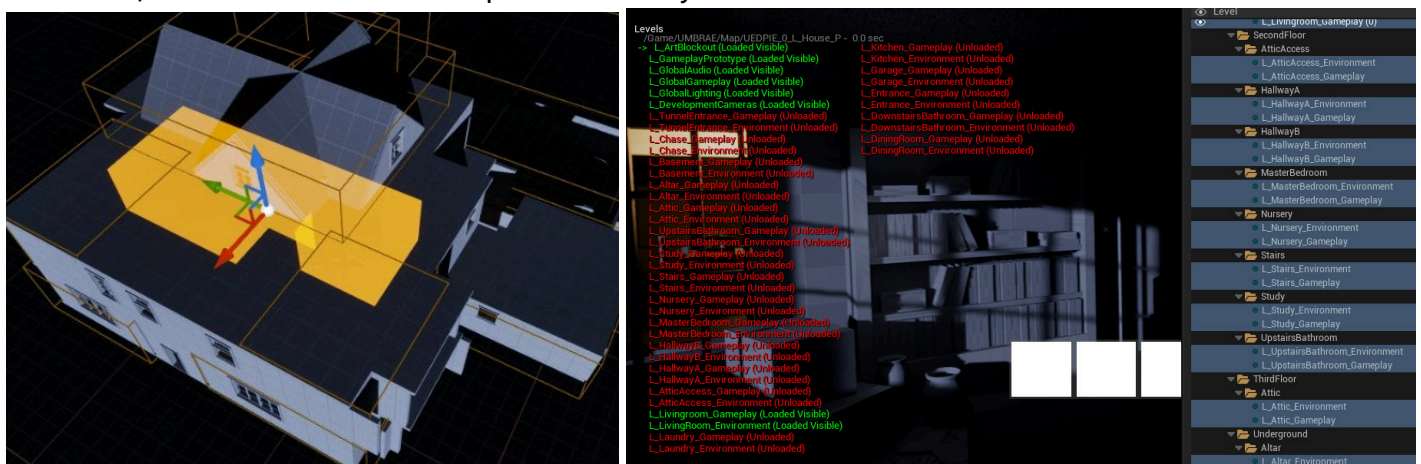


First and second floors of the preproduction layout on the knowledge base.

Level Streaming

Now, with this room-by-room setup, it should also make optimisation much easier and enable aggressive culling, supporting a higher-fidelity environment, a core principle of our project. In preparation, I created a level streaming volume containing the actors for one room and assigned them to that room's gameplay and environment sublevels.

For now, they only load their room, but this can easily be configured to any rooms visible within that room's view. This means the game will not even need to consider it during later culling if an entirely different floor is shown, and it will be explicitly culled before then. It cannot be overstated how much easier it makes my job as a TA for performance and how much it reduces the computer's workload, since it needs to perform many fewer checks.



Left: the kitchen's LSV highlighted. Right: a PiE view of the loaded levels.

These are now controlled by blueprint rather than always being loaded, and I also made sure that these levels all become visible in the level sequence, as otherwise they don't trigger the level streaming volume and won't be visible.

Weekly Reflection:

This week went quite well; I'm pretty happy with what I achieved, and organising the project will be very useful to the team. I feel better going into production. I've focused a lot, especially this week, on planning and preparation, making sure that the team and I don't face any major bottlenecks or challenges in workflow, pipeline, limitations, or goals. Next week, I plan to work on the asset validator, which will hopefully help the team maintain the style guide and asset limitations. I feel I'm on the right track.

I didn't face any issues this week.

Objective:

The second week of the holidays focused on only one thing: the asset validator. I started to concept this tool very early on, but knew I would have other priorities first. With the team not actively working and production not started, it was the perfect time to work on it. I also hoped that if I could get it into a good first state, it might also be beneficial to offer to any other team, as it could provide useful information on any Unreal project, no matter the production context. As such, I focused on making the tool generic and easily configurable. However, I would provide our team's setup as an example template.

Research:

This week's resources used/researched:

Forum/Blogs:

- **DowlingSoka, R.** (2023a). #notGDC-2023: *Unreal Post Asset Import Actions*.
- **ExtraLifeMatt** (2018). *Python to Open / Save assets using Editor*.

Articles/Documentation:

- **DowlingSoka, R.** (2023b). *GitHub-Ryan-DowlingSoka/UnrealImporterRules-CPP*.
- **DowlingSoka, R.** (2024). *GitHub-Ryan-DowlingSoka/UnrealImporterRules-Python*.
- **Unreal Engine** (2026a). *Data Validation in Unreal Engine | Unreal Engine 5.7 Documentation | Epic Developer Community*.
- **Unreal Engine** (2026b). *Plugins in Unreal Engine | Unreal Engine 5.6 Documentation | Epic Developer Community*.

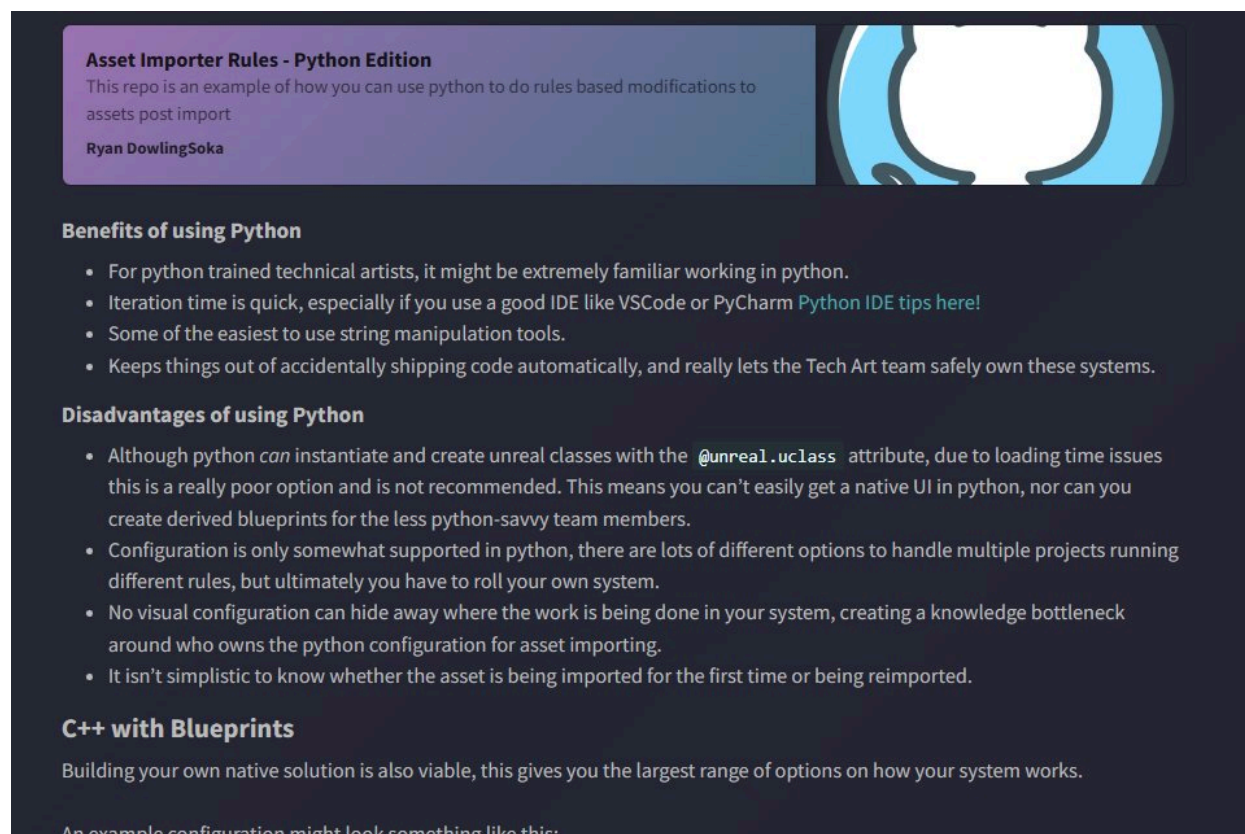
Research Analysis:

This week, I focused entirely on the asset validator tool and didn't spend much time researching anything else. My idea of what I wanted to make was quite set since I did most of my tool research last week. The most useful source found during research was Unreal Post Asset Import Actions, which provided a jumping-off point for my tool development.

Weekly Summary:

Asset Validator

Firstly, I spent a long time thinking about how I wanted it to work and what it should feel like. Unreal already has a data validator, so I looked into it. I immediately felt it was quite restrictive and would have to be in a blueprint, which I've learnt with tools like this should be avoided, as they can be very limited in what is exposed from C++. I also didn't like their workflow, so I decided to base it on a tool I had found, reposted by its author. Looking into it more deeply, it approached the problem the way I wanted to, and while the context was different, I thought it would serve me well. Ryan's UnrealImporterRules actually provided two proof-of-concept versions: Python and C++. In his blog, he delved deeper into the pros and cons of each. With the control I wanted, I quickly realised that while the Python version was useful for checking, his C++ version would be necessary, as Epic hasn't exposed all the classes and variables I wanted access to.



Asset Importer Rules - Python Edition
This repo is an example of how you can use python to do rules based modifications to assets post import
Ryan DowlingSoka

Benefits of using Python

- For python trained technical artists, it might be extremely familiar working in python.
- Iteration time is quick, especially if you use a good IDE like VSCode or PyCharm [Python IDE tips here!](#)
- Some of the easiest to use string manipulation tools.
- Keeps things out of accidentally shipping code automatically, and really lets the Tech Art team safely own these systems.

Disadvantages of using Python

- Although python *can* instantiate and create unreal classes with the `@unreal.uclass` attribute, due to loading time issues this is a really poor option and is not recommended. This means you can't easily get a native UI in python, nor can you create derived blueprints for the less python-savvy team members.
- Configuration is only somewhat supported in python, there are lots of different options to handle multiple projects running different rules, but ultimately you have to roll your own system.
- No visual configuration can hide away where the work is being done in your system, creating a knowledge bottleneck around who owns the python configuration for asset importing.
- It isn't simplistic to know whether the asset is being imported for the first time or being reimported.

C++ with Blueprints

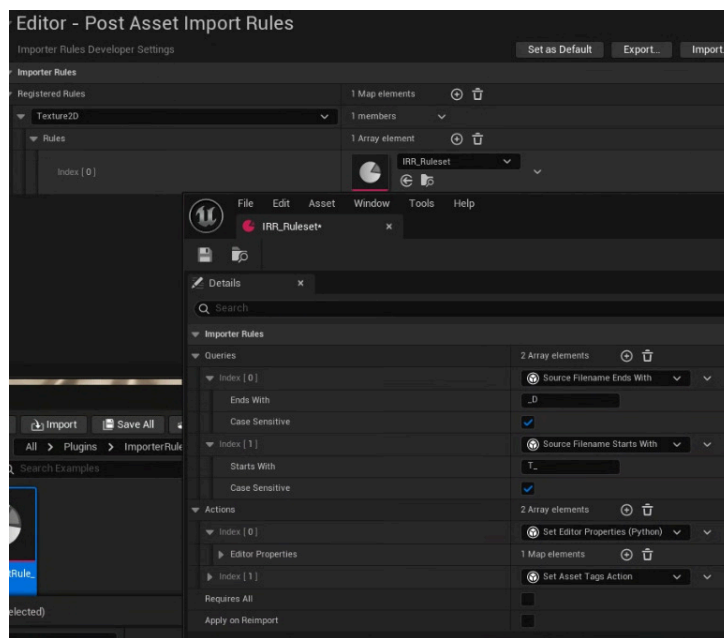
Building your own native solution is also viable, this gives you the largest range of options on how your system works.

An example configuration might look something like this:

Pros and cons of using the Python approach (DowlingSoka, 2023a)

I spent most of this week getting familiar with how his code base worked:

- **Rules** - A pattern that applies to an asset of the set unreal class that checks its queries; it can require all queries to apply, or just one. It then applies all its actions to that asset.
- **Queries** - A if statement we can create to check if the asset meets the condition.
- **Actions** - Any arbitrary behaviour that can be executed.



Example of a rule with queries and actions.

Within the project setting, the team can define what rules are run. I want to expand on this and create a variety of actions and queries that should cover most use cases. I would also like to use metadata tags so a team member can have assets ignored. Getting the system to a state where it not only tells the user an asset is misconfigured but also provides an auto-fix button would be perfect.

Weekly Reflection:

Reflection

I wasn't able to work on it as much as I wanted, but I have a good idea of the tool's direction and hope to continue development next week so we can include it in the project before we get too far into production. I would like to clean up my Python code for the automatic development progress log. I still feel like I am on track for the project.

Problems

No problems encountered, but it took me a bit to remember how to read and write in C++, as it has been a while. It requires precompilation, header files, explicit variable types and more defined syntax such as brackets. While I have experience, I haven't done very much Unreal integration yet and have been focused on other languages since. I was able to pick it back up fairly quickly and hope that will be reflected next week.

Reading: Production Prep

Objective:

In the reading week, I focused on the other major aspect of my plugin: naming conventions. I spent a lot of time thinking about this before this project, and so I had a fairly good understanding of how I would implement it. I wanted to spend basically the entire week creating this. I also wanted to revisit my previous Python code to create a more cohesive, modular system for introducing new Python scripts.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Epic Developer Community** (2026a). *Python logging calls result in an error in LogPython.*
<https://forums.unrealengine.com/t/python-logging-calls-result-in-error-in-logpython/446430>
- **Epic Developer Community** (2026b). *Python to Open / Save assets using Editor.*
<https://forums.unrealengine.com/t/python-to-open-save-assets-using-editor/119251>
- **Epic Developer Community** (2026c). *Descriptor 'get_editor_world' of 'UnrealEditorSubsystem'.*
<https://forums.unrealengine.com/t/descriptor-get-editor-world-of-unrealeditor-subsystem-object-needs-an-argument/625651>

Articles/Documentation:

- **Allar, M.** (2022). *ue5-style-guide: An attempt to make Unreal Engine projects more consistent.* <https://github.com/Allar/ue5-style-guide>
- **DowlingSoka, R.** (2023a). *#notGDC-2023: Unreal Post Asset Import Actions.*
<https://ryandowlingsoka.com/unreal/post-asset-import-rules/>
- **DowlingSoka, R.** (2023b). *UnrealImporterRules-CPP.*
<https://github.com/Ryan-DowlingSoka/UnrealImporterRules-CPP>
- **DowlingSoka, R.** (2024). *UnrealImporterRules-Python.*
<https://github.com/Ryan-DowlingSoka/UnrealImporterRules-Python>
- **Lei, X.** (2025). *Unreal Rendering Workflow with Python Automation | Tech Art Learning.* <https://www.xingyulei.com/post/ue-rendering-basic/index.html>
- **Epic Games** (2026a). *EditorLevelLibrary | Python API Reference.*
<https://dev.epicgames.com/documentation/en-us/unreal-engine/python-api/class/EditorLevelLibrary>
- **Epic Games** (2026b). *SoftObjectPath | Python API Reference.*
https://dev.epicgames.com/documentation/en-us/unreal-engine/python-api/class/SoftObjectPath?application_version=5.5

Videos/Talks:

- **Ben Cloward** (2026a). *Painting Holes in Terrain - Terrain Shaders - Episode 10.* https://www.youtube.com/watch?v=-gVJq8_aEA
- **Ben Cloward** (2026b). *Terrain & Landscape Basics - Terrain Shaders - Episode 11.* https://www.youtube.com/watch?v=r_tiyXdFlSE
- **Game Maker's Toolkit** (2025). *The World Design of Hollow Knight: Silksong.*
<https://www.youtube.com/watch?v=l7xXW0EbK4g>
- **SIGGRAPH** (2025). *Advances in Real-Time Rendering: Fast as Hell: idTech8 Global Illumination.* <https://www.youtube.com/watch?v=VTrdeqMMMk0>
- **Ghislain Girardot** (2025). *Hacking your way around UE's Character Movement Component.* <https://www.youtube.com/watch?v=2sLa4z4n0lI>

Research Analysis:

This week, my research focused on best practices for Unreal Python coding and naming conventions. I also continued to do some general research, look at industry examples of good optimisation and game design techniques, and kept up with Ben Cloward's landscape terrain shader series. A small portion of my research involved fixing minor bugs in asset loading. I reused the post-import actions from Dowling Soka's Import Rules that I had not used in my rule setup to start the Naming Dialogue.

Weekly Summary:

Naming Convention system

Based on the style guide we used previously and given that professional practices are a part of our grade for this module, I thought it would be beneficial to provide an automated process that is non-invasive and requires user action before any final decisions are made, while still speeding up asset naming.

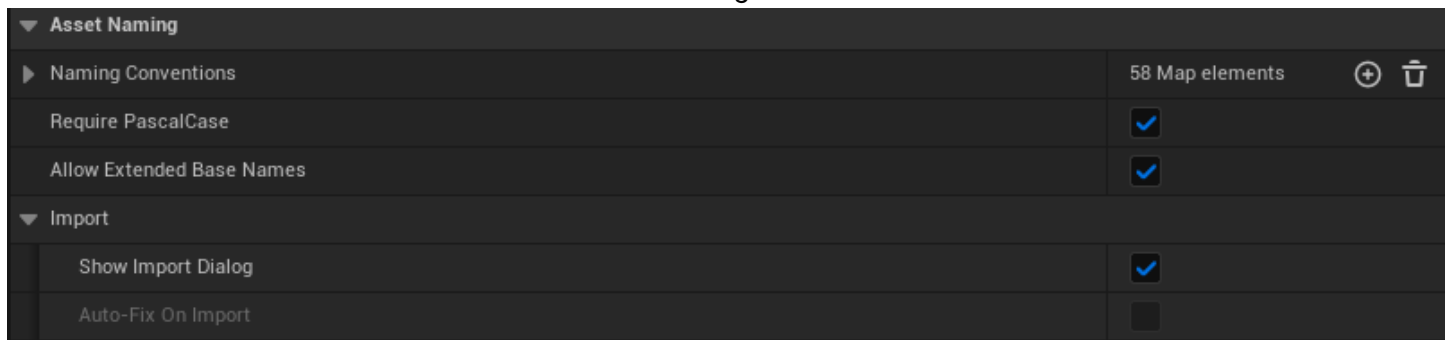
It follows the fairly standard practice of 'Prefix_BaseAssetName_Variant_Suffix' With prefixes that define the asset type, e.g., BP for Blueprint, T for Texture, etc.

BaseName should be the core asset name and use PascalCase, e.g. PlayerCharacter. A variant should separate between assets of the same type, e.g., 01, Red, Large. The suffix refers to a subtype, such as textures: D for Diffuse, N for Normals.

So a diffuse texture with the source name 'cardboard box.png' becomes 'T_CardboardBox_D'.

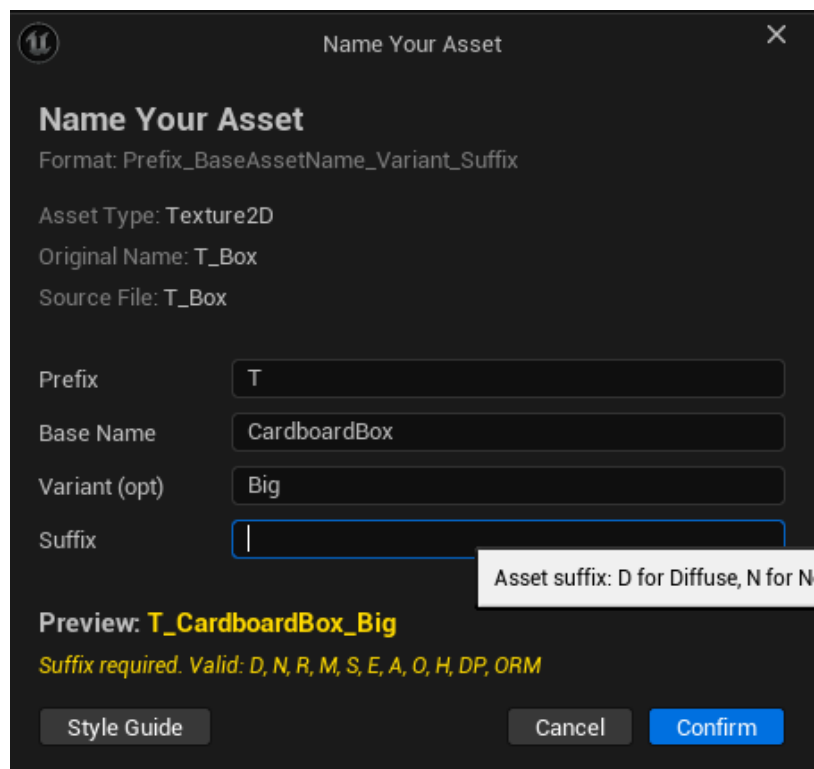
I built two main components to make this possible:

- **Plugin settings** - Let users add their own conventions on top of the existing ones set out by Allar's Style Guide, with support for any class to have any prefix or suffix status, as well as compact common suffix aliases like Diffuse to D. I also included settings for the other new features.



Asset Naming plugin settings.

- **Import dialogue** - A pop-up that shows a preview of how assets would be named, and an explanation of why it wouldn't meet the guideline. If the file contains this information, it is automatically added or set in the plugin settings.



Import dialogue showing preview with no suffix and suggestions.

There's a lot of boring but important work behind the scenes to make it fill out fields automatically and verify that it meets the conditions, which involve many exception cases for semantics and error handling.

```
FString UAssetNamingParser::GenerateSuggestedName(const FString& CurrentName, const FAssetNamingConvention& Convention)
{
    FParsedAssetName Parsed = ParseAssetNameWithConvention(CurrentName, Convention);
    FParsedAssetName Corrected;

    Corrected.Prefix = Convention.GetPrefixWithoutUnderscore();

    const UAssetHandlingSettings* Settings = UAssetHandlingSettings::Get();
    if (!Parsed.BaseAssetName.IsEmpty())
    {
        FString ProcessedBaseName = ConvertNumbersToWords(Parsed.BaseAssetName);
        if (Settings && Settings->bRequirePascalCase)
        {
            // Extended names: apply PascalCase per segment to keep underscores
            if (Settings->bAllowExtendedBaseName && ProcessedBaseName.Contains(TEXT("_")))
            {
                TArray<FString> Segments;
                ProcessedBaseName.ParseIntoArray(Segments, TEXT("_"), true);
                for (FString& Segment : Segments)
                {
                    Segment = ToPascalCase(Segment);
                }
                ProcessedBaseName = FString::Join(Segments, TEXT("_"));
            }
            else
            {
                ProcessedBaseName = ToPascalCase(ProcessedBaseName);
            }
        }
        Corrected.BaseAssetName = ProcessedBaseName;
    }

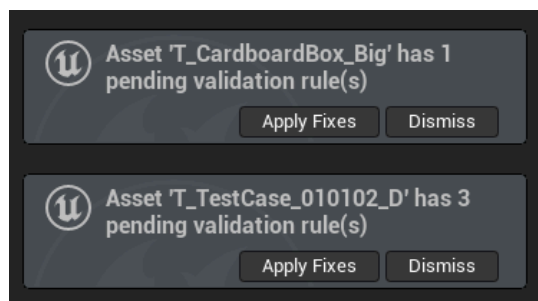
    if (Corrected.BaseAssetName.IsEmpty())
    {
        FString FallbackName = ConvertNumbersToWords(CurrentName);
        Corrected.BaseAssetName = ToPascalCase(FallbackName);
        if (Corrected.BaseAssetName.IsEmpty())
        {
            Corrected.BaseAssetName = TEXT("NewAsset");
        }
    }
}
```

Generate suggested name functions for parsing in the import dialogue.

General Plugin Improvements

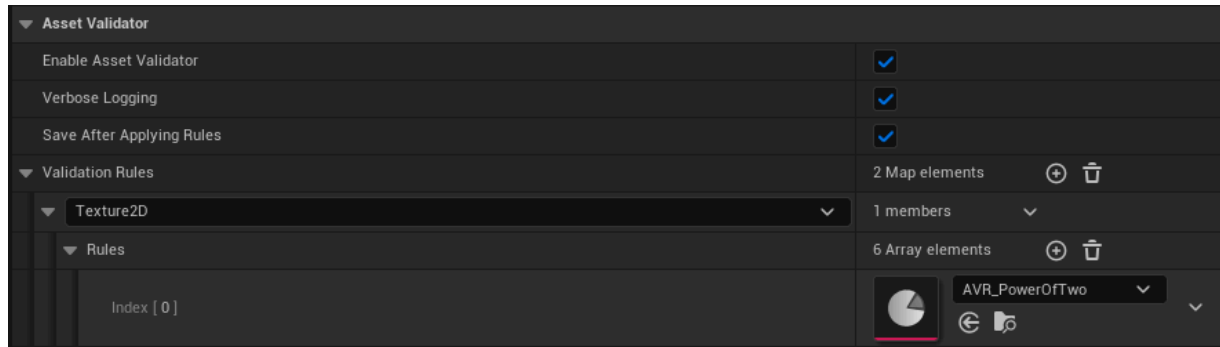
I also made several more general improvements to the plugin:

- **Rule pop-ups** - Toast notifications appear when assets have pending validation rules, with buttons to apply fixes.



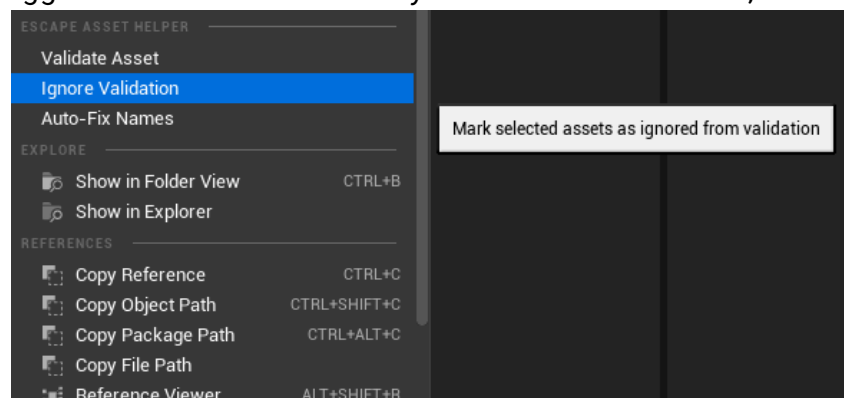
Pending rules notification.

- **Better rule plugin settings** - Reorganised the Asset Validator settings layout with more control and a better structure.



Asset Validator Plugin settings.

- **Default settings** - So the user doesn't need to fill out the naming conventions and rules; the plugin passes the recommended default settings to the project automatically.
- **Separating Modules** - I separated the plugin into handling, naming, and rules modules to make it easier to enable or disable and to read the plugin logic.
- **Style guide link** - Added a configurable style guide URL used in the import, incorrect assets, and project help sections to make it easily accessible.
- **Ignore system** - Assets wouldn't be checked if they had a metadata tag or were in a certain directory, useful for engine or third-party content.
- **Manual validate asset** - A batch check to validate that all selected assets have the right naming convention and rules applied.
- **Auto-Fix names** - Using the same logic as the import dialogue, a content asset can suggest and automatically fix an asset name, which can be batched.



Context Menu Actions for validate, ignore, and auto-fix.

Python Refactor

I also refactored the scripts I had written earlier because a minor bug prevented the JSON from being written correctly. I also took this as an opportunity to refactor the system to use a shared library, called `utils`, that I can reuse later in other modules, such as logging and asset retrieval, and to comment the overall code now that it was simplified.

```

8  # Libs
9  import os
10 import datetime
11 import time
12 import unreal
13 import utils
14 log = utils.Log("Progress Log")
15
16 # Config
17 STATE_FILE = "ProgressLog"
18 USER = "MarlandROG"
19 QUEUE_ASSET = "Meta/DevelopmentSequence/MPC_Development"
20
21 def run_mrqr():
22     # Load State
23     state = utils.load_state(log, STATE_FILE)
24     last_run = state.get("last_run", "")
25     version = int(state.get("version", 0))
26
27     # Check if its correct user
28     if USER != os.getlogin():
29         log.info("Not correct User. Aborted.")
30         return
31
32     # Check if its a new day
33     if last_run == str(datetime.date.today()):
34         log.info("Already ran today. Skipping.")
35         return
36
37     # Create MRQ
38     version += 1
39     log.info(f"New day, preparing MRQ. (Version {version})")
40     mrq_subsystem = unreal.get_editor_subsystem(unreal.MoviePipelineQueueSubsystem)
41     queue = mrq_subsystem.get_queue()
42     queue.copy_from(utils.load_asset(log, QUEUE_ASSET))
43
44     # Wait for level to be loaded
45     loaded = False
46     TRIES = 20
47     for i in range(TRIES):
48         if unreal.get_editor_subsystem(unreal.UnrealEditorSubsystem).get_editor_world():
49             log.info("MRQ level is loaded. Executing.")
50             loaded = True
51             break
52         log.warning(f"MRQ is waiting on the level to be loaded! Tried {i}/{TRIES} times.")
53         time.sleep(1)
54
55     if not loaded:
56         log.error("Level failed to load in time!")
57         return
58
59     # Execute MRQ
60     executor = unreal.MoviePipelinePIEExecutor()
61     try:
62         executor.on_executor_finished_delegate.add_callable(on_mrqr_finished)
63         mrq_subsystem.render_queue_with_executor_instance(executor)
64         log.info("MRQ now running.")
65     except Exception as e:
66         log.error(e)
67
68     # Save new state
69     state = {"last_run": str(datetime.date.today()), "version": version}
70     utils.save_state(log, STATE_FILE, state)
71
72 def on_mrqr_finished(executor, success):
73     if success:
74         log.INFO("MRQ completed successfully!")
75     else:
76         log.ERROR("MRQ failed or was canceled by the user.")

```

Updated, simplified, and modularised progress log script.

Weekly Reflection:

Reflection

This week went very well; I was much more productive, as predicted. I got back into the flow of C++ coding and implemented many of the core features I wanted for this plugin. The systems work together nicely, and I think it'll be a genuine improvement to my team's workflow. Next week, I plan to continue working on the plugin by creating an asset status overlay. I think I'm in a good position heading into the first week of production.

Problems

The main challenge was with hooking actions on event delegates; otherwise, I was fairly good this week.

WEEK 5: Production Start

Objective:

Week 5 marked the first week of production. First, I wanted to finish the plugin by adding an asset status overlay to make it easy for users to see whether an asset was valid. Secondly, I needed to prepare to release the plugin, which meant fixing any bugs and testing that it worked in the project environment, with unique aspects such as Perforce. Thirdly, as we were now in production, I would need to talk a lot about what everyone would be working on.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Procedural Pixels** (2026). *Procedural Pixels Blog*.
<https://www.proceduralpixels.com/blog>

Articles/Documentation:

- **Epic Games** (2026). *Expert's guide to Unreal Engine performance*.
<https://dev.epicgames.com/community/learning/tutorials/3o6/expert-s-guide-to-unreal-engine-performance>

- **Nistor, I.** (2025). *Asset Status Overlay | Fab*.
<https://www.fab.com/listings/2adac10a-50d4-4232-a9de-a88b8b387ac1>
- **DowlingSoka, R.** (2023). *UnrealImporterRules-CPP*.
<https://github.com/Ryan-DowlingSoka/UnrealImporterRules-CPP>

Videos/Talks:

- **Ben Cloward** (2026). *Adding Terrain Detail Meshes - Terrain Shaders - Episode 12*. <https://www.youtube.com/watch?v=cDs0vgtGjeA>
- **Hugh Chew** (2026). *Flow Maps Explained*.
<https://www.youtube.com/watch?v=eS0dzFP9GW0>
- **Matt's Ramblings** (2023a). *Quake's PVS: A hidden gem of rendering optimisation*. <https://www.youtube.com/watch?v=IfCRHSIg6zo>
- **Matt's Ramblings** (2023b). *How Quake's software renderer ELIMINATES overdraw*.
<https://www.youtube.com/watch?v=zdXsHWHxeBY>

Research Analysis:

This week's research focuses on the asset status overlay plugin I found and on understanding exactly how they extended the content browser. Without this reference, it would have been very difficult to understand. I also looked into broader technical art research, such as performance and culling techniques. As always, I also watched Ben Cloward's latest episode.

Weekly Summary:

Producer

Throughout this first week of production, I ramped up my producer responsibilities. I organised this weekly SCRUM and set out the goals, using YouTrack's Agile Cards to provide a clear list of tasks that everyone could update. This also required reorganising the YouTrack knowledge base into more clearly defined sections, so that any page now falls under Tracking, Design, or Guidance.

Projects / First-Person Psychological Horror / Knowledge Base / FPPHorror-A-8

Filter articles by title

First-Person Psychologi...

[SEARCH INDEX]

Meta

This is to link to the other articles cleanly in subsections for when there are a lot of articles.

TRACKING:

- Master Project Tracker
 - Asset List
 - Sound List
 - Animation Project Management Spreadsheet
 - Animatics
 - Shot List
- Timeline & Milestones
 - Stretch Goals - "The Great Wishing Well"
- Preproduction
 - Pre-Prod Layout
 - Pre-Prod Reference Links
 - Early Concepts & Notes
- Production
 - MVP

DESIGN:

- GDD
 - Puzzles Information
 - Items and Interactables
- Gameplay Walkthrough
- Weekly Feedback
- Art Bible
- Full Story

GUIDANCE:

- Project Structure Guide
- Asset Guide - Style & Targets
- Team Roles
- Original Brief
 - Tutor Notes for First-Person Psychological Horror
- Base Expectations

Updated system for organisation on YouTrack.

I also created a stretch goal document outlining any features we'd like to create, but that are only possible if we have extra time. Having these clearly defined separately makes it clear what we aim to achieve and helps us avoid overscoping our current plan while core features still need work.

Next, I went through the entire asset list and reorganised it to fit our new structure of one asset, one person ownership, to make it easier to track. I then went through each section and moved/removed duplicate assets from different rooms into the generic sheet.

Base Name	Variant (if any)	Type	References	Priority	Interactable	Reusable	Status	Current Stage	Assigned To	Notes	Ideal Locations	Blockout status	Blockout modeller
Potted Plant	Standing, Small, large	Small	Image	5.Stretch	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Dust Bunny	Big, Small	Small	Image	5.Stretch	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Clutter	2 x Small, 2 x Medium, 2 x Large	Small	Image	4.Low	No	Yes	Not started	N/A	Δ Name	Previously from living room		Completed	Charlotte Knight
Clothes	Jeans, 2 x Tshirt	Small	Image	4.Low	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Plug Sockets		Small		4.Low	Yes	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Radiators		Big		4.Low	No	Yes	In progress	Blockout	Δ Name	Each letter		Completed	Δ Name
Floor lamps		Big		4.Low	No	Yes	In progress	Blockout	Daniel Curley	The rings should be one mesh w/ diff textures and separate n		Completed	Daniel Curley
Buckets	1x lid & handle, 1x no lid & no handle, 1x no lid & no handle	Small		4.Low	No	Yes	In progress	UV	Adam Honey	Gameplay use of pill bottles is put on hold		Completed	Adam Honey
Wooden Plank	2 x Short, 1 x Long	Small		4.Low	No	Yes	In progress	Blockout	Δ Name	Notes		Completed	Charlotte Knight
Glass	Tall	Small	Image	4.Low	No	Yes	In progress	Blockout	Max Ebdon	Used in Dining, Kitchen	Dining, Kitchen	Completed	Max Ebdon
Curtains		Big		3.Medium	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Table lamps		Hero		3.Medium	Yes	Yes	In progress	Blockout	Daniel Curley	Notes		Completed	Daniel Curley
Rug	Square	Hero		3.Medium	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Rug	Circular	Small		3.Medium	No	Yes	In progress	Blockout	Sam Tiplady	Previously from nursery		Completed	Sam Tiplady
Books		Small		3.Medium	Yes	Yes	In progress	Blockout	Sam Tiplady	Notes		Completed	Δ Name
Newspaper piles		Small		3.Medium	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Wall mounted clock		Small		3.Medium	Yes	Yes	Not started	N/A	Daniel Curley	Notes			Δ Name
Toolbox		Small		3.Medium	No	Yes	Completed	UV	Adam Honey	Was 4 versions in nursery, basement, garage lol	Nursery, Kitchen, Garage, Basement	Completed	Adam Honey
Glass	Wine	Small	Image	3.Medium	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen	Dining, Kitchen	Completed	Max Ebdon
Glass	Short	Small	Image	3.Medium	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen	Dining, Kitchen, Bathroom	Completed	Max Ebdon
Cardboard Box	2 x Large, 3 x Small	Small	Image	2.High	No	Yes	In progress	Model	Charlotte Knight	Previously from living room & Attic		Completed	Charlotte Knight
Light Switch	Single, Double	Small	Image	2.High	Yes	Yes	Not started	Blockout	Adam Honey	Notes			Δ Name
Lightbulbs/ ceiling lights		Small		2.High	No	Yes	Not started	N/A	Δ Name	Notes			Δ Name
Picture frames	Wall, Table	Small		2.High	Yes	Yes	Not started	N/A	Sam Tiplady	Notes	Nursery & Basement		Δ Name
Tools	Screwdriver, Wrench	Small		2.High	No	Yes	In progress	Blockout	Adam Honey	Was 3 versions in nursery, basement, garage lol	Nursery, Basement, Garage	Completed	Adam Honey
Wooden Chest		Big	Image	2.High	Yes	Yes	In progress	Blockout	Daniel Curley	Looks old	Attic, Bathroom	Completed	Daniel Curley
Plate	China	Small	Image	4.Low	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen	Dining, Kitchen	Completed	Max Ebdon
Plate	White	Small	Image	4.Low	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen	Dining, Kitchen	Completed	Max Ebdon
Match Box		Small		5.Stretch	Yes	Yes	Not started	N/A	Δ Name	Closed and maybe open?	Basement, Kitchen, Garage		Δ Name
Candles		Small		2.High	No	Yes	Completed	UV	Adam Honey	Notes	Dining + Storage Areas		Δ Name
Mug	White	Small		4.Low	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen		Completed	Max Ebdon
Mug	Escape	Small		5.Stretch	No	Yes	In progress	Blockout	Max Ebdon	Previously from Kitchen		Completed	Max Ebdon
Cutlery	Knife	Small		4.Low	No	Yes	Not started	N/A	Max Ebdon	Previously from Kitchen	Dining, Kitchen		Δ Name
Cutlery	Fork	Small		4.Low	No	Yes	Not started	N/A	Max Ebdon	Previously from Kitchen	Dining, Kitchen		Δ Name

From a technical art perspective, I advised Charlotte on the modular kit plans and the lead animators on how best to implement animations in the project, especially for the player. Getting people aligned meant that we would have less technical debt later. As part of this, I added the creature and player test animation to figure out export and import settings.

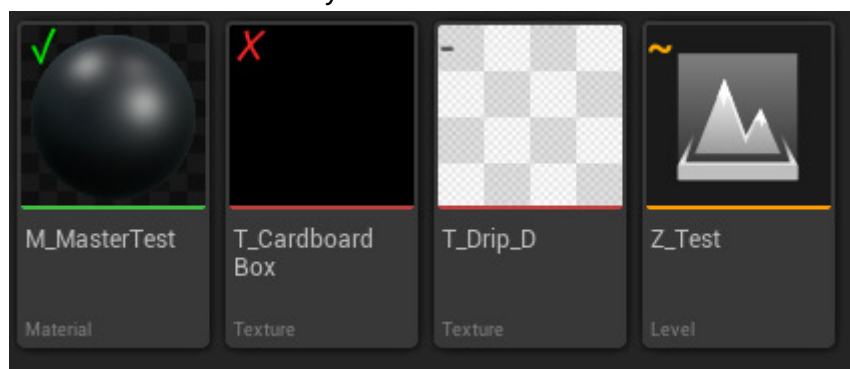
I presented the project's status in the Friday meeting with the tutors. We didn't do a playtest this week, but still needed to prepare for next week's.

Asset Status Overlay

The main focus of this week was the asset status overlay. Making clear to the user whether an asset was validated was important. Inspired by IANistor's approach, which describes the development stages of an asset, I aimed to learn the techniques required to extend the content browser's UI. Interestingly, the browser did not support icons, so text symbols were used.

There are 4 validation states, each with a different symbol and text colour:

- **Valid (✓)** - A green tick means the asset passes all validation checks and follows the naming convention.
- **Invalid (X)** - A red cross means the asset fails the naming convention and needs to be renamed.
- **Ignored (-)** - A grey dash means the asset has been marked as ignored and won't be checked.
- **No Convention (~)** - An orange tilde means there's no naming convention defined for that asset class yet.



Example Assets with their statuses.

Unreal does cache assets in the content browser, so once something has been updated via a rename, the status needs to be re-cached. This took a fair amount of time to figure out, but once working, it meant all assets were properly reflected in 'real-time'. I also made this configurable in the plugin settings.



Asset Status Overlay Plugin Settings.

Plugin Release Preparation

I also spent some more time polishing the overall plugin so that it was configurable, organised and provided a good user experience. This meant fixing as many bugs as possible and improving the settings layout. I also expanded the number of actions and queries included with the plugin so that, in theory, most user cases can be covered without creating custom ones.

Weekly Reflection:

Reflection

In the first week of production, many aspects of the project moved quickly, bringing a different energy to the team as they began creating final content. The plugin is in a really good state; it's now feature-complete, and I feel confident that, after some more fixes and organisation, it will be accessible. One big consideration this week was that the exact execution of puzzles and game design needed to be laid out, so I want to have a big talk starting next week, running through the game from start to finish. I would also like to work on releasing the plugin. I feel like we're in a good position overall on the schedule.

Problems

I had some headaches figuring out how to extend the content browser and integrate with the internal system, since I hadn't modified it before. With the other plugin as a reference, I eventually got it to work. I also took a while to figure out how to re-cache the assets, as it's not the same as marking them as modified and resaving.

WEEK 6: Planning Content

Objective:

Week 6 had two focuses. First, I wanted to run through the entire game with the team so we could all clarify our ideas and come to a decisive decision about what was in the game. Second, I wanted to release a public version of my plugin, Escape Asset Helper, on GitHub, which meant polishing it and writing documentation and an installation guide.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Epic Developer Community** (2026). *Source Control: "Query File State" not Returning Correct Data.*
<https://forums.unrealengine.com/t/source-control-query-file-state-not-returning-correct-data/481512>
- **Epic Developer Community** (2026). *Revision Control Error.*
<https://forums.unrealengine.com/t/revision-control-error/2008839>
- **Epic Developer Community** (2026). *How can I get the width and height of a texture in pixels?*
<https://forums.unrealengine.com/t/how-can-i-get-the-width-and-height-of-a-texture-in-pixels/285691>
- **Nistor, I.** (2025). *IANistor - Asset Status Overlay | Forums.*
<https://forums.unrealengine.com/t/ianistor-asset-status-overlay/2670578>
- **Tech Art Aid** (2026). *Tech Art Aid - Technical Art Resources.*
<https://techartaid.com/>

Articles/Documentation:

- **Epic Games** (2026a). *UTexture2D* | C++ API Reference.
<https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Runtime/Engine/UTexture2D>
- **Epic Games** (2026b). *TextureCompressionSettings* | C++ API Reference.
<https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Runtime/Engine/TextureCompressionSettings>
- **Epic Games** (2026c). *ETexturePowerOfTwoSetting* | C++ API Reference.
https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Runtime/Engine/ETexturePowerOfTwoSetting_Type
- **Allar, M.** (2022). *ue5-style-guide: An attempt to make Unreal Engine projects more consistent.* <https://github.com/Allar/ue5-style-guide>
- **Nistor, I.** (2025). *Asset Status Overlay* | Fab.
<https://www.fab.com/listings/2adac10a-50d4-4232-a9de-a88b8b387ac1>
- **Epic Games** (2026). *ISourceControlModule* | C++ API Reference.
<https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Developer/SourceControl/ISourceControlModule>

Videos/Talks:

- **YouTube** (2026). *Massively reduce texture size with Editor scripting in UE5.*
<https://www.youtube.com/watch?v=cdiZeJ5LSwU>
- **YouTube** (2026). *Your First Blueprint Validator in Unreal Engine - Square Texture Validator.* <https://www.youtube.com/watch?v=Y4eBybP2LUo>
- **YouTube** (2026). *Get Rid of LightMapUVs (Editor Utilities from SoloDev to AAADev).* <https://www.youtube.com/watch?v=MfNUu0xZn2Q>
- **Ben Cloward** (2026). *Terrain Texture Node Parameters - Terrain Shaders - Episode 13.* <https://www.youtube.com/watch?v=DKG-qnJ8GDq>
- **Dylearn** (2026). *How I made grass better than 99% of games | Stylised grass 3D pixel art.* <https://www.youtube.com/watch?v=0xsuWDtjuGw>
- **jacksepticeye** (2026). *BrokenLore: UNFOLLOW (FULL GAME).*
<https://www.youtube.com/watch?v=A4cQwP2I-E8>

Research Analysis:

This week's research focused on creating a solid default set of validation rules and on source control integration, since every time we used Perforce, it required slight rework or support for handling renames. I spent a lot of time reviewing the UE5 C++ API docs to understand how to do this, and revisiting a blueprint source control implementation I had found earlier. I also used the two plugin setups I had previously used as a basis for formatting my documentation and installation guide. On the side, I continued my general technical art research and watched a playthrough of a game very similar to what we wanted to create, to learn what they did well and what they could improve, and to keep in mind for our game.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. Unfortunately, I was very busy and wasn't able to properly reflect on it with cards this week.

A core aspect of this week was to go through the whole week and clarify every aspect of the gameplay with the whole team. Across Monday and Tuesday, we went from start to finish, and I noted down every branch and puzzle. I would then compile these notes into a proper Gameplay walkthrough document that describes each step and choice in the game. It was very beneficial, as while everyone was

fairly well-versed in their areas, we hadn't sat down together. It also turned out that many ideas and puzzles weren't fully fleshed out, and we had gaps between points.

★ Gameplay Walkthrough

This document describes the route and gameplay goals across the game.

Reminder: Tiles can be placed in any order of collection and at anytime in the Basement Stone Door.
(Stretch - The room that the tile is connected to would become more derelict and contain tentacles.)

Gameplay walkthrough

Start:

- **Wake Up on Couch:** You wake up on the couch, with the TV in front, rambling about some news.
- **Mother's Voicemail:** A blinking red light is on a side table near the door. Your mother explains that you must do some house-sitting, and your aunt's recent passing.
- **Power Outage:** Your mother explains how your aunt grew distant from the family, and the line cuts out as the house plunges into darkness, lit only by moonlight.
- **Heading to the Garage:** Exploring past a pitch-black basement door and into the garage, a single workshop light remains on.
- **Obtaining the Flashlight:** Amongst the workshop clutter is an old flashlight, clearly not reliable, but it will do.
- **Into the Basement:** With the flashlight, the player can enter the basement. Ahead, dim red lighting creeps around the corner.
- **Basement Door:** Behind a ruined wall lies a tunnel leading to an ancient stone door with nine recessed square symbols. The Basement Tile on the ground in front fits.
- **Breaker Switch:** At the back of the basement is the breaker switch; turning it on restores power to the house. Loud footsteps are heard above.
- **Into the Living Room:** You return upstairs. Entering the living room. The door next to the receiver is now open.
- **Entrance:** Entering the entrance, a key in the key bowl gives access to the Dining room and Pathway A, or can be ignored and can go upstairs with Pathway B.

First Floor Branches:

You have three choices, which can be completed in any order: Pathway A - Living Room & Kitchen Tile, Pathway B - Go Upstairs.

Pathway A- Living Room & Kitchen Tile:

- **Shadow Puzzle:** In the living room, a statue is missing from the fireplace mantel; in the dining room, a glass case holds this statue. Putting it in its place lets you move them to spell out the words 'HIDE'. The fireplace opens. hands come bursting out, and with it, The Living Room Tile.
- **Kitchen Access:** Inside the dining room, a double glass door has a note attached. The note suggests getting a new handle for the door in the garage. Searching in the garage, you can find said handle and use it to open the Kitchen.

Gameplay Walkthrough on YouTrack.

To realign with our updated plans, I reduced the sub-levels so each level was either in environment or gameplay, updated the development camera positions, added a simple set dress to the study, and improved the organisation of the Knowledge base.

Playtest feedback

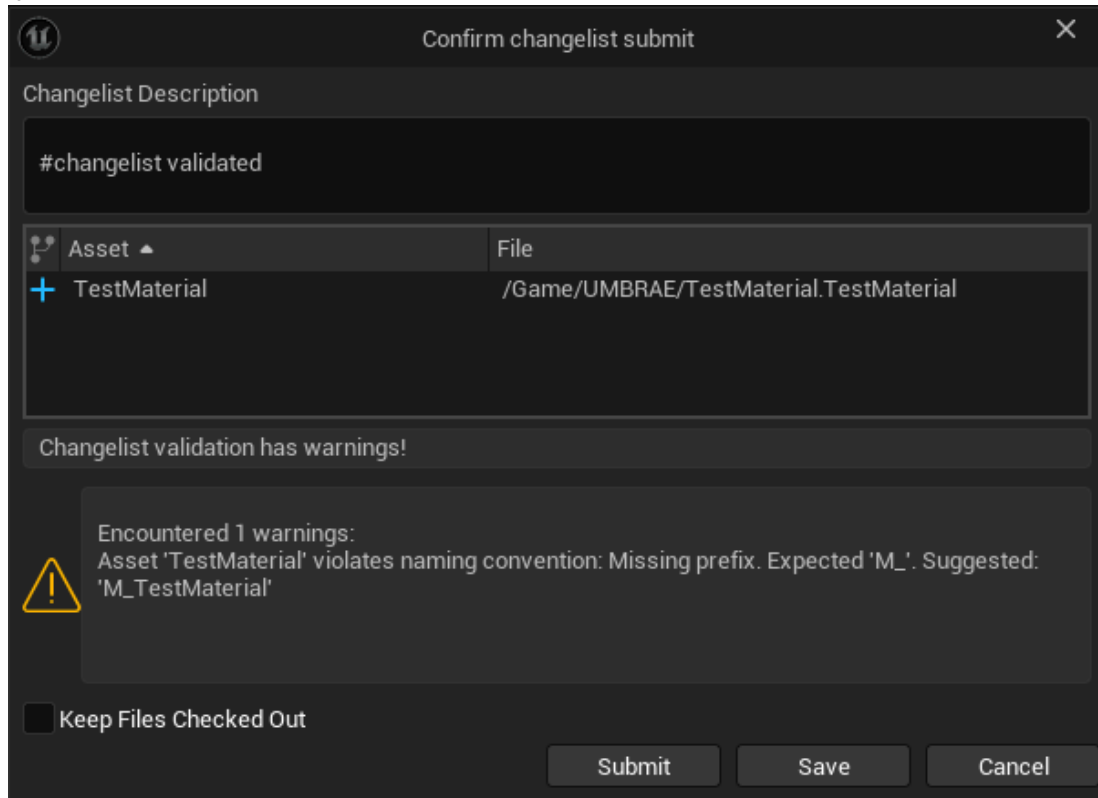
With Lauren and Kanye, we conducted our first playtest session. Which was quite positively received, even at such an early stage. With Lauren preparing the feedback form and Kanye leading the collection, we were able to gather some useful feedback. The main insights were:

- Player experience reinforced that, even with limited content, exploration was natural and fun. This is great, as it's a core pillar and will remain a focus.
- Puzzles were well received and gave rewarding 'Ah Hah!' moments, but players wanted more depth and challenge. Suggestions included hiding the code lock puzzle code across the house.
- Players often did not go to the basement first and, as such, did not understand the overarching objectives until they did. Set the players up so they encounter it first with the flashlight, which leads them to it.
- Players struggled to interact properly with small puzzles due to head bobbing and scale. Locking the player into a set puzzle cam would fix this.
- Players did not like the flashlight's fixed position. Allowing them to move it up and down as well would provide more freedom.
- Even with the basic scare, it was quite successful and showed that the environment and tone were set.
- Controls were not explained, and often required different keys.

Plugin Source control

A tough challenge I faced was implementing source control. Checking whether it was active and querying the asset's status in the Source Control Module were annoying. I also had to manually set up renamed assets to be added to the changelists and checked out, etc. I also made it so that on the Submit Changelist, it would warn the user or block commits that didn't meet naming

convention standards. This also took some time to figure out, but eventually I was able to. This was quite important as it should really help catch mistakes before they are submitted and deter bad practices.



Submit Changelist has a custom-naming-convention warning.

Plugin Default Rules

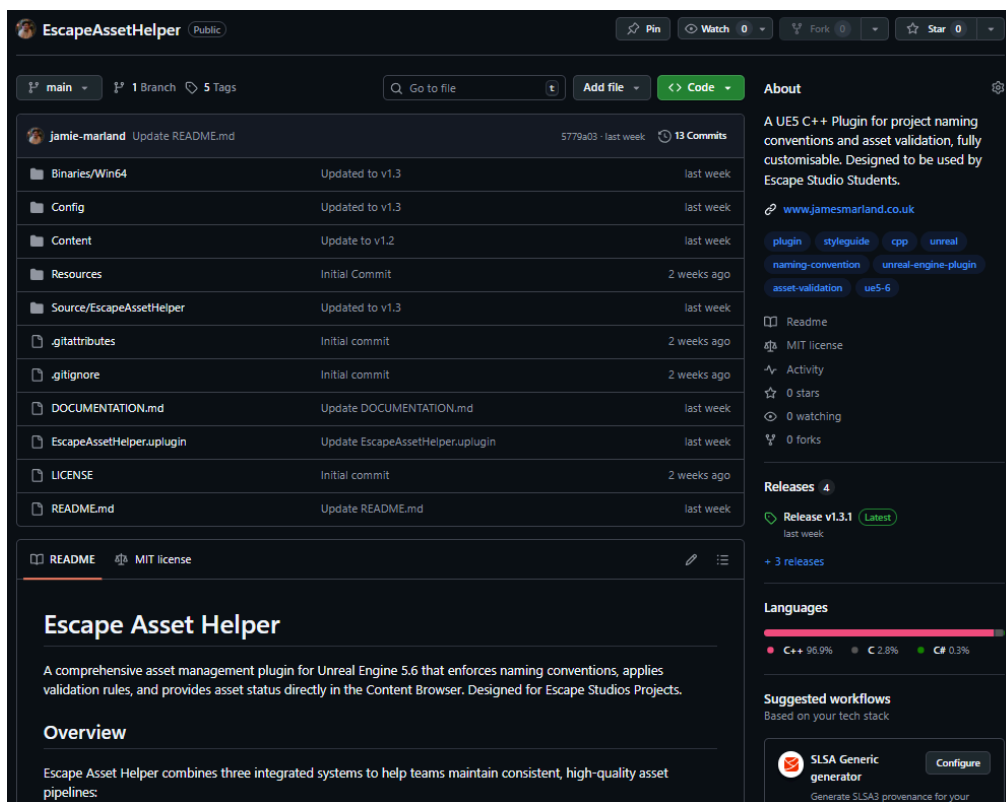
As part of prepping for release, I wanted to have a good default setup for rules, as most people don't change them. I tried to cover the most common cases with Textures Suffixes, setting the texture compression mode and SRGB, and with Static Meshes having Nanite Enabled and Disabled LightMap UVs, and adding a warning for high-polygon assets over 30,000.



Default Rules, Actions and Queries.

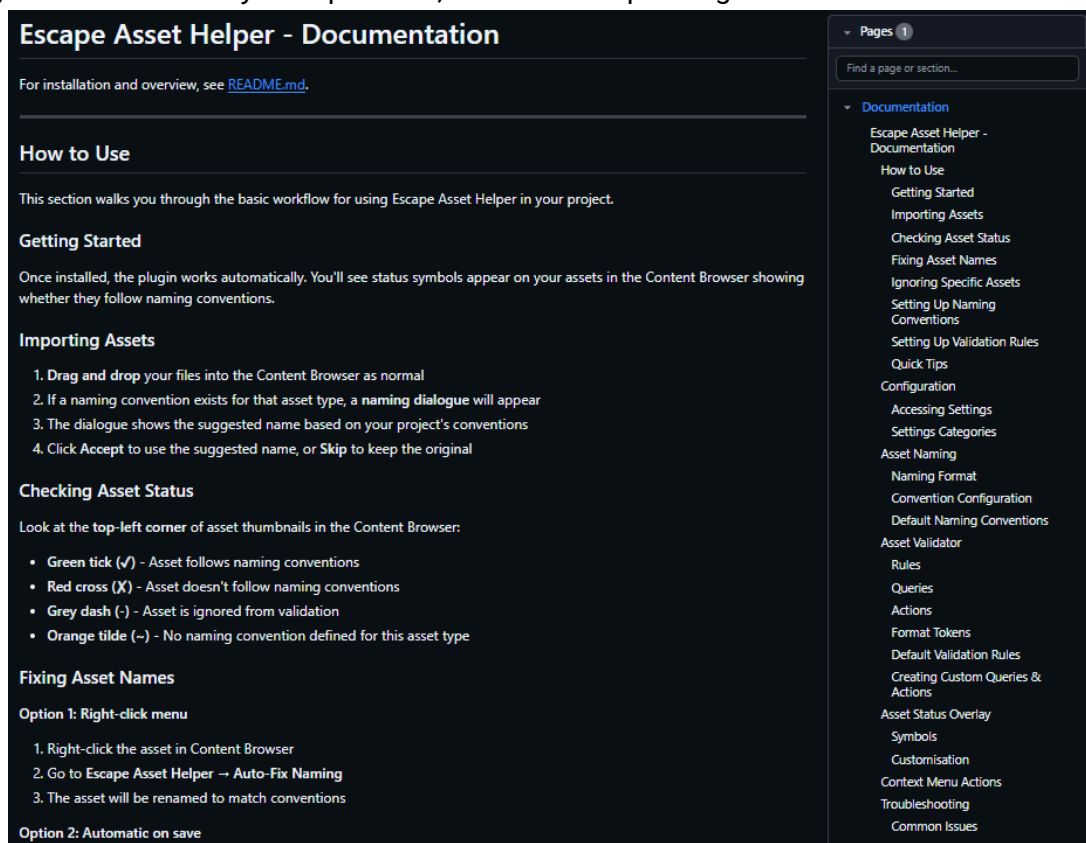
Plugin Release

A major goal this week was to release Escape Asset Helper to the public on GitHub. This involved creating a repository with a README, an installation guide, and proper documentation on how to use and modify the plugin for other teams. This needed to be as easy to install and understand as possible, with clear guidance on what can be customised for non-technical users.



Escape Asset Helper Release on GitHub.

I released v1.0 and then followed up with v1.1, which included the binaries so that Unreal would not request compilation for every user who opened the project, especially on university computers, where compiling from source is not possible.



Escape Asset Helper Documentation.

Animation

To help the animators, I also exported the player's camera location within the model and provided it as a frame of reference for first-person animations.

With Shanti providing more of the player walk animations, I implemented all 4 cardinal directions. It did require turning off IK as it messed with our approach. Lauren also provided the creature's idle animation, and I created a

basic event scare where the creature would flicker in and out. I also made it so the creature could cast shadows even when not visible.



Test scare with mesh shadows when invisible.

Weekly Reflection:

Reflection

This week went very well, as I was able to properly establish the gameplay and release my plugin. This was a big milestone for me. I feel good about where we are in the project. Next week, I will focus on helping any teams that would like to use the plugin and create a sound list for Jamie, an external sound designer, to start making audio for the game.

Problems

I had some issues with source control because Epic Games' documentation was poorly written and hard to find, and I also had to ensure the plugin was in a suitable state for release. Once I sorted this out, it wasn't too bad.

WEEK 7: Making Content

Objective:

In Week 7, I wanted to formally announce the release of Escape Asset Helper on Teams and assist any other groups interested in using it. For my team's project, I also wanted to create the sound list, since we now had an external sound designer.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Epic Developer Community** (2026). Creating a playable build.
<https://forums.unrealengine.com/t/creating-a-playable-build/420143>

Articles/Documentation:

- **Epic Games** (2026a). *Alembic File Importer in Unreal Engine*.
<https://dev.epicgames.com/documentation/en-us/unreal-engine/alembic-file-importer-in-unreal-engine>
- **Epic Games** (2026b). *Quick Start Guide for UCX Collision*.
<https://dev.epicgames.com/community/learning/tutorials/5VqX/quick-start-guide-for-ucx-collision-getting-started-in-unreal-engine-5>
- **Epic Games** (2026c). *Preparing Unreal Engine Projects for Release*.
https://dev.epicgames.com/documentation/en-us/unreal-engine/preparing-unreal-engine-projects-for-release?application_version=5.6

- **Microsoft** (2026). *PIX on Windows - GPU Profiler*.

<https://devblogs.microsoft.com/pix/download/>

Videos/Talks:

- **YouTube** (2026). *Randomising Audio with MetaSounds [UE5]*.
<https://www.youtube.com/watch?v=RLWeioN3LJc>
- **Ben Cloward** (2026). *Simple Terrain Auto Material - Terrain Shaders - Episode 14*. <https://www.youtube.com/watch?v=CGTPW2mWeGQ>
- **Indie Game Clinic** (2026). *Game Mechanics & Systems Thinking*.
<https://www.youtube.com/watch?v=nkLmjJK3v0w>
- **Robin-Yann Storm** (2026). *Gyms, Zoos, and Museums: Your documentation should be in-game*. <https://www.youtube.com/watch?v=5PJRCz0t7yY>
- **Hugh Chew** (2026). *Texel Density Explained*.
<https://www.youtube.com/watch?v=vehIyWK0a1o>

Research Analysis:

This week's research focused either on updating Escape Asset Helper to extend the subsystem logic or on helping teammates with any problems or questions they raised. Covering randomising sounds at runtime, profiling methods, UXC collision, glass shaders, and Alembic. Alongside this, I also reviewed broader research on tech art and game design. This covered texel density, in-game documentation such as zoos, system thinking for game design, and continued Ben Cloward's series.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. Unfortunately, I was very busy and wasn't able to properly reflect on it with cards this week.

Plugin Release

First and foremost, I released the tool and made it known to other teams. The stylised beat-em-up piqued interest, and Denzel, in particular, wanted to use it.

Before anyone had downloaded it, though, I wanted to change one final aspect. Currently, using the Unreal Importer Rules approach, Custom Queries and Actions are first created as Editor Utility Blueprints and then reparented to the custom classes, which is not intuitive. Not only that, but if you use a subsystem inside, it will appear to work okay until you click Play, at which point it says it failed to compile and doesn't have access. Ignoring this error will let it play fine, but it can be scary and annoying for users.

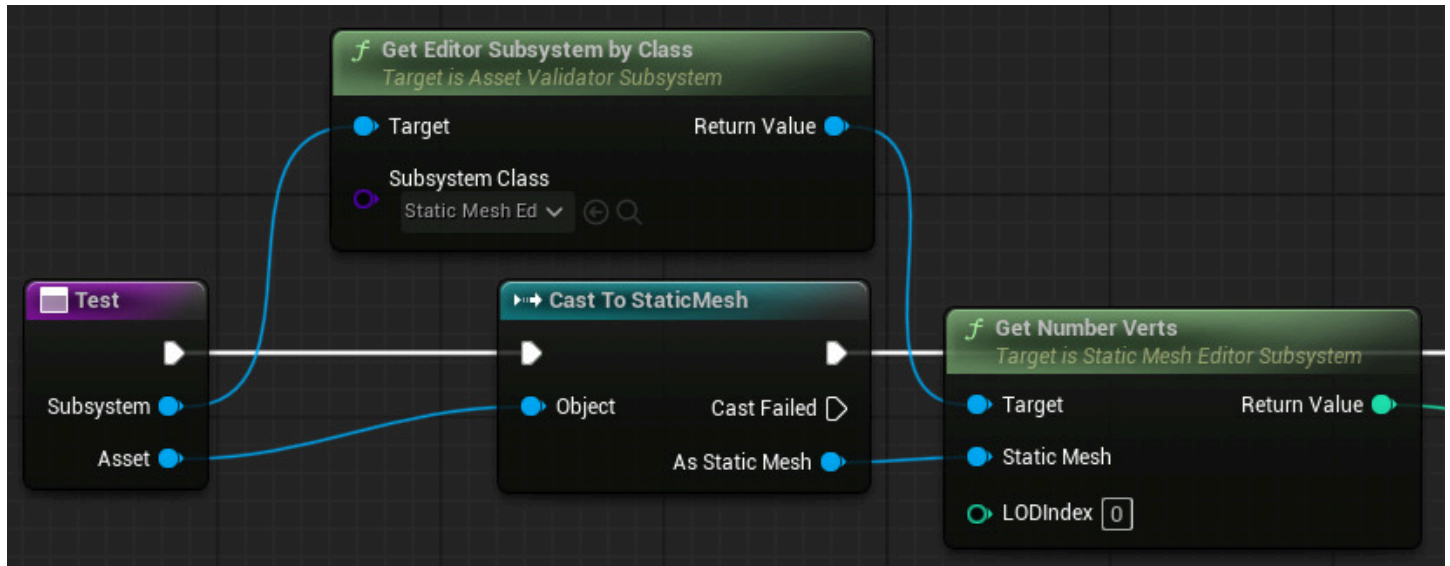
```
// Get Subsystem user selected if used
if (UAssetHandlingSubsystem* HandlingSubsystem = GEditor->GetEditorSubsystem<UAssetHandlingSubsystem>())
{
    if (HandlingSubsystem->IsAssetIgnored(Asset))
    {
        return Result;
    }
}

UEditorSubsystem* UAssetValidatorSubsystem::GetEditorSubsystemByClass(TSubclassOf<UEditorSubsystem> SubsystemClass) const
{
    if (!SubsystemClass || !GEditor)
    {
        return nullptr;
    }

    return GEditor->GetEditorSubsystemBase(SubsystemClass);
}
```

Escape Asset helper Subsystem extension C++ code.

To avoid this, I decided to redesign how my plugin handles subsystems. Now I have a custom node called Get Editor Subsystem by Class that can be fed a generic subsystem and assigned any subsystem in the project. The function already has an input variable with this generic, so now any subsystem is sent out in whatever type the user asks for, outside the context of Editor utilities.



Custom Blueprint Node for getting Subsystems.

This timed well with Denzel, who was facing a minor issue with the tool requesting compilation. I realised I had two mistakes in the release build. One was "Installed": true, which needed to be set in the .uplugin to tell Unreal it doesn't need to be recompiled, and the .gitignore was removing the .dll from the compiled binaries. Submitting a new version with these fixes, Denzel was able to install it properly, and it was received positively. I also updated the documentation to reflect all changes in this version.

Release v1.3.1 LatestCompare   jamie-marland released this last week · 3 commits to main since this release  v1.3.1  67b5c84 

Escape Asset Helper v1.3.1

A comprehensive asset management plugin for Unreal Engine 5.6+ that enforces naming conventions, applies validation rules, and provides visual feedback directly in the Content Browser.

Release v1.3.1 Changelog

- Added "Allow Extended Base Names" setting - validates underscore-separated names like `BP_Event_Creature_MirrorScare` where each segment is PascalCase.
- Added case-only rename warning - shows a toast notification when auto-fix performs a capitalisation-only rename with source control enabled.
- Fixed overlay not updating when settings change - toggling settings now refreshes Content Browser thumbnails immediately.
- Blueprint-only project compatibility - release builds now include `Installed: true` for projects without a C++ compiler.

Installation

- Go to the bottom and download EscapeAssetHelperV1.3.1.zip, and extract it.

	EscapeAssetHelperV1.3.1	28/01/2026 16:25	File folder	
	EscapeAssetHelperV1.3.1.zip	28/01/2026 16:35	Compressed (zipp...	15,364 KB

- Copy the `EscapeAssetHelper` folder to your project's `Plugins` directory. You may need to make the folder!

OS (C:) > Users > MarlandROG > Downloads > UnrealProject >				
Name	Date modified	Type	Size	
 Config	28/01/2026 16:10	File folder		

Release 1.3.1 for Escape Asset Helper on GitHub.

Sound List

Since I will be implementing sounds in the project as a technical artist, I needed to have a say in the sound list. So I created a comprehensive table and catalogued any sounds I thought the team would need, such as footsteps across different floor types and variations, creature noises, voicemail, and more, resulting in 80+ rows. I also added the following columns to the table for each sound: Category, Looping, Modulate, In 3D space, Length, and Trigger, all on top of the ordinary asset information, like status and assigned to. I then sent this to our external sound designer, Jamie (the Other one!), with instructions and a file location to put them in so he could get started.

Category	Base Name	Variant	Priority	Looping	Modulate	3D Pos	Len (s)	Trigger	Status	Current Stage	Assigned To	Notes	3rd Party Asset
Player	FootstepWood	01,02,03	1.Very High	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Creaky, hollow	
Player	FootstepConcrete	01,02,03	2.High	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Solid, cold imp	
Player	FootstepCarpet	01,02,03	2.High	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Muffled thuds	
Player	FootstepTile	01,02,03	2.High	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Sharp, hard cl	
Player	FootstepTunnel	01,02,03	2.High	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Gravel/Squelc	
Player	FootstepStairsWood	01,02,03	3.Medium	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Rhythmic woo	
Player	FootstepStairsConcrete	Concrete	3.Medium	No	Yes	Yes	1	AnimNotify	Not Started	NIA	A. Name	Solid riser imp	
Player	ClothingRustle	01,02,03	3.Medium	Yes	Yes	No	1	AnimNotify	Not Started	NIA	A. Name	Subtle "swish"	
Player	Breathing	Normal	1.Very High	Yes	Yes	No	4	Blueprint	Not Started	NIA	A. Name	Idle, steady 2f	
Player	Breathing	Stressed	2.High	Yes	Yes	No	4	Blueprint	Not Started	NIA	A. Name	Heavier exhal	
Player	Breathing	Gasp	2.High	Yes	Yes	No	4	Blueprint	Not Started	NIA	A. Name	Hyperventilat	
Player	Heartbeat	Low	2.High	Yes	No	No	2	Blueprint	Not Started	NIA	A. Name	Muffled rhyth	
Player	GrunLight	01,02,03	2.High	No	Yes	No	2	AnimNotify	Not Started	NIA	A. Name	Light grunt	
Player	GrunHeavy	01,02,03	3.Medium	No	Yes	No	2	AnimNotify	Not Started	NIA	A. Name	Heavy Grunt	
Cinematic	CouchWakeUp		1.Very High	No	No	Yes	6	AnimNotify	Not Started	NIA	A. Name	Start of game;	
Creature	VocalSobbing	Close	1.Very High	Yes	Yes	Yes	3	Ambient	Not Started	NIA	Lauren Chan	human-like bu	
Creature	VocalSobbing	Distant	3.Medium	Yes	Yes	Yes	3	Ambient	Not Started	NIA	Lauren Chan	human-like bu	
Creature	VocalScream		2.High	No	No	Yes	5	Blueprint	Not Started	NIA	Lauren Chan	High-frequen	
Creature	VocalHeavyBreathing	Normal	2.High	Yes	Yes	Yes	6	Ambient	Not Started	NIA	Lauren Chan	Raspy, wet, an	
Creature	VocalHeavyBreathing	Angry	2.High	Yes	Yes	Yes	6	Ambient	Completed	Made Sound	Lauren Chan	Raspy, wet, an	
Creature	VocalHis		4.Low	No	No	Yes	2	Blueprint	Not Started	NIA	Lauren Chan	Short, aggress	
Creature	VocalDistortedWhisper		4.Low	Yes	Yes	Yes	4	Ambient	Not Started	NIA	Lauren Chan	Indecipherabl	
Creature	FootstepCreature	Thud	2.High	No	Yes	Yes	2	AnimNotify	Not Started	NIA	A. Name	Sudden, heav	
Creature	FootstepCreature	Dragging	2.High	No	Yes	Yes	2	AnimNotify	Not Started	NIA	A. Name	Sudden, heav	
Creature	WallScramble		4.Low	No	No	Yes	3	Blueprint	Not Started	NIA	A. Name	Sound of som	
Creature	NeckSnap		3.Medium	No	No	Yes	1	AnimNotify	Not Started	NIA	A. Name	The "Bone cra	
Environment	InfantCry		2.High	Yes	No	Yes	8	Ambient	Not Started	NIA	A. Name	Distant, muffle	
Environment	LowGrowl		3.Medium	No	Yes	Yes	4	Blueprint	Not Started	NIA	A. Name	Sub-bass voc	
Environment	DoorRattle		3.Medium	No	Yes	Yes	4	Blueprint	Not Started	NIA	A. Name	When the crea	
Envrionm...	HouseSettling	01,02,03	3.Medium	No	Yes	Yes	2	Ambient	Not Started	NIA	A. Name	Randomly trig	
Envrionm...	Wind	Light	2.High	Yes	Yes	Yes	10	Ambient	Completed	Implemented	James Marland	Heard near wi	
Envrionm...	Wind	Heavy	2.High	Yes	Yes	Yes	10	Ambient	Completed	Implemented	James Marland	Heard near wi	
Envrionm...	TVAudio		2.High	Yes	No	Yes	60	Ambient	Not Started	NIA	James Marland	Muffled news	
Envrionm...	TVStatic	Light	1.Very High	Yes	No	Yes	4	Ambient	Not Started	NIA	A. Name	Becomes loud	
Envrionm...	TV Static	Heavy	3.Medium	Yes	No	Yes	4	Ambient	Not Started	NIA	A. Name	Becomes loud	
Envrionm...	LightHum	Fluorescent	1.Very High	Yes	Yes	Yes	3	Ambient	Not Started	NIA	A. Name	Garage uses f	
Envrionm...	LightHum	Warm	3.Medium	Yes	Yes	Yes	3	Ambient	Not Started	NIA	A. Name	Garage uses f	
Envrionm...	ClockTicking		2.High	Yes	No	Yes	4	Ambient	Not Started	NIA	A. Name	Steady, rhyth	
Envrionm...	Fireplace	Crackle	2.High	Yes	Yes	Yes	6	Ambient	Not Started	NIA	A. Name	Comforting w	
Cinematic	Fireplace	Roar	4.Low	Yes	Yes	Yes	6	Ambient	Not Started	NIA	A. Name	changes durin	
Cinematic	WaterPipeSpray		4.Low	Yes	No	Yes	6	Blueprint	Not Started	NIA	A. Name	High-pressure	
Envrionm...	LeakyFaucet		3.Medium	No	Yes	Yes	3	Ambient	Not Started	NIA	A. Name	Persistent rhy	

Sound List for all sounds and variations to be tracked.

Gameplay

For gameplay this week, I added two new example scares and helped figure out what was wrong with Shanti's diagonal walk-direction animations. Once fixed, I imported these. I also addressed some general gameplay bug fixes, collision, the TV video, and some lights.

Weekly Reflection:

Reflection

This week felt really good, as I was able to provide both my team and the stylised beat-em-up with my plugin, and I received positive feedback on its use. I also worked a lot on helping others and made sure people were set up with their respective areas, as is quite common for technical artists. The gameplay is really coming together now, too, and a basic experience is in place. I feel like we're in a good place.

Problems

Shanti and I encountered many problems with the diagonal export because it required an extra controller to copy the animations quickly and flip them. Eventually, we fixed it, and I was able to import them. I also had some minor difficulties figuring out why the plugin wasn't working for Denze, but I was able to fix them relatively quickly. Aside from that, there weren't any major problems halting progress this week.

WEEK 8: Catchup and Cleanup

Objective:

The objective for week 8 was to catch up on my production logs, as I was too busy in previous weeks to work on them. I also wanted to help my teammates with their roles, as it is a core aspect of tech art. I also wanted to improve the flashlight's visuals, fix some lumen light leaking, and tweak the altar's position and size.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Epic Developer Community** (2026). *Lumen scene black radiance probes / Reflections.*
<https://forums.unrealengine.com/t/lumen-scene-black-radiance-probes-reflections/2649947>
- **Epic Developer Community** (2026). *How to fix extremely dark Lumen Interior scenes.*
<https://forums.unrealengine.com/t/how-to-fix-extremely-dark-lumen-interior-scenes/2263003>
- **David Gruwier** (2026) *DGruwier Flashlight Textures*
<https://dgruwier.gumroad.com/l/dgruwier-flashlights>

Articles/Documentation:

- **Epic Games** (2026). *Before Rendering and Occlusion - An In-Depth Look at Real-Time Rendering.*
<https://dev.epicgames.com/community/learning/courses/EGR/unreal-engine-an-in-depth-look-at-real-time-rendering/p91/before-rendering-and-occlusion>
- **Coenen, S.** (2026). *DOOM Eternal - Graphics Study.*
<https://simoncoenen.com/blog/programming/graphics/DoomEternalStudy>

Videos/Talks:

- **Unreal Engine** (2026). *Lighting in UE5: Scaling for Quality & Performance | Unreal Fest Bali 2025.*
https://www.youtube.com/watch?v=Q1whHlGJB_o
- **SimonDev** (2026). *How to optimise (almost) anything.*
<https://www.youtube.com/watch?v=phbaxNPJxss>
- **Ben Cloward** (2026). *Terrain Altitude Blending - Terrain Shaders - Episode 15.*
<https://www.youtube.com/watch?v=MC95bu9VTJQ>
- **Vertex School** (2026). *Becoming a Technical Artist for Games | Live with William Harvey.*
<https://www.youtube.com/watch?v=mEDp2Ce-jHM>

Research Analysis:

This week's research primarily focused on identifying the best solution to address lumen light leakage and on theorising optimisation techniques to implement as the project becomes more intensive. I also watched some general technical art videos to learn more in-depth about the DOOM Eternal graphics pipeline, a talk by William Harvey about the process of being a technical artist, and, as always, continued Ben Cloward's series.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. I did not have the chance to reflect this in the cards on YouTrack.

Flashlight System

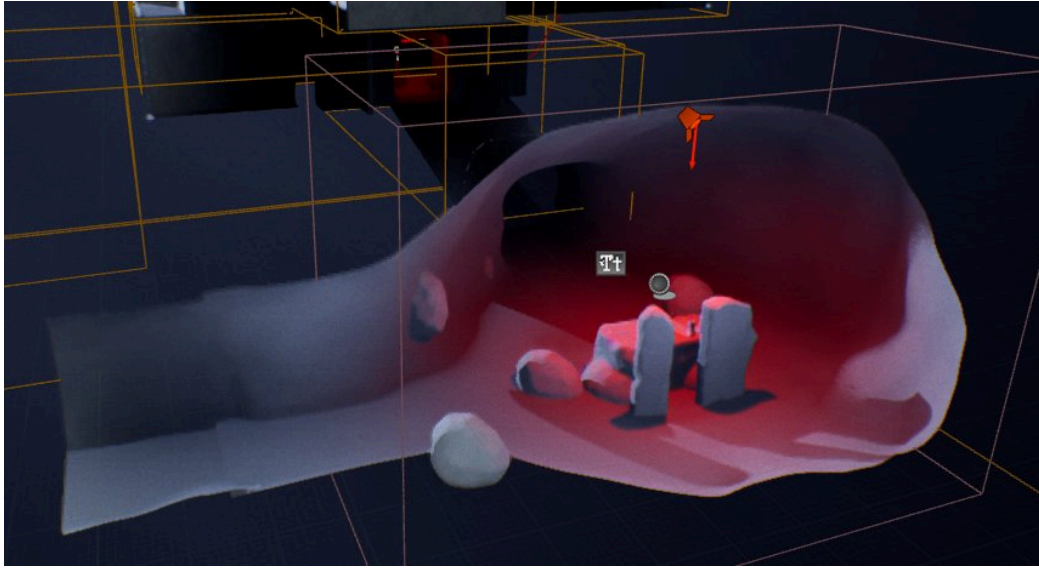
This week, I spent some time refining the flashlight mechanics. First, I made a proper light-function material for the light. I used a flashlight mask with a texture sourced from David Gruwier's Flashlight textures, which I chose specifically because it has actual colour details for some slight refraction, while most online ones are grayscale. I then adjusted the lighting intensity to match and updated the cone radius and position. I also fixed the issue where the player's head movement would go crazy at low fps due to large delta times. I also took the time, based on playtester feedback, to make the flashlight face where the mouse was pointing rather than off to the side.



New flashlight texture and a follow-mouse system with improved light positioning.

Level Tweaks

I added in Adam's latest tunnel meshes and moved the tunnel area closer to the basement to consolidate the layout. The aim was to reduce the amount of work required without wasting any previously made work. It also helped the flow of the level, as the player could clearly see where to head next after opening the door.



New consolidated tunnel section.

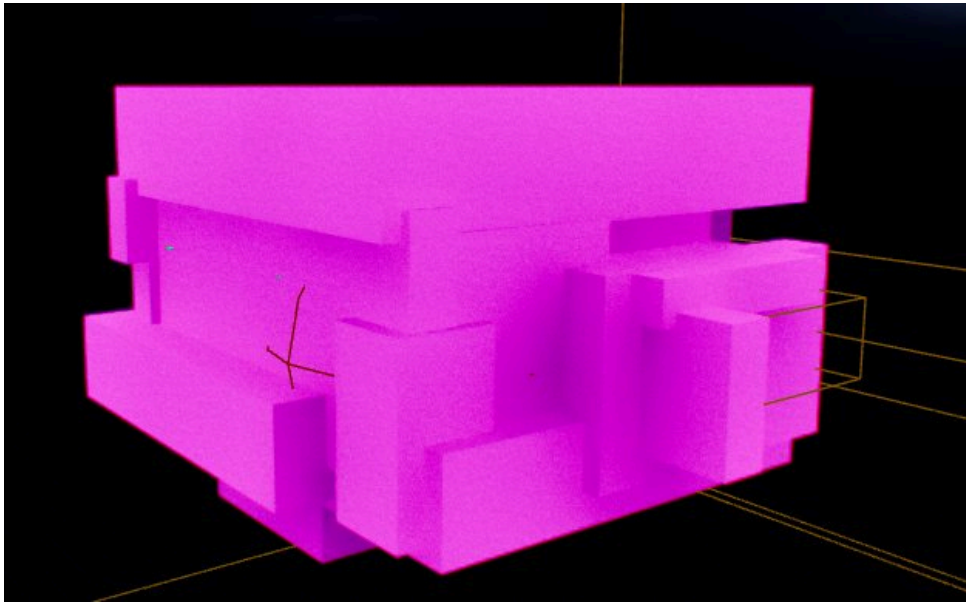
I also fixed various collisions and created a basic glass material that can be built upon later, providing something for the flashlight to reflect off. I then made an Art Zoo level for artists to add finished props to as references.



Assets collected for reference in the Art Zoo.

Lumen Fixes

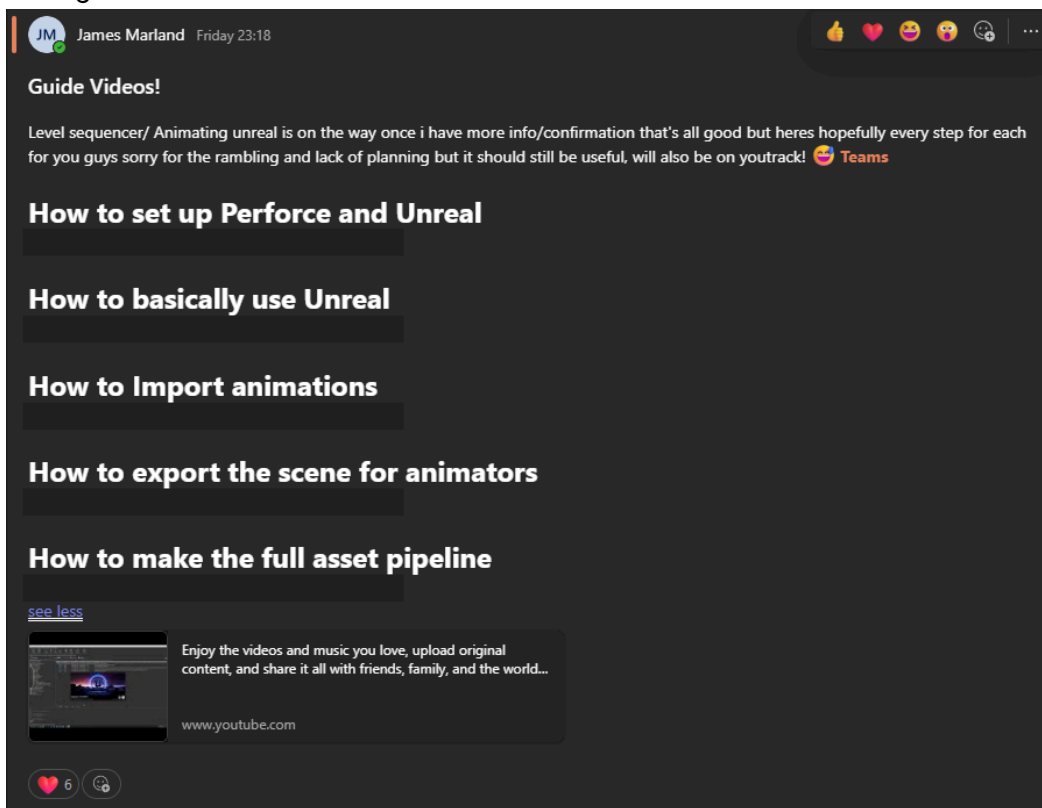
As the project has progressed, this week has focused on tackling some of Lumens' challenges. While I still have much more to look into, I wanted to focus on the biggest issue: the light probes leaking through the thin walls, even with Charlotte extending them out. Annoyingly, there's no easier method, but I resolved it by creating large masked material boxes across the entire exterior with 0 opacity. This stopped probes from being placed outside the walls. I also took the time to enable SSAO on top of Lumen, as I think it helps capture many small details that Lumen tends to blur, and Matty agreed. The lighting is in a much better visual position, but low performance is still causing strong temporal ghosting. This is something I would like to profile and fix next week.



Lumen Light Leaking Volumes that carefully wrap the exterior walls.

Video Guides

Finally, this week I worked on creating some guides for the rest of my team. Many members of my team needed help, and I needed to avoid repeating myself so they could refer back to it. I recorded videos on various subjects to clarify the best approach and workflow for team collaboration on the project. I only did this after I had already gone through it with various team members, so I had an understanding of what I should mention and explain, which helped a lot. The subjects were setting up Perforce and Unreal, covering the basics of using Unreal, importing animations, exporting assets for animators, and a full end-to-end asset pipeline breakdown with tips and best practices. This should help bring everyone onto the same level, even those who are new. Lauren was able to follow the guide and import the animations despite not having worked with Unreal before or having it set up. They were well received. I put off a guide to creating events/cinematics since I want to focus on that next week, after I have talked it through with Michael and Lewis.



My team posts after sending out the Video Guides.

Weekly Reflection:

Reflection

This week, I spent time catching up on my production log and fixing various smaller issues rather than working on higher-level systems. I helped many of my teammates this week by solving various problems they faced, creating guides, and fixing bugs. Improving the flashlight system was a very useful addition, as it really helped convey the game's atmosphere, and fixing the light leaks improved visual clarity. I was able to save my team a fair bit of time by creating the project-specific guides, and I hope they will remain useful throughout the project. Next week, I really want to focus on getting the best guidance on meta-human characters, animating in Unreal Engine, and handling the technical aspects of cinematography. I also now have the new creature model from Amy, so with that all in place, animators should be able to start working on cinematics and scare events by the end of next week. I still feel we're on track.

Problems

The main problem was identifying the source of lumen issues and determining the best way to resolve them. It took a while to figure out that the crude masked-cube approach was the best way to resolve it, since most other approaches didn't work. I finally took a second to fix the player character's violent head movement and set up the flashlight system's rotation.

WEEK 9: Pivoting Development

Objective:

Week 9 was a defining week as we implemented some core changes. The original objective of the week was to find the best approach to handling the cinematic systems by talking to both Michael and Lewis. I gained a much better understanding through these conversations. I also, based on this week's feedback, had a big conversation with the team about where we were in development and what changes we should implement to achieve our goal. Alongside this, I worked on fixing some graphical bugs and, based on this week's feedback, implemented a Look At condition in the event system.

Research:

This week's resources used/researched:

Forum/Blogs:

- **Reddit r/unrealengine** (2026). *How can I detect if a player is "staring" at something for a few seconds, which then triggers it?*
https://www.reddit.com/r/unrealengine/comments/z0ywhq/how_can_i_detect_if_a_player_is_staring_at/
- **Epic Developer Community** (2026). *Checking if the player is looking in a specific direction?*
<https://forums.unrealengine.com/t/checking-if-player-is-looking-in-a-specific-direction/423325>
- **Epic Developer Community** (2026). *MetaHuman Clothing in 5.6.*
<https://forums.unrealengine.com/t/metahuman-clothing-in-in-5-6/2541589>

Articles/Documentation:

- **Epic Developer Community** (2026). *Getting Started with MetaHuman Creator.*
<https://dev.epicgames.com/documentation/en-us/metahuman/getting-started-with-metahuman-creator>

- **Epic Developer Community** (2026). *Custom Clothing for MetaHuman 2.0 in UE5.6 - Step-by-Step*.
<https://dev.epicgames.com/community/learning/tutorials/b6EZ/custom-clothing-for-metahuman-2-0-in-ue5-6-step-by-step>
- **GroundZero** (2026). *The Hidden Cost of Unreal Engine Defaults: What TSR, Lumen, and Shadows Are Really Doing*.
<https://medium.com/@GroundZero/the-hidden-cost-of-unreal-engine-defaults-part-1-199bd591e858>
- **Quilez, I** (2026). *computer graphics, maths, shaders, fractals, demoscenes*.
<https://iquilezles.org/>

Videos/Talks:

- **Hugh Chew** (2026). *Texture Optimisation*.
<https://www.youtube.com/watch?v=QqlZYGEJz90>
- **Unreal Engine** (2026). *The Technical Journey of Clair Obscur: Expedition 33 | Unreal Fest Stockholm 2025*.
<https://www.youtube.com/watch?v=t1CrbTx606w>
- **Fryer, L** (2026). *When Fun Was the Mission at Xbox*.
https://www.youtube.com/watch?v=Is_pbJUkdmI
- **Martin Donald** (2026). *Retro graphics, Cloverpit addiction, authentic PlayStation vertex wobble*.
https://www.youtube.com/watch?v=_HBIC0aFepY
- **Unreal Engine** (2026). *UE5 Constraints Are EASY! Parent Constraint Workflow for Animators*.
https://www.youtube.com/watch?v=LHK3J5m_43c

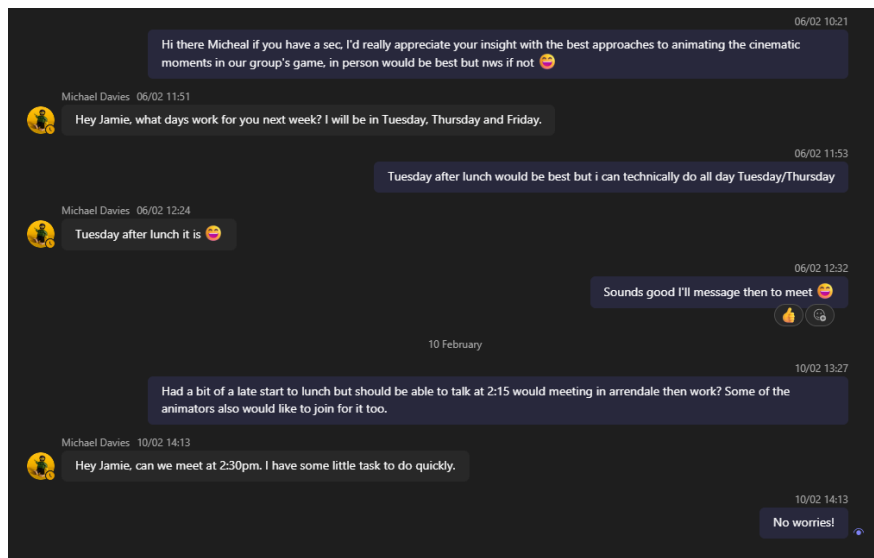
Research Analysis:

This week's research was heavily focused on MetaHumans, as we aimed to use them for the player head, but it ended up being about adding MetaHuman characters entirely. I went through a fair amount of documentation and video guides to understand how to add, adjust, and manipulate the MetaHumans system for our project. One clear limitation was that it still required setting up the characters' clothes, which were not available. I also searched for guidance on the best approach to implementing the Look At system and animating in the level sequence so the animators can take over. Alongside this work, I also conducted my general technical research, including a breakdown of the technical pipeline developed by the team behind Clair Obscur, topics such as texture compression, vertex wobble, UE5 Technical Debt, and articles by other technical artists.

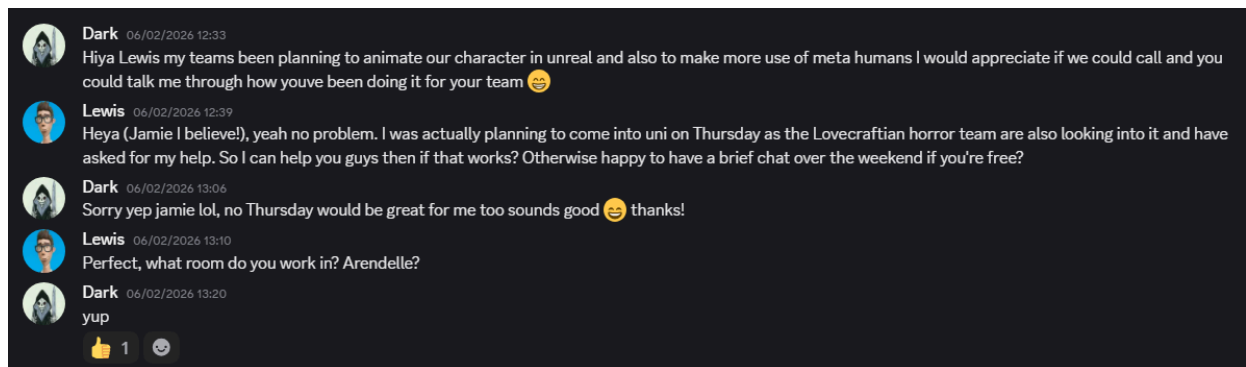
Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This was a particularly big week for producer decisions. I met with Michael about the best way to approach animations in Unreal Engine, and with Lewis about adding MetaHumans.

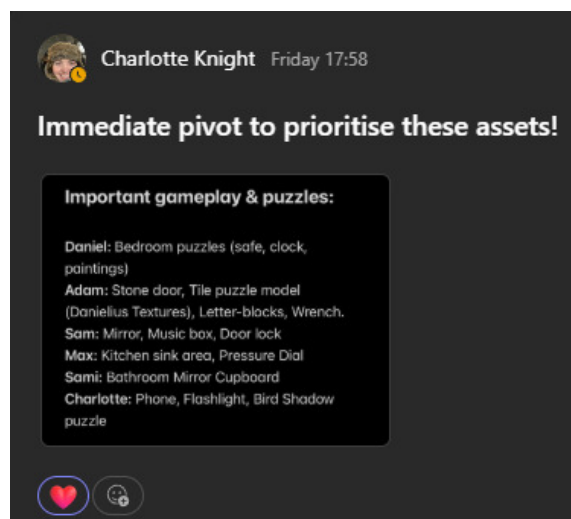


Meeting with Michael.



Meeting with Lewis.

On Friday, after research and clarification with the people it would affect, we had a big team talk where we decided to pivot to full MetaHuman characters, which Amy would continue to create and still work on clothing. We also decided to flesh out the scene with Megascan assets until we have the final props. From the weekly feedback, there was a clear need to prioritise the puzzle and item assets. With my guidance, Charlotte organised every game artist to focus on getting through them all first. At the same time, one person would work on fleshing out the entire map with generic Megascans assets, which I recommended be assigned to Charlotte, as she was already in charge of art direction. I also sent Matty a fire FX pack he needed as a visual reference.

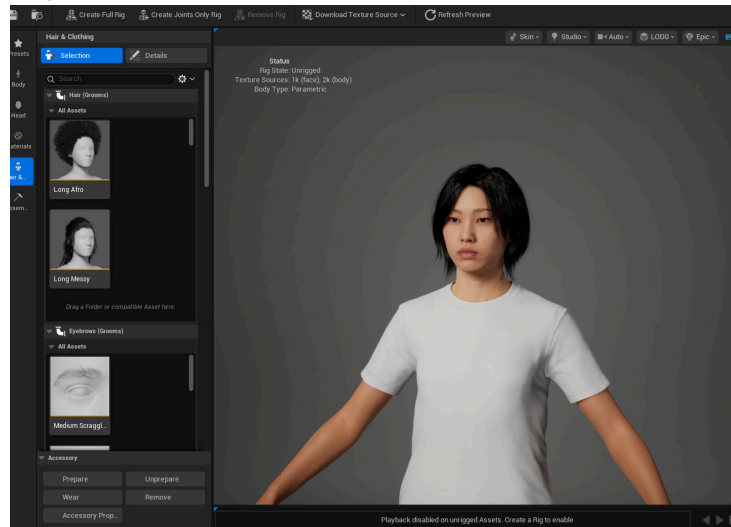


Charlotte delegation of assets based on the call.

The deadline for completing this transition would be the 6th of March, as this was our next Industry showcase. If we could make these changes in the meantime, we would be in a very good position, with a baseline version of the entire game in place.

MetaHuman Integration

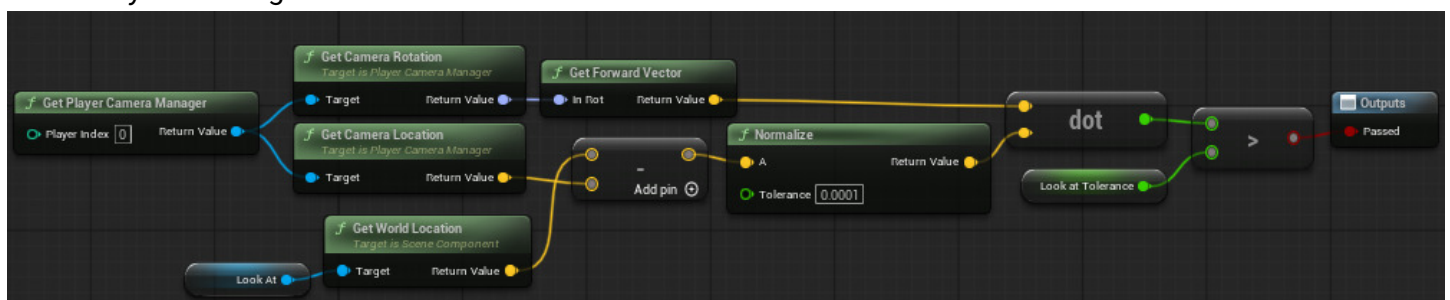
After the team decided to pivot to MetaHumans, I spent time researching how best to integrate them into the project and spoke with Lewis from Outlast Trails, who had a similar approach. I enabled MetaHumans in the project so Amy could start using them as soon as possible. This would mean Lauren could get started much more quickly on animating the creature and would also enable Amy to work on more of the character at the same time. Importantly, the rig proportions needed to be set up. One of the biggest challenges was clothing, since MetaHuman doesn't have an official clothing library. I forwarded the most useful resources to Amy and ensured they were happy with the decisions.



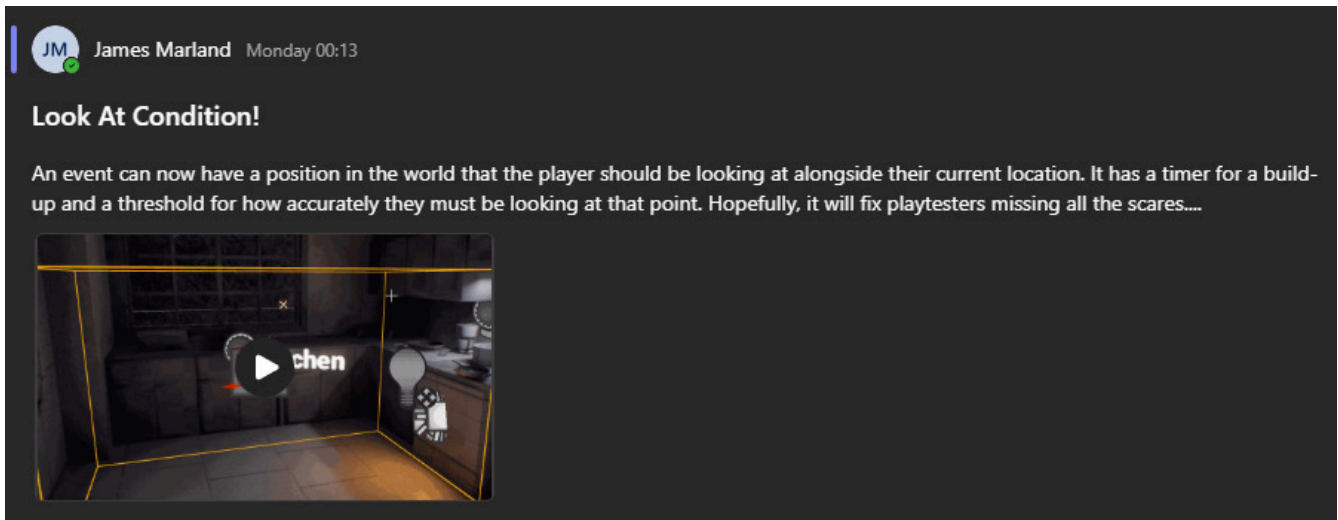
MetaHuman setup in the project for Amy.

Gameplay Feedback

This week's playtest drove a lot of the decisions of this week, and gameplay was a part of that. One of the first things they actually needed was missing. Player, despite being in the triggers, wouldn't always look during a scare, so I implemented a Look At condition for them that used a Dot Product check to see if they were looking at a predefined point. This was customisable with a threshold and duration. This was intended to increase the likelihood that the player was actually looking at it.



Dot product check for if the player is looking.



Team post and video showing it in action.

We also decided to remove the inventory slot system as it seemed it was causing more confusion to player than anything else and instead opted for a global state that meant that the use of items was automatic as long as they had it, alongside what the puzzle and items props the game artists were making it became clear that all the puzzle concepts needed to be made and item pickups needed clear obvious animation to show the player they had it. Kanye compiled the feedback, which was useful for these decisions. I also fixed some bugs that caused the MRQ to break when the gamemode changed.

Weekly Reflection:

Reflection

This week was a pivotal point as we clarified as a team and made some hard but important decisions to keep the project on track and ready for the next industry showcase. The decision to switch to full MetaHuman characters was difficult because it required scrapping much of Amy's old work, but they agreed it was best to bring the project to a complete state within our timeframe. The same goes for the decision to use Megascan assets; it was ultimately the right decision and would allow us to focus on the most important assets first. The Look At Condition was a satisfying addition since it has an immediate positive effect. Next week, I want to focus on getting the MetaHuman character working as soon as possible, helping Danielius get his Houdini work in, and sorting both Danielius's and Matty's problems with importing simulations using VATs. I think the pivots were necessary to stay on track and will enable us to produce a high-quality level of work for the industry showcase.

Problems

The main problems this week were keeping the project within our remaining timeframe and making some hard decisions; luckily, as a team, we were able to work around them and stay on track. I also encountered a few minor bugs, such as MRQ breaking due to an invalid gamemod, but these were fairly trivial to fix.

WEEK 10: Player Revamp

Objective:

The objective for Week 10 was focused on the player. Specifically, I wanted to transfer them to a MetaHuman with the correct skeleton, properly fix the post-process foot-placement animation, and handle general bug fixes and improvements, particularly around the flashlight.

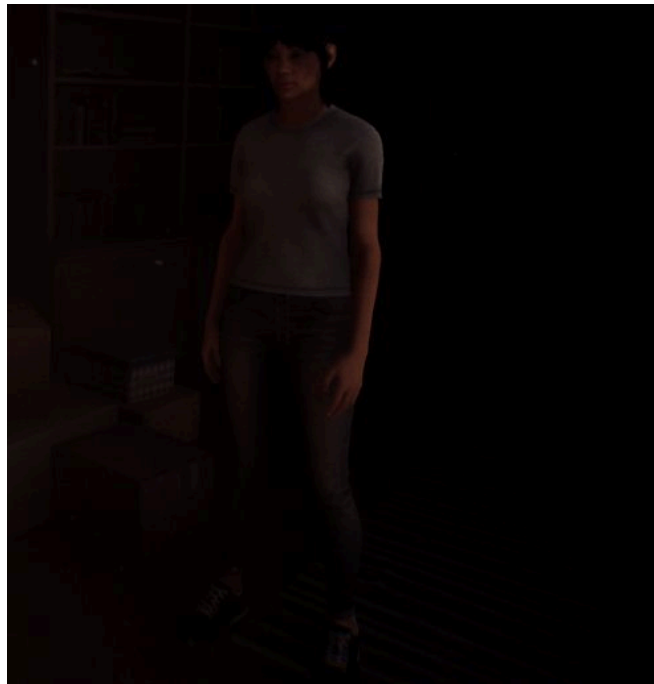
Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included creating and providing the weekly build and fixing general and game-breaking bugs.

Player Revamp

This week, I focused on updating the player model to use the new MetaHuman setup. This required many small steps and checks to ensure the player behaved and animated as intended after the change, since many underlying systems had to be ported. I made sure to set it up so the player could take the updated meshes directly from Amy's export location, so they didn't need to be manually replaced every time a change was made. I also took it as an opportunity to re-enable the post-process foot-placement IK that had been disabled early on to get the walk animations working correctly. Once I revisited it, it wasn't a difficult fix.



The player character is now a metahuman.

I also updated the flashlight model to the new one and added a fake fog beam emanating from it. While it looks okay, I think it still needs more work, and I aim to get around to it again before the end of development.

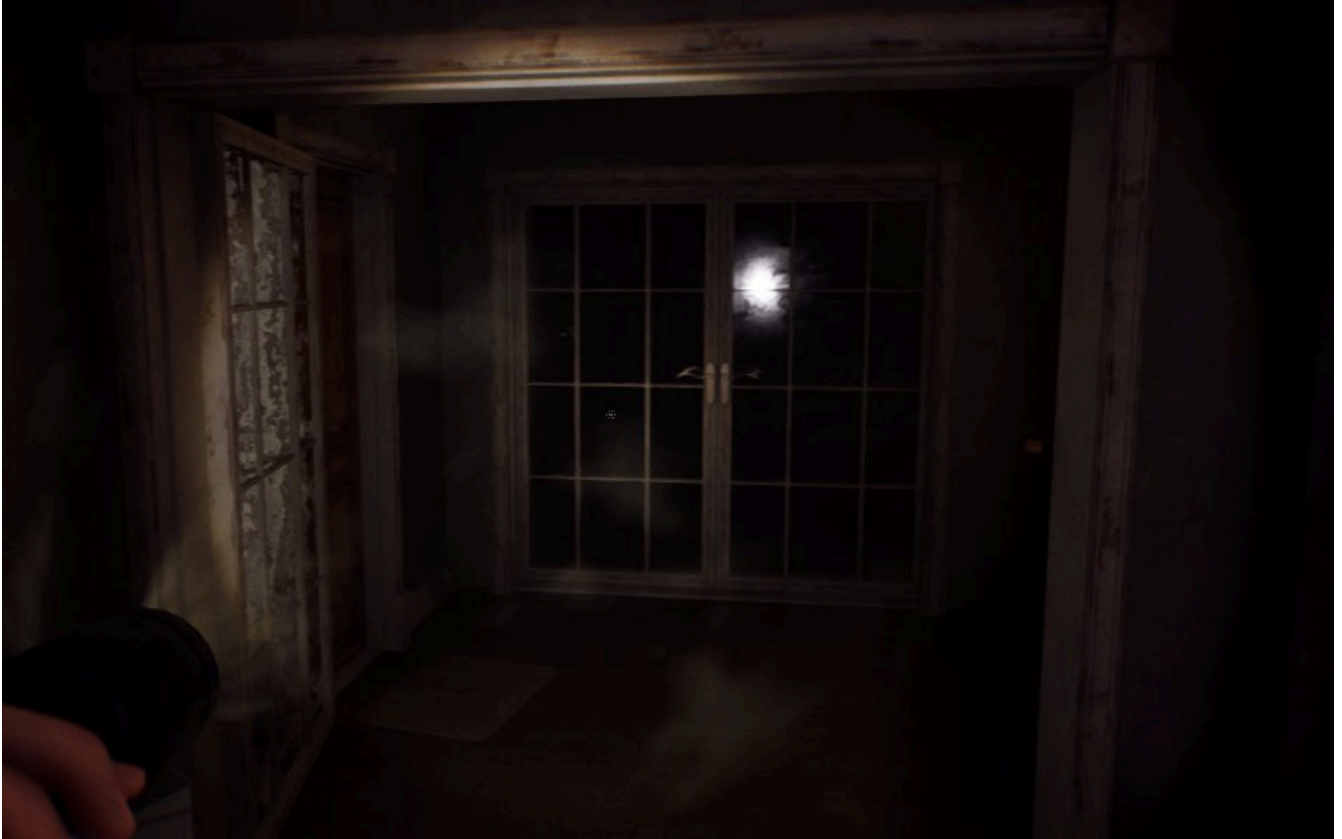


The new flashlight shader

Bug fixing

Across the week, I also worked on fixing a range of bugs. First, the house map now loads on startup rather than when Play is pressed, and to avoid the player

falling through the floor during loading, I added a persistent collision plane below spawn. I also did a pass of Lumen blockers and shadow blockers around the house, so that areas like the altar stayed in darkness despite being built from one-sided interior meshes. When the levels are unloaded, these shadow blockers, placed inside the walls and floors, ensure that areas aren't weirdly lit due to the absence of the surrounding building. Lumen blockers are used in places where shadows still need to pass through, but thin meshes need something to stop probes from leaking through. Finally, I fixed a bug where a light with a soft radius of 0 would create a white sphere visible only in reflections; I resolved it by increasing the radius.



New doors with open logic

General Improvements

Some other improvements I made during the week were updating the loading screen UI, adding cockroach and fly models for Matty's VFX, setting up a CRT material for the TV, and adding a basic outdoor environment to create silhouettes during lightning strikes and to hide the altar from the kitchen view.



Outside environment

I also flipped the hinge direction on some doors around the house and reworked the opening and closing animations for single and double doors so they take longer and aren't linear, adding to the sense of realism. The basement door now slams more forcefully, indicating its supernatural cause. I added a listening callback event for when doors finish opening, for situations where another

blueprint needs to know. I finished by connecting up more lights that didn't have a switch assigned and improving the subtitles' visuals and text.

Weekly Reflection:

Reflection

This week was a lot of work, as I made sure the project was pivoting correctly and kept on top of everyone's deadlines with Charlotte to keep the scope manageable. It was also tough to swap the player character, as many small functions were affected in subtle ways, and it took a lot of debugging, with Shanti's help in spotting issues, to fix them all. Once done, though, it was very rewarding. I was able to cover a lot in the week, and I still feel confident about our timeframe. Next week, I plan to work on refining gameplay as much as possible to get out of the alpha stage, as we have an important showcase coming up.

Problems

The main problems this week were related to metahuman, such as retargeting animations, migrating bones for the skeletons, and fixing mesh detail settings. It was a lot of pain, but I was ultimately able to solve it.

Reading: Break

WEEK 11: Refinements

Objective:

Week 11 was focused on polishing gameplay and setting up profiling, so that in the following weeks the project could be shown to an external group of optimisation-focused technical artists I had organised a session with, as well as to our industry mentor. Most of the work was on bug fixing, general improvements, profiling tools, and producer tasks.

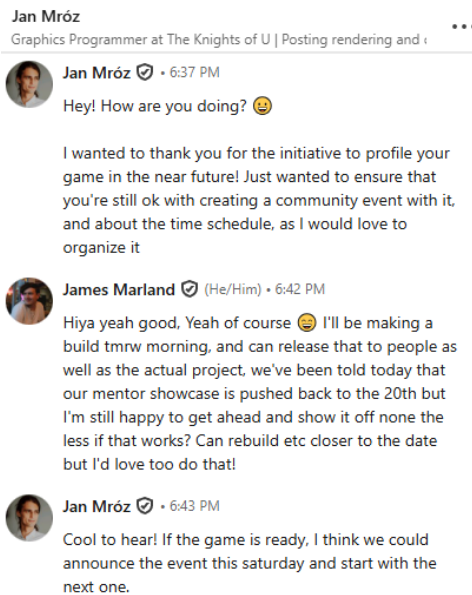
Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included making and providing the weekly build and fixing general and game-breaking bugs. I helped Shanti with importing and retargeting animations the new way, and with using the Level Sequencer, especially for animating the MetaHuman rig. I also explained to Lauren the basics of using the event system so she could create her own scares. I provided a Fab Suburbs asset pack to Charlotte and Matty for set dressing. At the end of the week, we had our tutor showcase, which I prepared for and presented.

External Profiling

Outside of my studies, I participate in an optimisation community organised by a graphics programmer called Jan Mroz. As part of our regular calls, I brought up my university group project and received positive responses from people who wanted to try it. Jan and I arranged for him to present our game to the community and run an optimisation challenge around it, where members could discuss and share what they found. This was greatly appreciated and a really exciting thing to organise.

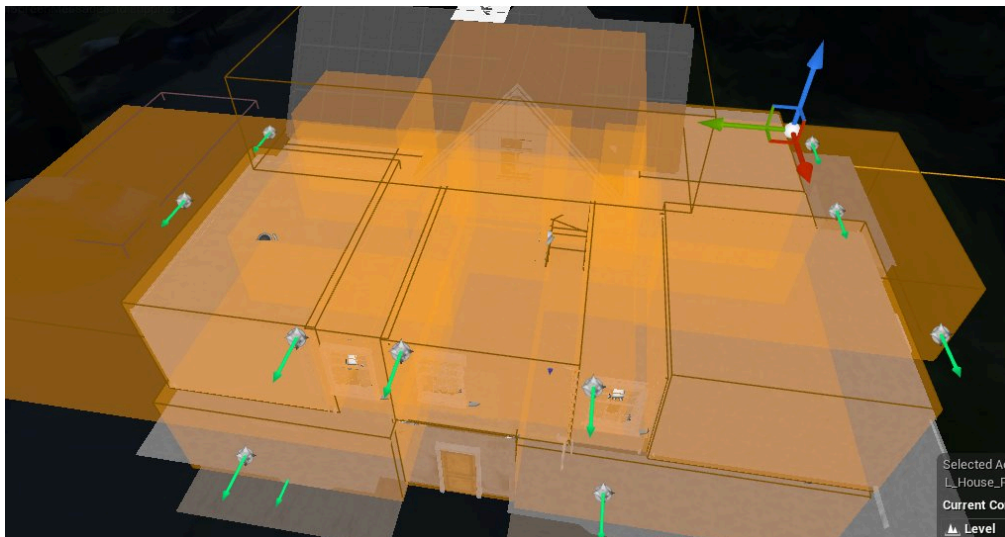


Discussion with Jan

In preparation for next week's call, I sent Jan a build of the game with a performance-free camm mode I made, which lets the community gather deterministic data points. With Jan's feedback, I expanded on this further by adding a locked-off camera view and toggleable flashlight and player character, so anyone profiling could isolate variables more consistently.

Bug fixing

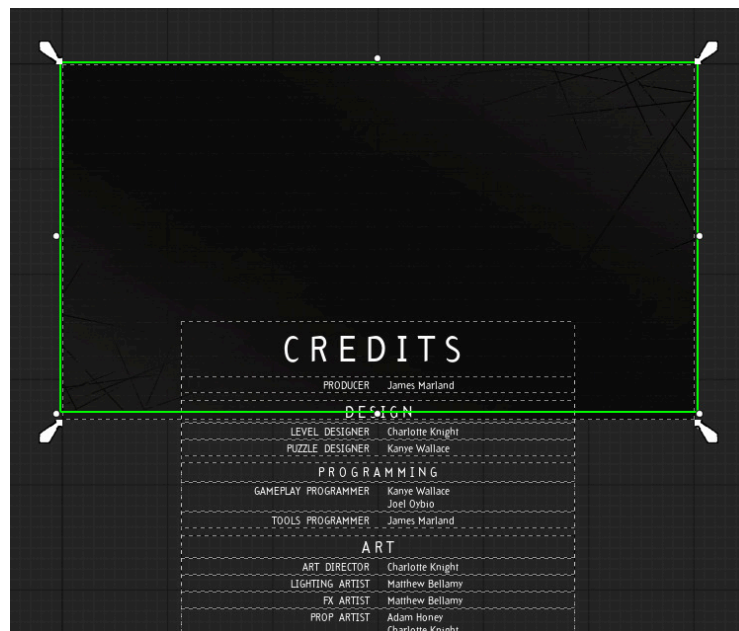
In preparation for the build I was sending to Jan, I aimed to fix as many bugs as I could and polish the game. The fixes I implemented were: the flashlight's scale is now consistent between pickup and held states, fog appears only when the flashlight is on, emissives on light meshes were lowered, various collisions were fixed, and softlocks exiting certain puzzles were resolved. I also changed all Level Streaming Volumes to use non-complex shapes, which stops the player from falling through unloaded levels. I made them bigger and overlapping, and in some cases, segmented them into multiple smaller parts to reduce the total loading overhead in certain rooms.



New updated LSV boundaries.

General Improvements

I also worked on some general improvements, like using temporary textures for the doors, kitchen counters, and altar room walls, so the space felt less grey while final art caught up. For assets that had their new model and textures ready, I updated them in-engine, including the letterblocks, tiles, fusebox, flashlight, and stone door. I also created a detailed credit list, breaking down every role and who worked on it.



New scrolling credits

Weekly Reflection:

Reflection

This week was interesting as I was able to organise my own external meeting with people in the industry. It would mainly be about the game performance as an example, and seeing what could be improved, but it was still good to show off. My plans for next week will continue to revolve around making sure the project is polished for this meeting and that the tools are set up for deterministic profiling. I think we're still on a good track; it's mainly on the asset front that we need to have all puzzle assets in, which takes time.

Problems

I didn't face any real problems this week any bugs I encountered I was able to solve easily.

WEEK 12: External Profiling

Objective:

Week 12 again focused on ensuring gameplay was polished and profiling was set up, following feedback from Jan Mroz, so that over the weekend, the game could be shown on a call to the entire community, and we could get optimisation-related feedback from industry technical artists. It also had the side benefit that the week after was our industry mentor call, so having the game polished benefited both.

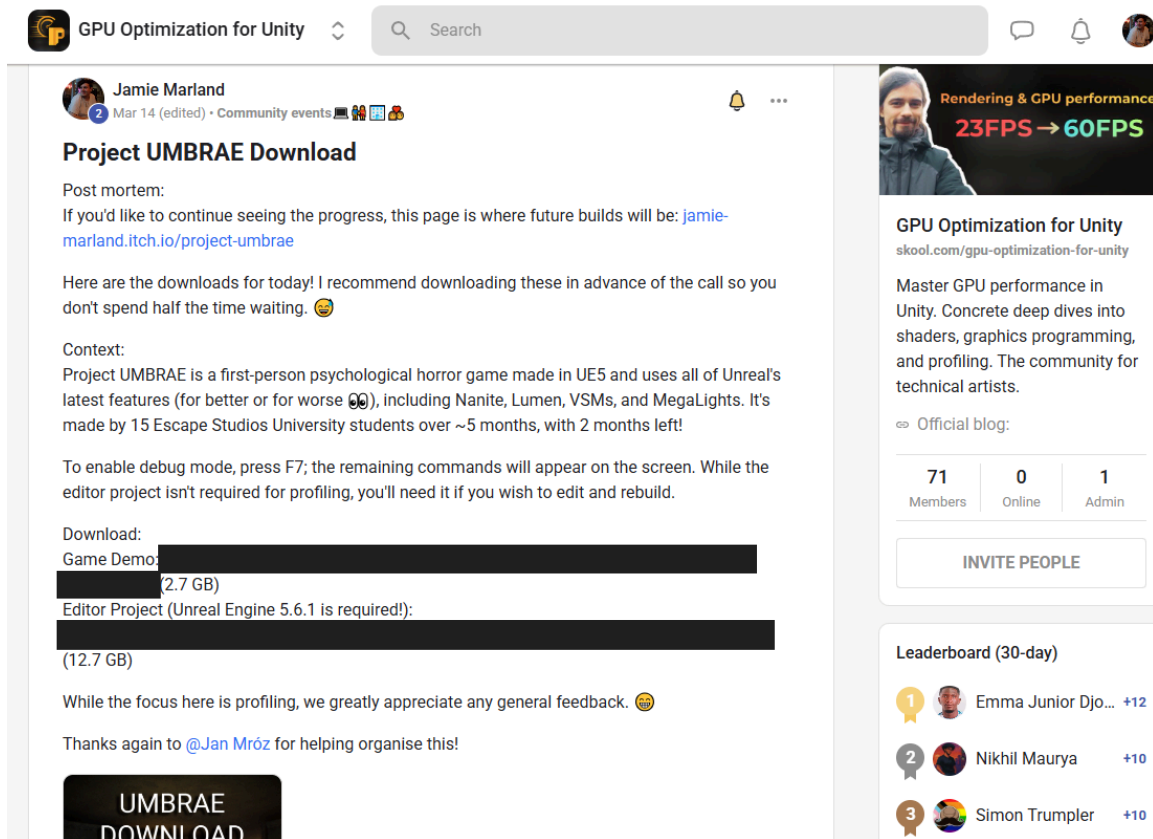
Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included creating and providing the weekly build, and fixing general and game-breaking bugs in preparation for sending it out to Jan's community over the weekend.

External Profiling

The main external focus this week was the build going to Jan Mroz's community. I uploaded both the shipping build and the editor project to Google Drive, so members could not only play the game with the deterministic free-cam mode I had set up the previous week, but also open the editor directly to take Unreal Insights and GPU captures. Jan and I made a community post about the project, framing it as an optimisation challenge, and had a session to show it off and see what initial findings people had. In week 13, we would do a call to discuss what people found.



Download the post on SKOOL.

Bug fixing

In parallel with the profiling prep, I aimed to clear out as many bugs as I could so the build would represent a fair version of the game to profile. The fixes I implemented were:

- Face rig - Fixed the face rig so it now properly animates the face during cutscenes.
- Start sequence camera - Made the start sequence use the first-person camera. I also helped Lauren set up the equivalent for the menu sequence.
- Item inspect performance - Joel had set up the item inspect using a very expensive render target system, so I worked to improve this with reduced resolution and a dedicated light channel. Also, it's no longer written by every item in the level every frame; only the item being inspected is updated when the mesh moves.
- Construction script captures - Removed old construction script captures from non-inspect item blueprints, which were running unnecessarily.
- Controller support fallout - After controller support was added, the sliding and sink puzzles broke. I fixed the input handling for both.
- Bird puzzle - The bird target rotation is now correct, and each bird piece has collision.
- Build-version casting - Stone door and clock blueprints now retry their player character and gamemode casts in build versions if they fail on the first attempt, since cast timing differs slightly between editor and packaged builds.



This week, I built a unified light-flickering system. Rect, spot, and point light blueprints now share the same underlying setup, and each can be turned on or off either via a switch or by clicking directly. Each light has a "light health" variable that defines how much it flickers and the intensity of the flicker; on top of the flicker, it also produces spark VFX. The light type also affects the audio, so a fluorescent light flickers and hums differently from a bulb. This consolidates much of the per-light ad hoc work scattered across the project into a single component that's quick to drop in and tune.



This week went well. The main focus was getting the build clean enough to hand over to Jan's community without it looking like a mess of unrelated bugs that obscure the real performance picture. Joel's item inspection was the most important fix, as it was causing a major performance problem; the artist couldn't

work on the project. All optimising it made a big difference. The light-flickering overhaul also feels like a meaningful consolidation that the team can now reuse cleanly. Next week, I'll be focused on integrating any feedback from the community call, in addition to preparing for the Karl Wooley showcase. I still feel we're on track.

Problems

The main problem this week was the controller support breaking the sliding and sink puzzles in subtle ways, which took a moment to track down to the input handling. Figuring out that the item inspection render target system was also more involved, as it wasn't clear that it was causing the specific problem to optimise more than I'd hoped, since it touched several systems at once, but each change was straightforward.

WEEK 13: Cutscenes and Showcase

Objective:

Week 13 focused on getting everything in and as polished as possible before the showcase to Karl Wooley, our industry mentor, at the end of the week. As such, my focus was on general fixes and improvements wherever I could find them. On the side, I also received feedback from the external profiling session I had with Jan Mroz's community, and I did my own checks and timing before implementing the easiest fixes.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included making and providing the weekly build and fixing general and game-breaking bugs. I also set up our itch.io page and distributed builds there going forward, which makes it much easier to share consistent versions externally. At the end of the week, we had our presentation with our industry mentor, Karl Wooley. The reception was very positive, aside from one softlock with the inspection system. He liked every aspect of the game and confirmed we were on the right track.

Bug fixes

This week's bug fixes were focused on what would be most visible during the showcase:

- MetaHuman sockets - Fixed the sockets on the MetaHuman and the flashlight position so it sits correctly in the hand.
- Inspecting - Fixed the item inspection system, which had broken since the perf rework last week.
- House lighting - Fixed unconnected lights and rebalanced light values across the house to make the lighting more consistent.
- Gameplay bugs - General gameplay fixes across puzzles and interactions.

General Improvements

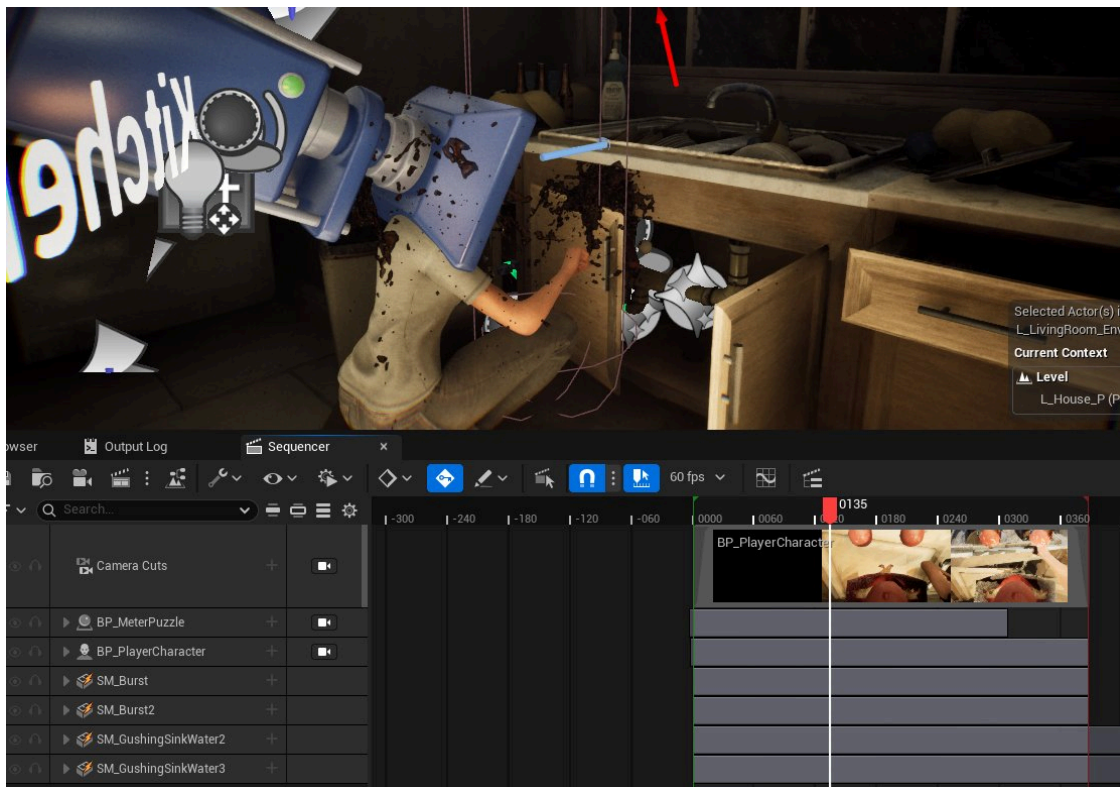
I also worked on several general improvements aimed at the overall feel of the game:

- Voicemail revamp - The voicemail sequence was previously hardcoded to a fixed length. I rewrote it so the length now follows the actual audio, and swapped the placeholder text for a proper recorded audio file.
- Door mesh - Updated to the new version.

- Player speed - Lowered the player speed slightly from 240 to 200, which makes moves feel less arcade-y and more deliberate.
- Volumetric fog - Added volumetric fog throughout the house to make the gameplay feel more immersive and to give the lights and flashlight something to scatter through.
- Fluid VATs - Did VATs from Matty's alembic fluid sims so they could be played back efficiently in-game.

Cutscenes

The bigger system-level addition this week was a centralised Cutscene Manager. Before this, cutscene logic had been handled in a scattered way across different blueprints, with each event managing its own player movement, HUD visibility, and follow-up logic. The Cutscene Manager now handles how the level sequence is executed, disabling player movement, hiding the HUD, swapping cameras as needed, and also defines what code executes once it ends. Most existing setups were ported to use it.



Sink Cutscene in editor

On top of the manager itself, I made and implemented the Menu, Start, Breaker, and Sink burst sequences using it. Each one slots into the manager rather than reinventing the playback logic.

Performance

Following Jan's community feedback and my own profiling, I did a performance pass:

- Nanite - Disabled Nanite, since the project's mesh budget didn't justify the overhead it was adding.
- Light culling - Disabled lights past 1000 units of view distance.
- Flashlight on fog - Stopped the flashlight from affecting volumetric fog, as it was producing a lot of trailing light.
- Hair - Switched hair from strands to cards and lowered the sim quality, then bumped to cinematic hair only inside cutscenes.
- Volumetric fog quality - Lowered the overall volumetric fog quality as well.



Scene with volumetric

Weekly Reflection:

Reflection

This week went really well. The Karl Wooley showcase landed positively, with only one softlock to point at; it confirmed and boosted our confidence in the direction the project has been heading. The Cutscene Manager felt like a long-overdue piece of consolidation, and getting Menu, Start, Breaker, and Sink all running through it cleanly was satisfying. The performance pass also made the game noticeably smoother, which was the main thing I wanted to get out of Jan's community feedback. Next week, I want to focus on immersion, like footsteps audio passes, and on pulling more interactions into the game. I feel we're in a really strong position.

Problems

The inspect softlock that Karl noticed is the one issue I want to get to the bottom of properly next week. Otherwise, no major problems; most of this week's fixes were straightforward to track down once I had time to sit on each one.

WEEK 14: Immersion

Objective:

Week 14, with the Karl Wooley showcase out of the way, the goal was to focus on immersion and interaction with the environment and events. My focus was on the things that pull a player into the world without necessarily being noticed: footsteps and surface audio, polished cutscene transitions, more meaningful interactions across the house, and finishing off the cutscene state-handling work I had started in Week 13. On top of that, the usual sweep of bug fixes and improvements to keep the build clean.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included creating and providing the weekly build and fixing general and game-breaking bugs.

Bug fixes

This week's bug fixes were a mix of cutscene polish and inspection issues:

- Furnace sequence - Fixed the furnace sequence, still letting the player move during it.
- Sliding tiles - Fixed the sliding tiles drawing getting stuck.
- Inspectables scale - Fixed inspectables being tiny and not in perspective, since the items themselves weren't actually being scaled for inspection.
- Voicemail - Lowered the voicemail flash intensity, which was previously too aggressive.
- Door appearance - Added a temporary fix until the new mesh and material were finalised.
- Light audio fix - Fixed an audio issue where not all lights played audio.
- General gameplay - A range of smaller gameplay fixes across the house.



Inspect item appearance

Improvements

On the improvements side, most of the work was around extending the Cutscene Manager and tightening how the player transitions in and out of sequences:

- Furnace sequence - Built out the furnace sequence itself.
- Lerp to position - Cutscenes now lerp the player to the correct position before starting, so the camera handover is smooth instead of snapping.
- Save player pos - The player's position is now saved after a cutscene, so they end up where the sequence intends, rather than at a default.
- Save interacted state - Interacted object states are also saved after cutscenes, so doors, lights, and items don't reset to default after a sequence plays.
- Improved lerping - Generally improved the player's lerping into and out of sequences.
- Audio drop - Added the remaining audio Jamie had provided up to this point to the project.
- Cutscene flashlight - Hide the flashlight in certain cutscenes where it shouldn't be visible.
- Puzzle player visibility - Hide the player model in all puzzles, since the puzzle cameras could see the body and flashlight clipping into the frame.
- Voicemail draft sound - Added a draft sound for the voicemail.

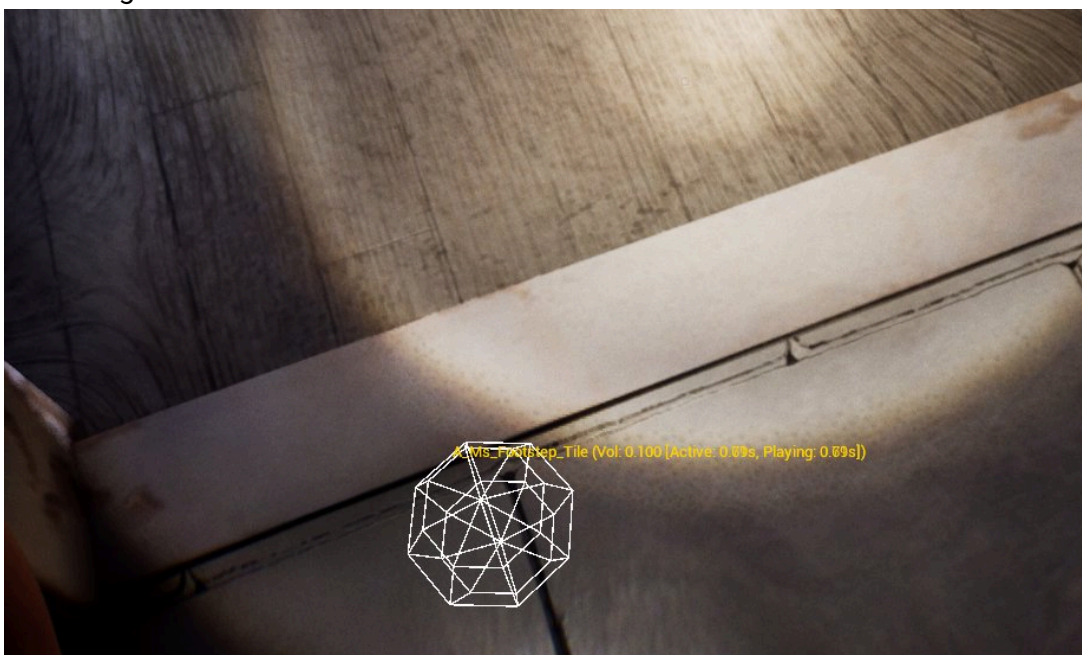


Furnace sequence

Footsteps

A big part of this week's immersion push was footsteps. Up to this point, they were a generic global sound that felt off in a horror game where footsteps carry a lot of weight.

- Spatial sound - Lowered the overall footstep sound volume and made it spatial at the player's feet, so it feels grounded instead of being in your head.
- MetaSounds - Built MetaSounds with random variations for each surface, so back-to-back steps don't sound identical.
- Foot notifiers - Updated the foot notifiers on the walk animations to fire at the correct points.
- Phys materials - Did a pass of physical materials around the house so each surface returns the correct surface type, which the footstep system uses to pick the right sound bank.



Footstep audio in debug view

Interactions

The other major immersion thread was making more of the house interactable, instead of relying on just a few puzzles:

- Lights - Light blueprints can now be directly interactable instead of requiring a switch, so the player can click a lamp itself.
- Projector - Made the projector and have it so it's the main light for the bird puzzle; it's fully interactable now.
- Kitchen sink - The kitchen sink as a whole is now interactable, not just the gauge model or the wrench.
- Interactable TV - Built an interactable TV with dynamic lighting, audio, and a shader effect on the screen, with video feed tracking logic. Hence, it selects the appropriate video to play based on the game state. When the TV is off, it keeps track of where the live feed would be during its duration, so when it's turned back on, it resumes at the correct point. I also tweaked the TV's texture and mesh to suit. It is also now used for the in-game main menu at the start.
- In-game main menu - Built an in-game main menu with a shader effect on the UI and a background video setup; the UI was adjusted to fit the TV screen.
- Clock hands - Added the clock hand item, with cutscenes for picking up from its painting and placing it on the clock, and modified the logic for the clock and picture frames to fit this flow.



Interactable TV with video and VFX

Weekly Reflection:

Reflection

This week felt very productive on the immersion side. Footsteps, in particular, make a big difference; the game already feels noticeably more grounded, with surface-aware steps coming from the player's feet rather than a generic global sound. Extending the Cutscene Manager so it properly saves player and object states was also a meaningful follow-up to last week, since many of the awkward "snap" moments after a cutscene are now gone. The interactive TV with the

live-feed tracking is one of the things I'm most happy with from this week. Next week, I want to focus on the event system overhaul, which has been on my mind for a while and will need a proper block of time to do properly. I still feel we're on track.

Problems

Nothing major this week. Most of the cutscene-related fixes were finicky in isolation, but each one was small. The trickiest piece was making sure the interacted object states actually persisted across the cutscene boundary without conflicting with the existing cutscene logic. Still, once I had a save/restore pattern that worked, it slotted in cleanly.

Holiday: Development

Holiday: EPC2026

WEEK 15: Event Overhaul

Objective:

Week 15 was focused almost entirely on the event system overhaul, which I had been wanting to do for a while. The original event system worked but was rigid, and events couldn't really react to game state, and a lot of behaviour was only accessible in the creature encounters type. The plan for the week was to introduce a proper conditions system that could check player state, milestones, and items; expose those conditions to the Level Sequencer for cinematic use; add staged support so events like the creature encounter could wait while primed; and migrate every existing event to the new system. At the end of the week, with the event work in a good place, I also wanted to get a generic openables system in for drawers, cupboards, and similar.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included creating and providing the weekly build (Build v20) and fixing general and game-breaking bugs as the event migration brought issues to light.

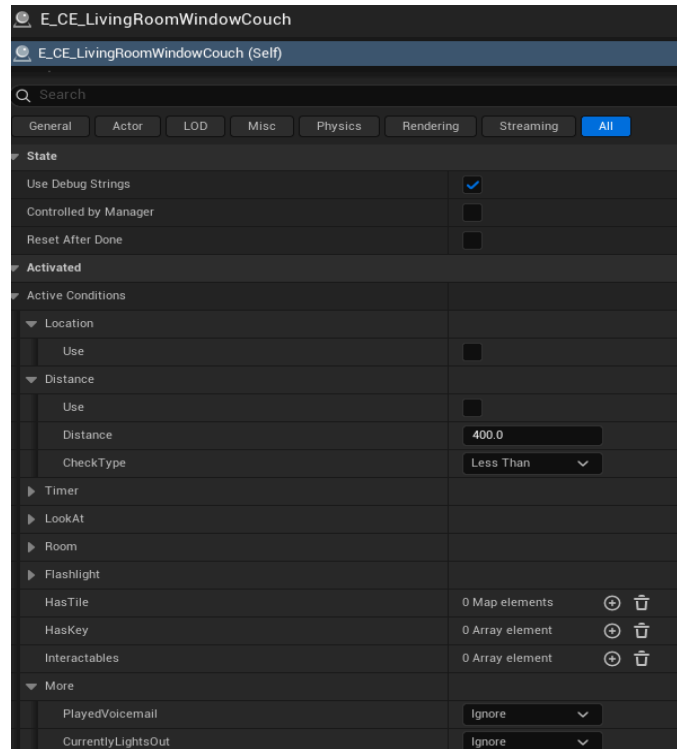
Event system overhaul

This week was the proper time to rebuild the event system. Previously, events could be primed and fired, but they had no real awareness of the game state, and a lot of behaviour was duplicated across event types. The overhaul introduced a proper conditions system and brought all event types onto a common foundation.

The new conditions can be checked both when an event is being primed and when it is about to fire. They include:

- Keys - Whether the player has or used any keys, including puzzle items.
- Tiles - Whether the player has or doesn't have any tiles.
- Interactables - Whether the status of a light or door is on / open, etc., for all interactable objects.

- Milestones - Whether the player has completed a particular milestone in the game, like Lights Out.
- Flashlight - Whether the player has the flashlight, whether it's currently on, and whether it's flickering.
- Timer - Time-based conditions for events that should only fire after a minimum or maximum delay.
- Location - Whether the player is inside a specific location box.
- Distance - Whether a player is less than or greater than a distance from a point.
- Room - Whether the player is or is not in any room.
- The Event Manager can also trigger any primed events that meet their conditions on demand, which is mainly used for triggering audio events at the right moment without each event needing its own logic to listen for state changes.



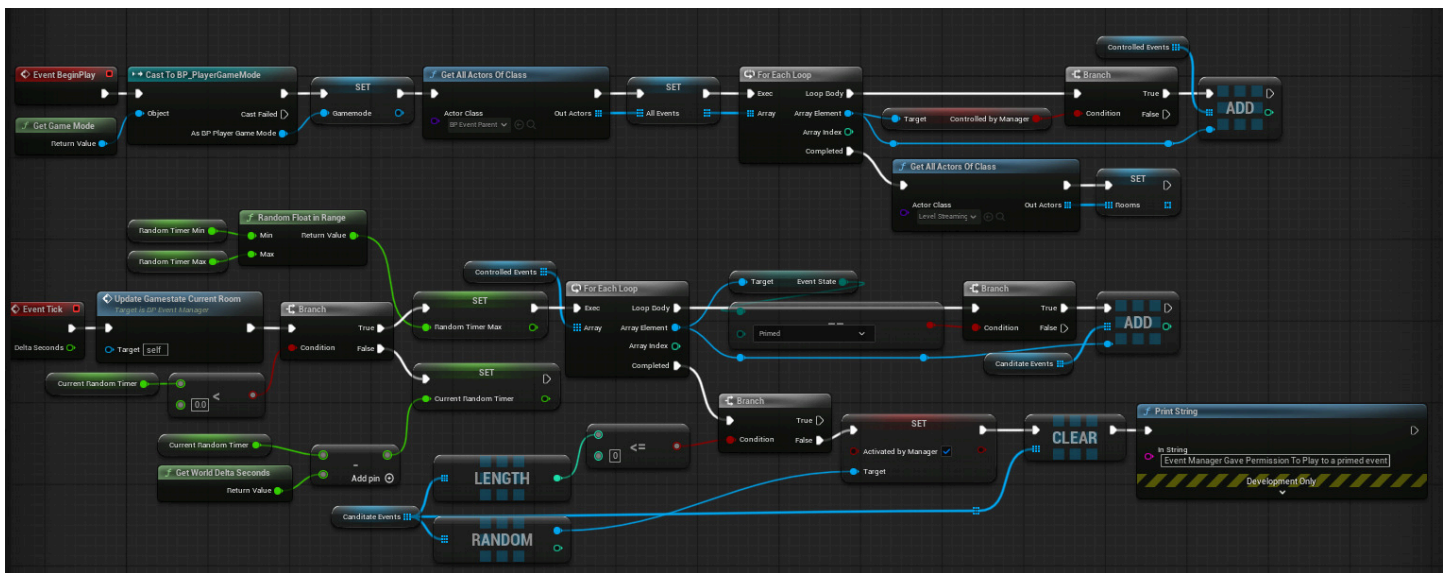
Event conditions to choose from.

Event conditions are configurable for each event at the prime and fire stages. On top of the conditions, all event types: Audio, Visual, Cutscene, and Creature now have full functionality on the new system, where previously only some types were fully wired up. The creature encounter gained staged support: it can sit in a "waiting" state while primed via a level sequence, progressing only when the player activates the condition that advances it.

The Level Sequencer also gained accessible event-system actions, so cinematic work doesn't need bespoke blueprint plumbing. Inside a sequence, I can now:

- Change the player's speed.
- Close doors.
- Trigger lightning.
- Turn the lights off.

The bulk of the migration work was upgrading every existing event to use the new conditions system, which most of the second half of the week was spent on. Old events that were sub-children of a type had to be deleted and their conditions redone manually, since the structure had shifted.



Event manager logic

Bug fixing

Alongside the overhaul, several smaller fixes came out of the migration, touching everything:

- Breaker switch door - Made the door open on the breaker switch event.
- Framegate pause - Added a pause function to the framegate, so it can be held mid-state when needed.
- Basement creature - Fixed the basement main floor creature pausing properly when expected.
- Bathroom blend - Blended the bathroom cinematic so the transition in and out is smooth.
- Clock hands custom mode - Fixed the issue of the player staying in custom mode after placing the clock hands.

Generic Openables System

Towards the end of the week, with events in a stable state, I built a generic openables system. It's a custom static mesh component with extra variables defining the closed and open transforms, and it lerp-s between them when clicked. This gives drawers, cupboards, the fridge, ovens, and similar all the same interactable behaviour without each one needing its own blueprint.



Openable doors and drawers

Weekly Reflection:

Reflection

This week went really well. Finishing the event system overhaul was a big weight off, and the new conditions system already feels far more flexible than before, especially with the Level Sequencer integration. Migrating all existing events was tedious but worth it for consistency. Getting the openables system in at the end was a nice cap on the week, since it adds a lot of small interactable moments around the house for very little ongoing maintenance. I still feel we're on track.

Problems

The main difficulty was the event migration itself: old events that were sub-children of an event type couldn't just be re-parented; they had to be deleted and reconfigured under the new conditions system, which was time-consuming and error-prone. Once each event was converted, though, things behaved more predictably than before.

WEEK 16: Polish

Objective:

Week 16 was a polish week with the tutor showcase coming up the following week. The plan was to clear out a backlog of small visual and audio bugs, get a few outstanding interactables and animations in, fix some longstanding reflection issues, and tighten the project's VRAM usage by limiting non-structural textures. None of the items was large individually, but together they were what was needed to bring the project to a more presentable state heading into showcase week.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included making and providing the weekly build and fixing general and game-breaking bugs as they came up, with the deadline ahead as the main milestone everyone's work was being aligned toward.

Bug Fixing

A run of smaller fixes that had been sitting in the backlog:

- Pink HQ reflections - Fixed the pink colouring showing through on HQ reflections, which was caused by an incorrect reflection mode on the affected materials.
- Dining room light - Fixed the dining room light not turning off when expected; the trigger wasn't propagating through to the light's off state.
- Fly audio - Lowered the volume.
- Subtitle font - Changed the subtitle font for better readability against the in-game backgrounds.

General Improvements

Alongside the fixes, I got a few outstanding animations and interactables in:

- Clock hand animations - Updated the clock hand pickup and place animations so they read more clearly in the puzzle context.
- Code lock animation - Added the dial animation for the code lock so the puzzle has a proper interaction response rather than the value snapping.

- Short table interactable - Made the short table interactable so it slots into the existing interaction logic with the rest of the room dressing.
- Creature cloth - Amy's creature now wears a cloth dress, so the creature reads correctly in encounters rather than appearing as the bare model.



Creature with clothes

Performance

On the performance side, I limited all non-structural textures to a maximum of 2k. The structural textures the level relies on (walls, floors, large fixtures) are still allowed at higher resolution since they cover the most screen area, but all other props were capped in line with the asset guidelines. This frees up VRAM headroom without any visible loss in quality at gameplay view distances from them.

Weekly Reflection:

Reflection

This week was a steady polish pass. Nothing individually was large, but many small things came together to make the project feel noticeably more finished, particularly the animation and interactive additions, which fill in gaps a player would notice immediately. Capping non-structural textures was a small change that recovered a useful amount of VRAM. Next week is the tutor showcase, so my focus will shift to the bigger interactions and cutscenes that still need wiring up and a clean buildout. I feel I'm on track for the showcase.

Problems

No real issues for this week, just lots of little fixes that weren't difficult.

WEEK 17: Final week

Objective:

Week 17 was building toward the tutor showcase at the end of the week. The plan was to get the player's environmental IK working through the PP IK EnviroSense plugin, push out a round of UI improvements with proper world-space interaction icons, finish the bathroom cutscene, including the dirty-to-clean mirror switch, fix the persistent black HQ reflection issue affecting the bathroom and nursery, and make the Hallway B scare more reactive. Alongside that, build prep, asset cleanup, and put together the skeleton of the showcase presentation.

Weekly Summary:

Producer

Throughout the week, I stayed on top of producer tasks by organising the SCRUM Weekly and Daily meetings and clarifying with everyone what each person would be working on. This also included making and providing the weekly build, fixing general and game-breaking bugs, and putting together the skeleton of the showcase presentation to send across to Matty so we could start work on it as soon as the project was finished for Friday, so the structure of the presentation would flow before fleshing it out.

Envirosense

I set up the player character so they were ready to use a plugin called EnviroSense that let them use procedural animation to attach the hand to whatever it was animating. This would allow for a much more immersive player experience. Later in the week, however, I had to scale this down to meet the timeframe I had left on the project and make sure everyone's work was implemented. I attempted to get this working for the flashlight at least; however, this was arguably the most complex to do, and after facing difficulties, I decided to drop it as well. One of the main problems was that the torch system was designed to attach to the player via a socket, but the IK system then tried to move to attack the torch, creating a loop that caused it to spasm out of control. While it was a shame, it was for the best. It's still something the team would like to do for the rookies, though, later on, after submission.



Broken hand IK with torch.

Interaction Icons

Until now, players had to discover what was interactable largely by trial. I added a world-UI interaction icon that appears on every interactable when the player is nearby and looking at it. It also fades gently in and out, which minimises on-screen UI whilst making it clear what is interactable in the scene. This behaviour was implemented using a world-space UI widget and a custom component that can easily be added to any interactable object, so the code didn't need to be remade and required minimal setup. I did a pass of all interactive objects and made sure they would also update when no longer interactive, etc., so it did not become confusing.



In-World Interaction Icons

Notes

Now, with all notes and assets ready, and with Charlotte and Matty's help, I made sure all intended notes were placed around the level and displayed the right text and model, adding a layer of lore to the game for those interested in exploring it. This wasn't difficult, but it was time-consuming to do. It also required fixing some bugs in note inspection, such as scaling and initial rotation.

Improvements

Since this week was really focused on making sure everyone's updated work was in, I focused on that:

- More reactive Hallway B scare - Triggers faster, has lighting flashes during the scare, and slows the player down briefly so the scare actually lands rather than being able to see the monster disappear.
- Wall cabinet interactable - Made the wall cabinet an interactable that triggers the cutscene start.
- Locked bathroom door - The locked bathroom door now opens as part of the cutscene flow rather than blocking it.
- Player position - Fixed the player's positioning during the bathroom cutscene so the camera framing reads correctly.
- Dirty/clean mirror switch - The mirror switches from its dirty state to its clean state during the cutscene, giving the moment its narrative beat.
- Reflection mode swap - Swapped the bathroom's reflection mode in conjunction with the cutscene so the mirror reads correctly through the transition.
- Mirror door bug - Fixed an extra mirror door bug exposed once the cutscene was running end-to-end.
- Stopped assets appearing black in reflection - Small or complex props would not work with the current lumen scene detail, as well as using simpler shader models. This was also done for the nursery.
- Updated in-game credits - Reflect current areas people have worked on
- More general ambience - Windows have the outside ambience and rain hitting when near them.
- Final voicemail - Voice mail sounds are updated and use finished sound effects.
- Dear skull interactable - Can be interacted with to start the final cutscene.

- Shadow bird puzzle updates - Now gives time to see the final position and holds the player there before it starts the furnace sequence, and swaps the painting material once done.

Bug fixing

- Player movement during menu - Prevented the player from moving while the main menu was up.
- Mouse on screen - Fixed the mouse cursor staying visible when the game wasn't on the main menu.
- Asset cleanup - Cleaned up the last of the greybox / blockout meshes and corrected sizing on dressing that hadn't been brought through to final.
- TV heard across the house - Turned down attenuation radius for TV and general volume.
- Thin doors - Update the doors to Sami's new thin model.
- Collisions - General collision updates as people add new assets, most importantly, all the updated altar room meshes.

Weekly Reflection:

Reflection

A heavy week, but a necessary one going into the showcase. While plagued by various problems, I was still able to make some of the most important final additions and changes, as has repeatedly become a theme of making sure other people's work was in the project. The world-UI interaction icons were a small change with a big readability win; players can now see what's interactable at a glance. The bathroom, nursery and final cutscene coming together end-to-end was also a big win. Next week shifts into the final stretch toward submission, especially deliverables outside of the unreal project. I feel I'm on track for submission now.

Problems

While I was able to work through a lot this week, I did face a fair number of issues. The Perforce servers filled up twice this week, forcing us to wait until it was resolved, and I had to compromise and give up on some things I wanted to do, like the Post Process IK for hand interactables.

Conclusion

Project UMBRAE began as a first-person horror game set in American suburbs haunted by the player's aunt, and over 17 weeks of structured development, plus the Christmas and Easter breaks and reading weeks, it became a fully playable demo experience with cinematics, fleshed-out gameplay and systems, a set-dressed environment, and even audio. As Producer and Technical Artist, my role straddled the project's organisational and technical sides, and the weekly entries above capture the trade-offs that came out of that balance, often putting the team's work above my own.

On the producer side, the SCRUM cadence held throughout: weekly and daily meetings, weekly builds, and a steady process of unblocking teammates on tools, plugins, and pipeline issues. Coming together as a team to do a full 3-week-long pivot early saved us and made sure we remained on track, continuously, never fully falling behind on our goals. The reliance on Metahuman, third-party assets, and the team's existing models isn't ideal. Still, it worked and let us deliver on our plans without significant compromise to scale.

On the technical side, the systems that compounded across the rest of the project ended up mattering more than the visible features alone. The event system overhaul, with its conditions and Level Sequencer integration, sits behind almost every scare, audio cue, and puzzle trigger in the level. The Cutscene Manager also made it easy to integrate repeatable behaviour into our project. Creating generic behaviour like lights, footsteps, interactables, icons, and the entire project organisational structure saved countless hours of my development time. I even got to spend time exploring and learning about areas of tech art I hadn't touched or avoided before, like animation and profiling.

I was able to build multiple industry relationships across development, for instance, with our mentor Karl Wooley and in tech art profiling with Jan Mroz on the side. That feedback and positive reception motivated the team and me, as did their insight, which I value greatly.

Releasing the Escape Asset Helper plugin publicly on GitHub was secretly a goal of mine, as it allowed me to show my skills in developing a custom C++ plugin for the first time. While difficult, it was rewarding both to make and to use, and it even helped lead to me being offered a direct internship at Epic Games. It also helped another team, Stylised Beat-em-up, who picked it up to use themselves.

Looking back, the things I would do differently would mainly be about getting a less flashy system done early, so there's less dead time in developing aspects such as openable behaviour, cutscenes, and lights. The lesson I'm taking forward is that systems-level decisions, the ones a lot of other features will lean on, are worth resolving early, even when nothing is immediately pressuring them, because the later they're built, the more rework they cost downstream.

I'm leaving the project feeling that we delivered close to what we set out to make, that the team came through the inevitable rough patches well, and that the systems I built will hold up under the weight of the final level rather than being patches stacked on patches. I'm also leaving with a much clearer sense of where my own balance between Producer and Technical Artist sits, and that's something I'll carry forward into industry.



Beauty Shot 4