DIGITAL ARTS & ENTERTAINMENT

howest
/ we develop people

Internship @ MoonMonster Studios

MOONMONSTER studios

Project offered by

Burgisser Dylan

To obtain the degree of
Bachelor of Digital Arts and Entertainment
Academic Year 2025 - 2026

Introduction

MoonMonster Studios was founded in 2019, located at Hangar K they would start working a year later, on their 6 yearlong project named Space Control. The game I would spend most of my internship working on.

Released on April 1st this year, Space Control is a comedic episodic story game taking the player on crazier and crazier adventures. MoonMonster Studios was the first company I emailed for my internship. During my previous study's internship, I applied at MoonMonster Studios but lacking the required skills for game development I would eventually be refused. What made them stand out is that they would be the only company to reply to my emails and so would be on the top of my list.
Years later during my DAE internship I would apply to many companies and after rounds of interviews with many different companies, Brian the person I interviewed with at MoonMonster seemed to be the person I could learn the most from, solidifying my decision to join them!

During my internship I would work on many different systems, mostly on Space Control but not limited to. Here's a quick Overview of my work which you can find in the rest of this document.

Familie Fit: Monsterbuit

- Easter update
- Implementing the new Easter walk
- General Familie Fit Task
- Fixing some of the backtracking issues of the walk
- Making android build

Space Control: Pre-Launch

- Cutting voice lines
- A lot of playtesting
- Polish Tasks
- Customer Spray playing on all customers
- Collision Improvements
- Slime Shower Audio
- Audio Cue implementation
- Probe strength
- Fixing the Drum Sequence
- Fixing bugs, so, many, bugs.
- Plate Animations
- Implementing various VFX
- Interactive BazookiBooki
- GlebGlub Baby powers
- Fidget Toys:
    - Bobblehead
    - Bouncy Balls
    - Tambourine
    - Xylophone
    - Baby Shaker

    - Fidget Spinners VFX
- Interactive HUB screens

Space Control: Post-Launch

- More bugs
- Hand Tracking
- Cosmetics Persistence + Cosmetic Update
- Turret Minigame

Internship Report 01

Week 01 & 02

Burgisser Dylan

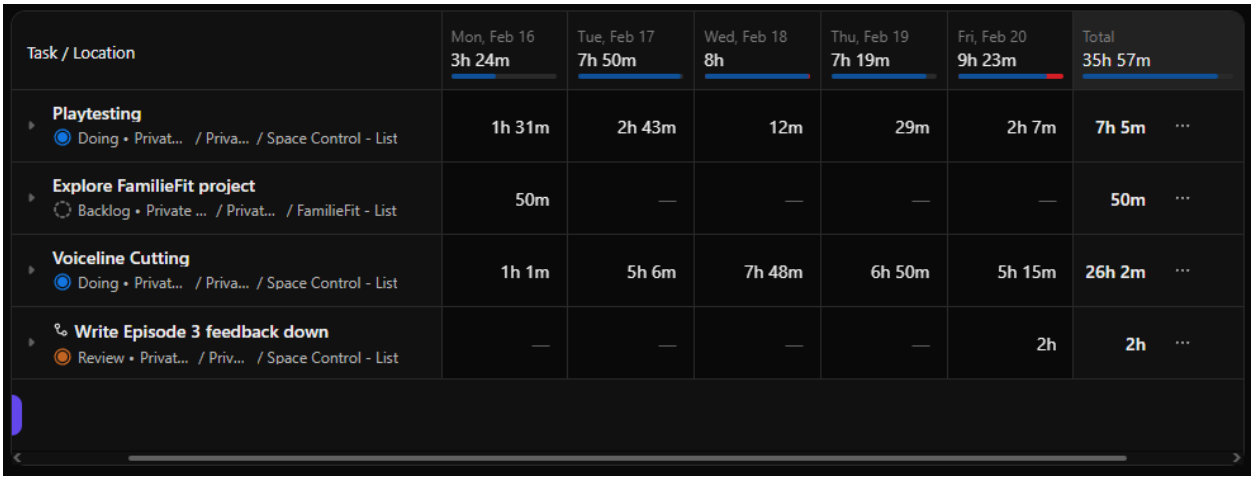
Introduction

This report outlines my activities during the first two weeks of my internship at MoonMonster Studios.

The studio is currently in a pre-launch phase for its upcoming game, Space Control. With the development deadline set for March 1st, the team is focused on final polishing and bug fixing.

During these two weeks, my main responsibilities included:

- Editing and preparing voice lines for in-game implementation
- Playtesting and identifying issues



Task / Location	Mon, Feb 16 3h 24m	Tue, Feb 17 7h 50m	Wed, Feb 18 8h	Thu, Feb 19 7h 19m	Fri, Feb 20 9h 23m	Total 35h 57m
Playtesting Doing • Privat... / Priva... / Space Control - List	1h 31m	2h 43m	12m	29m	2h 7m	**7h 5m**
Explore FamilieFit project Backlog • Private ... / Privat... / FamilieFit - List	50m	—	—	—	—	**50m**
Voiceline Cutting Doing • Privat... / Priva... / Space Control - List	1h 1m	5h 6m	7h 48m	6h 50m	5h 15m	**26h 2m**
Write Episode 3 feedback down Review • Privat... / Priv... / Space Control - List	—	—	—	—	2h	**2h**

Task / Location	Mon, Feb 23 7h 46m	Tue, Feb 24 7h 51m	Wed, Feb 25 7h 37m	Thu, Feb 26 8h 1m	Fri, Feb 27 7h 54m	Total 39h 10m
Voiceline Cutting Doing • Priva... / P... / Space Control - List	7h 46m	3h 22m	3h 14m	6h 12m	—	**20h 34m**
Playtesting Doing • Priva... / P... / Space Control - List	—	2h 35m	—	—	2h 6m	**4h 41m**
Bugfixing Backlog • Priva... / ... / Space Control - L...	—	1m	—	—	—	**1m**
Setting up environment Backlog • Priva... / ... / Space Control - L...	—	6m	—	—	—	**6m**
When spraying one customer in a group, the cloud particles should be... Review • Priva... / P... / Space Control - List	—	1h 47m	4h 22m	—	—	**6h 9m**
Bazooka in EP1 outro should be interactable Doing • Priva... / P... / Space Control - List	—	—	—	1h 49m	5h 48m	**7h 37m**

I will not cover every activity in detail. For example, my first day and initial project exploration mainly involved installing required software and reviewing the projects I will be contributing to.

Instead, this report focuses on the most significant portion of my work so far.

Voice Lines Cutting

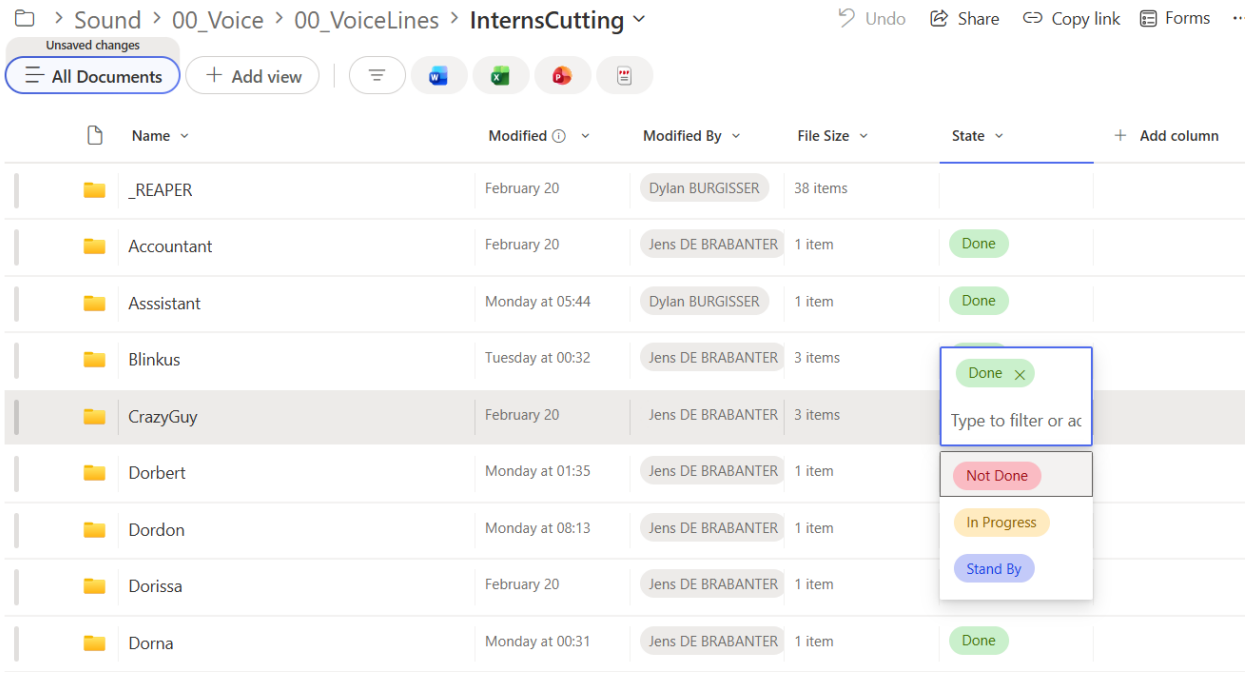
Cutting voice lines took up **more than half of my time** during the first two weeks of the internship.

The process involved using REAPER to split, organize, and prepare recorded voice lines so they could later be implemented into the game.

Below is the workflow that we followed.

1. Receiving the Audio Files

Brian, who was responsible for managing the voice lines, would upload the raw audio files that needed to be processed.



In the table, you can see a **“State”** column. This was one of the improvements I added to the workflow.

Since some voice lines had higher priority than others, it was important to keep everything organized while working together with the other intern. The column contained four possible states:

- **Not Done** – Not started yet
- **In Progress** – Currently being edited
- **Done** – Ready to be checked

- **Stand By** – Lines that may need to be recorded again (for example by influencers or due to audio issues)

2. Starting the Work

After downloading an audio file, we would immediately update its status in the table to **WIP**.

	A	B	C	D	E	F	G	H
1	**Character Name**		**Gender**	**Voice Actor Name**	**Recorded**	**Implemented**	**Volume Adjust**	**Reimported**
2	Speaker (Intro Video)	Intro Video Voice	M	**Sean Chiplock**	YES	YES	NO	NO
3	Sydra	Sydra	F	**Amanda Hufford**	YES	YES	WIP	NO
4	Melody	Melody	F	**Kelsey Jaffer**	YES	YES	NO	NO
5	Zorgle	Zorgle	M	**Tim Heller**	YES	YES	NO	NO

Once the work was completed, the status would be updated to **Review** so Brian could check the results.

3. Using the Voice Line Script

Next, we needed to locate the **script for the character** whose lines we were processing.

	A	B	C	D	E	F
1	InEngineFileName	Episode	Line Nur	Actor	Notes To Voice Actor	Script
7	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_011_Tutorial_Bored_cb3a9	EP2	2870	Saleswoman		Keep your head in the game, ugh!
8	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_005_Assistant_0fc53	EP2	2871	Assistant		Uhm, ma'am...
9	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_024_Tutorial_Bored_1ebf4	EP2	2872	Saleswoman		Shut up, Clyde, can't you see I'm in a meeting!
10						
11	CV_EP2_BoredTutorial_DROPOFF_GiveBaby_2	********	*******	**********	**********	CV_EP2_BoredTutorial_DROPOFF_GiveBaby_2
12	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_2_020_Tutorial_Bored_026f2	EP2	2873	Saleswoman		No, Martha, that was at Clyde, yes
13	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_2_010_Assistant_eb13d	EP2	2874	Assistant		I apologize on behalf of my boss
14						
15	CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await	********	*******	**********	**********	CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await
16	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_002_Assistant_703b9	EP2	2875	Assistant		We're actually here for her darling child
17	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_055_Sydra_0a706	EP2	2876	Sydra		That is something we can help with
18	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_004_Assistant_34251	EP2	2877	Assistant		Please take good care of her
19	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_000_Tutorial_Bored_bb149	EP2	2878	Saleswoman		Tell accounting that I will have the budget ready by tomorr
20	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_046_Assistant_314e5	EP2	2879	Assistant		Excuse me, ma'am
21	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_025_Tutorial_Bored_49ec4	EP2	2880	Saleswoman		What, didn't you hear me?
22	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_056_Tutorial_Bored_e2736	EP2	2881	Saleswoman		I said latte macchiato, extra foam and sugar, thank you!
23	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_026_Tutorial_Bored_c4e2b	EP2	2882	Saleswoman		The state of the service industry these days...
24	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_030_Tutorial_Bored_fe56f	EP2	2883	Saleswoman		Can't compare to 10 years ago, isn't that right, Martha?
25	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_031_Tutorial_Bored_105d6	EP2	2884	Saleswoman		I know, I know
26	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_006_Assistant_e71d9	EP2	2885	Assistant		I'm sorry but - could we get some help!
27	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_007_Tutorial_Bored_45df6	EP2	2886	Saleswoman		And tell marketing they need to up their game
28	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_008_Assistant_5771e	EP2	2887	Assistant		Uhm - hello?
29	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_009_Tutorial_Bored_196fd	EP2	2888	Saleswoman		What is taking so long, how hard is it to make a simple cof
30	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_011_Tutorial_Bored_17fce	EP2	2889	Saleswoman		So sorry about that, Martha
31	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_014_Tutorial_Bored_a9b18	EP2	2890	Saleswoman		Yeah, I know, how hard is it to make a latte macchiato, ho
32	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_015_Assistant_3464b	EP2	2891	Assistant		I'm sorry, but could you please hurry up?
33	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_010_Assistant_c261f	EP2	2892	Assistant		Could you - could you accept the baby?
34	VO_CV_EP2_BoredTutorial_DROPOFF_GiveBaby_Await_012_Tutorial_Bored_0620b	EP2	2893	Saleswoman		Tell the board that I'll be there in 7 to 9 minutes, tops

The script served as our guide during the editing process. In the document, the lines highlighted in blue represented the voice lines spoken by the character we were working on.

4. Editing in REAPER

We would then open the audio file in REAPER and begin the editing process.

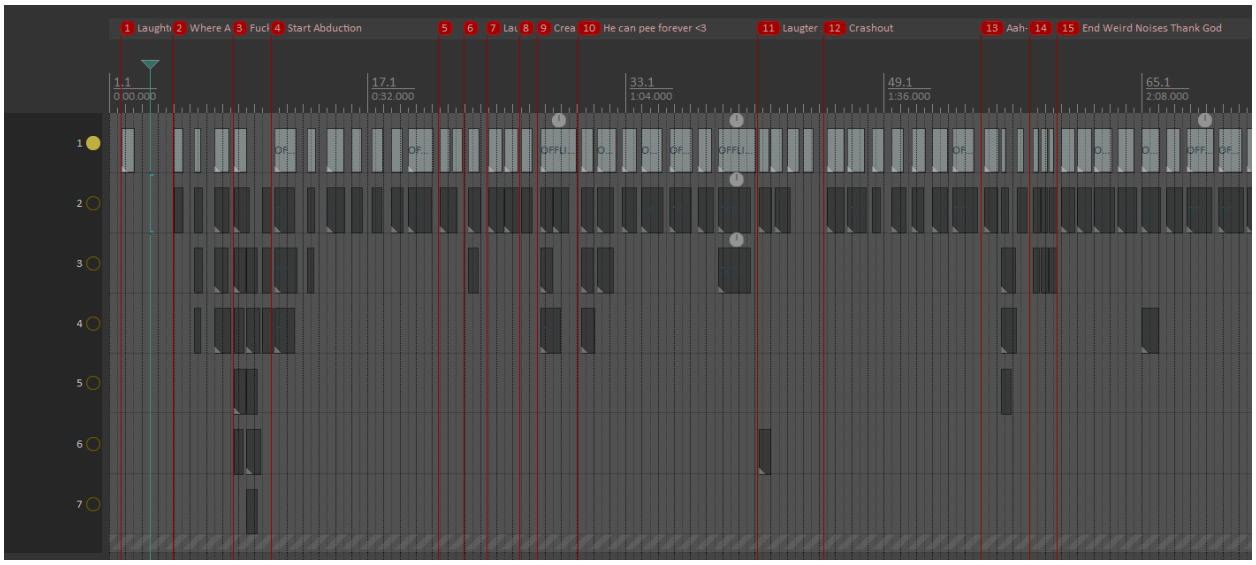
The steps included:

- **Normalizing** the audio
- **Splitting** the file into individual voice lines
- **Arranging** the lines according to the script

Each line usually had **at least two takes**, and sometimes more. When possible, we organized the takes so that the **best one appeared first**, which helped speed up later integration.

5. Final Result

After finishing the editing and organization, the project timeline would look like this:



At this point, the voice lines were ready for export. I would close the project and upload them to the designated folder so Brian could review and integrate them into the game.

Later in the process, I also started using **color-coded markers** to highlight sections that required Brian's attention. This made it easier to spot potential issues and helped speed up the review process.

It is now ready for export, so I would close the program and upload it in the designed folder ready for Brian to check and upload.

Playtesting

The second largest part of my work during these two weeks was **playtesting the game** Space Control.

I completed **two full playthroughs**, each with a different objective.

First Playthrough – Player Experience

During the first playthrough, my goal was to experience the game as a **regular player** and provide general feedback.

I focused on questions such as:

- Did I get stuck anywhere?
- Were any mechanics confusing?
- Were certain sections frustrating or unclear?
- Which parts felt engaging or interesting?
- Which parts felt slow or boring?

The goal was to identify areas where the **player experience could be improved** before release.

Second Playthrough – Breaking the Game

The second playthrough was more technical. My goal this time was to **actively try to break the game**.

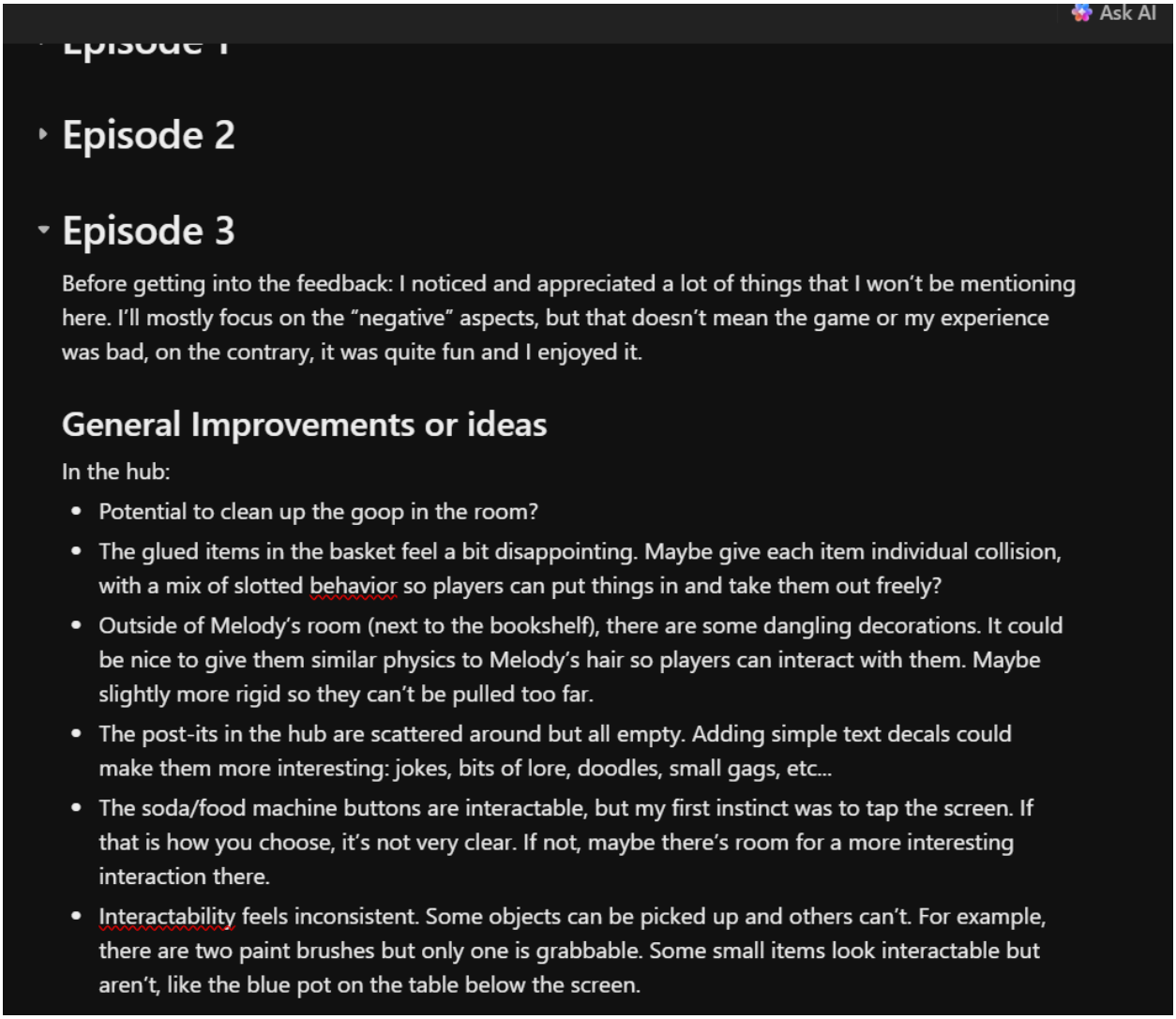
Since the game is divided into **three episodes**, I tested each one carefully.

- **Episode 1** was already quite stable. I did not encounter any game-breaking issues.
- **Episodes 2 and 3** were less polished, and I managed to break the game multiple times.

The issues I encountered ranged from **simple logic errors** to **collision bugs**. For example, I was able to clip into areas where the player was not supposed to go.

Reporting Feedback

All feedback and bug reports were documented in ClickUp.



The feedback was organized by:

- **Episode**
- **Section of the level**
- **Type of issue or feedback**

This structure made it easier for the development team to quickly locate and address the reported issues.

Programming

Finally, we arrive at the last part of my work during these two weeks - **programming**.

Although most of my time was spent on audio editing and playtesting, I also had the opportunity to implement **two small quality-of-life improvements** for the project.

Before starting, I had to get familiar with the project's version control setup. The codebase is hosted on a **private repository on** GitHub. To communicate with the repository, the team uses SourceTree.

Since I normally work with **Git Bash or GitHub Desktop**, this was a new tool for me. Learning how the team manages branches and commits through SourceTree is something I will continue improving throughout my internship.

Spray Particles Effect

The first task involved improving the **spray interaction with characters**.

In one of the episodes, the player can spray characters using a spray tool. The system was already functional, but there was a small issue:

When multiple characters were grouped together, the **spray particle effect would only appear on the front character**, even if multiple characters were hit.

My task was to ensure that **all affected characters would display the spray particles**.

My Approach

Since this was my **first time exploring the project**, the main challenge was understanding the existing code structure.

After investigating the relevant scripts, I identified where:

- Characters were being detected by the spray system
- The particle effect was being spawned

Originally, the system used:

- A **single transform reference**
- A **raycast** to detect the target
- The **raycast hit position** to spawn the particles

My Solution

To support multiple characters being sprayed, I modified the system to:

- Use a **list of transforms** instead of a single transform
- Detect multiple characters affected by the spray
- Play particle effects on each character instead of only at the raycast hit position

This allowed the spray effect to **correctly appear on every character in the group**.

(I would insert and image here, but I forgot to take one I will do better next week)

Interactable Bazooka

After becoming more familiar with the project structure, I moved on to my second task: making a **bazooka fully interactive**.

Fortunately, an interactable bazooka already existed in the project, meaning the player could **pick it up**. However, the weapon **did not yet have functional shooting mechanics**.

So, my next step was to implement the behaviour of firing the weapon.

My Implementation

I created a new script responsible for handling:

- Sound effects
- Particle effects
- Camera shake
- Projectile firing

Using an existing **factory pattern template** and **object pooling system**, I implemented a **rocket projectile**.

When the rocket impacts something, it:

- Uses an **OverlapSphere** to gather nearby colliders
- Applies a force to the connected rigid body

This resulted in a working bazooka with **proper explosion effects and performance-friendly projectile spawning**.

```
var cols :Collider[] = Physics.OverlapSphere(transform.position, _blastRadius, _collisionFilters);
foreach (var col :Collider in cols)
{
    var rb :Rigidbody = col.attachedRigidbody;

    if (rb)
        rb.AddForce((col.transform.position - transform.position).normalized * _blastForce, ForceMode.Impulse);
}
```

Conclusion

During these two weeks, I was able to gain insight into how a small studio manages the final stages of development for Space Control. Working close to release highlighted how much time is dedicated to polishing, testing, and preparing assets rather than creating new features.

From a technical perspective, I became familiar with new tools such as REAPER for audio editing and Sourcetree for repository management. I also refreshed my experience working with Unity and C# while implementing small gameplay improvements.

Beyond the technical aspects, this period also helped me better understand how development priorities change as a project approaches release. Tasks often shift toward **testing, polishing, and asset preparation**, which are essential for delivering a stable product.

Looking back on this period, my experience has been mixed. On a positive note, the team at MoonMonster Studios has been welcoming and supportive. The team environment is generally friendly, and I have been making an effort to integrate by joining group activities such as lunch breaks.

However, the team primarily communicates in Dutch, which sometimes makes it difficult for me to fully participate in conversations. While this is understandable given the composition of the team, it occasionally creates a sense of being slightly outside of the group dynamic.

From a technical perspective, most of my time during these two weeks was spent on tasks such as voice line editing and playtesting for Space Control. These tasks were important for the project's polishing, especially given that the team is currently approaching its release deadline.

Because the studio is in the final phase before launch, the priority is clearly on **polishing and completing the game**. This means that many of the tasks assigned to me were production-oriented rather than learning-oriented. As a result, I have not yet had many opportunities to apply or expand my programming skills.

Overall, I feel that my **integration into the team is progressing well**, but I hope that after the game's launch there will be more opportunities to contribute in ways that better align with my programming background. I believe that with more technical tasks and responsibilities, I will be able to contribute more effectively while also gaining valuable experience during the remainder of the internship.

I have been informed that the voice line editing phase is now complete. The project has entered a development lockdown, meaning the team is focusing on bug fixing and polishing ahead of the release and day-one patch for Space Control.

While the tasks assigned during these first two weeks were helpful for supporting the project's immediate production needs, they did not always align directly with the technical learning objectives I had initially hoped to pursue during this internship. However, with the transition

toward post-release development, I look forward to contributing to new features and gameplay elements while continuing to develop my programming skills.

Key Takeaways from the First Two Weeks

Project context

- The team at MoonMonster Studios is currently in the final development phase of Space Control.
- Most work during this period focuses on **polishing, testing, and preparing the game for release**.

Main tasks completed

- Processed and organized voice lines using REAPER.
- Conducted two full playthroughs to identify usability issues and bugs.
- Reported feedback and issues through ClickUp.
- Implemented two small programming improvements in Unity using C#.

Tools learned

- REAPER – audio editing workflow
- Sourcetree – repository management
- ClickUp – internal task and feedback tracking

Looking ahead

- With the upcoming release of Space Control, the team is currently focused on bug fixing and polishing.
- After launch, I hope to contribute more directly to programming tasks and the development of post-release features.

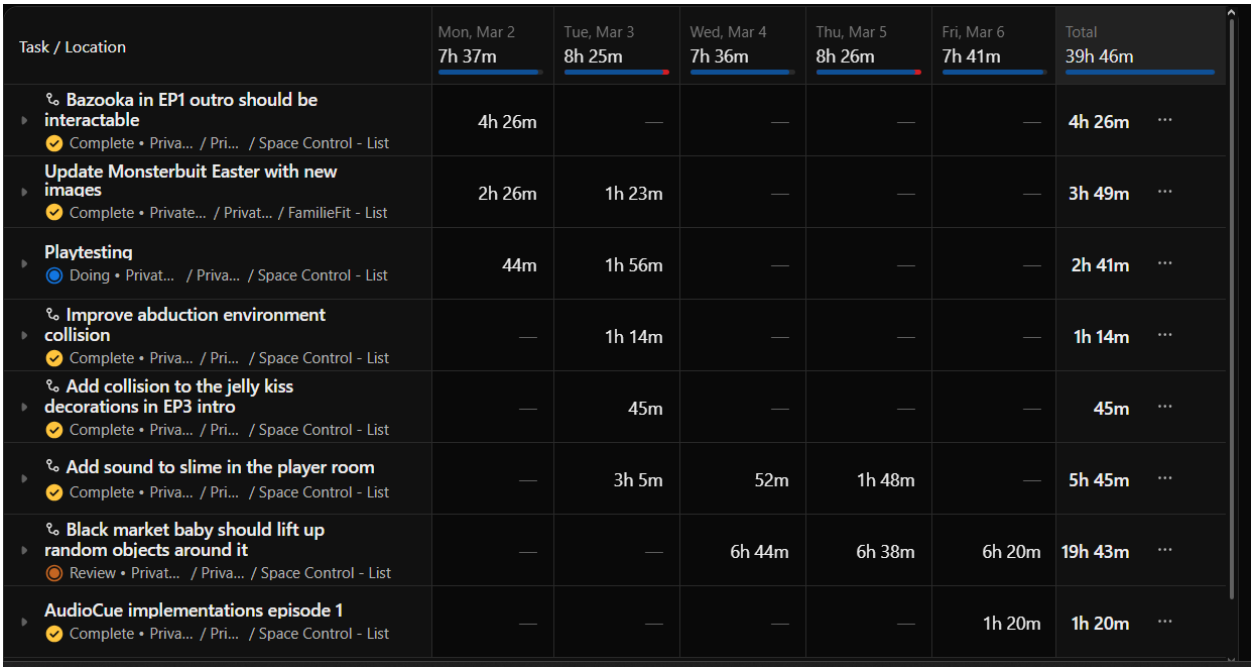
Internship Report 02

Week 03 & 04

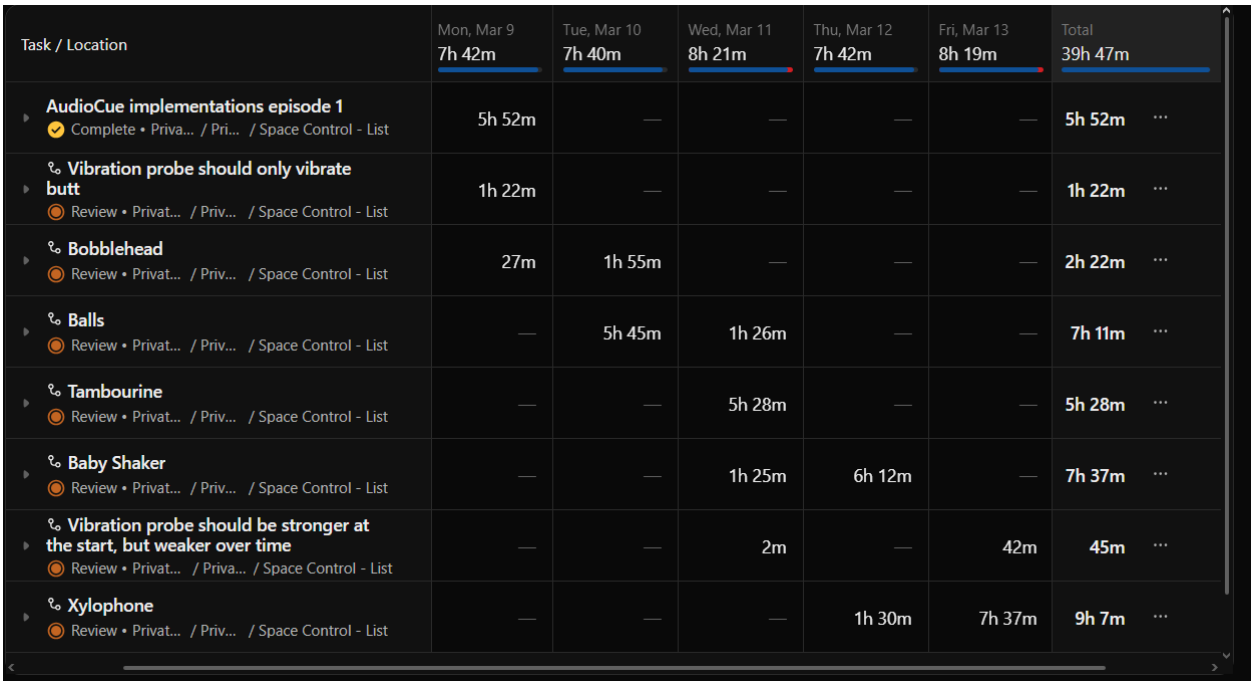
Burgisser Dylan

Introduction

This report outlines my activities during the third and fourth week of my internship at MoonMonster Studios. During this period, my responsibilities were more varied compared to the first two weeks. I worked on a range of tasks focused on polishing the game and preparing it for release.



Task / Location	Mon, Mar 2 7h 37m	Tue, Mar 3 8h 25m	Wed, Mar 4 7h 36m	Thu, Mar 5 8h 26m	Fri, Mar 6 7h 41m	Total 39h 46m
Bazooka in EP1 outro should be interactable Complete • Priva... / Pri... / Space Control - List	4h 26m	—	—	—	—	4h 26m
Update Monsterbuit Easter with new images Complete • Private... / Privat... / FamilieFit - List	2h 26m	1h 23m	—	—	—	3h 49m
Playtesting Doing • Privat... / Priva... / Space Control - List	44m	1h 56m	—	—	—	2h 41m
Improve abduction environment collision Complete • Priva... / Pri... / Space Control - List	—	1h 14m	—	—	—	1h 14m
Add collision to the jelly kiss decorations in EP3 intro Complete • Priva... / Pri... / Space Control - List	—	45m	—	—	—	45m
Add sound to slime in the player room Complete • Priva... / Pri... / Space Control - List	—	3h 5m	52m	1h 48m	—	5h 45m
Black market baby should lift up random objects around it Review • Privat... / Priva... / Space Control - List	—	—	6h 44m	6h 38m	6h 20m	19h 43m
AudioCue implementations episode 1 Complete • Priva... / Pri... / Space Control - List	—	—	—	—	1h 20m	1h 20m



Task / Location	Mon, Mar 9 7h 42m	Tue, Mar 10 7h 40m	Wed, Mar 11 8h 21m	Thu, Mar 12 7h 42m	Fri, Mar 13 8h 19m	Total 39h 47m
AudioCue implementations episode 1 Complete • Priva... / Pri... / Space Control - List	5h 52m	—	—	—	—	5h 52m
Vibration probe should only vibrate butt Review • Privat... / Priv... / Space Control - List	1h 22m	—	—	—	—	1h 22m
Bobblehead Review • Privat... / Priv... / Space Control - List	27m	1h 55m	—	—	—	2h 22m
Balls Review • Privat... / Priv... / Space Control - List	—	5h 45m	1h 26m	—	—	7h 11m
Tambourine Review • Privat... / Priv... / Space Control - List	—	—	5h 28m	—	—	5h 28m
Baby Shaker Review • Privat... / Priv... / Space Control - List	—	—	1h 25m	6h 12m	—	7h 37m
Vibration probe should be stronger at the start, but weaker over time Review • Privat... / Priva... / Space Control - List	—	—	2m	—	42m	45m
Xylophone Review • Privat... / Priv... / Space Control - List	—	—	—	1h 30m	7h 37m	9h 7m

As there is a considerable amount of work to discuss from these two weeks, the tasks will be grouped into two main categories: **Polishing**, which primarily covers the third week, and **Additions**, which mainly covers the fourth week. Tasks that do not fall into either category are generally self-explanatory, such as playtesting.

Polishing

1. Interactive Bazooka

Similar to the previous week, this task involved improving the interactive bazooka, which allows the player to shoot projectiles and blow things up. The feature was already functional but required additional polishing to make it game ready. One of the main improvements was making the recharge feedback clearer for the player.

Key Learning Points:

- Learned more about the project's codebase.
- Discovered and used Scriptable Objects.
- Used Factory and Object Pooling systems for the first time.
- Worked with asynchronous programming.
- Learned about torque forces.

2. Collision Improvements

This task involved improving collision behaviour for several existing objects. The goal was simply to ensure that proper collision was implemented to avoid inconsistencies during gameplay.

While the task itself was relatively straightforward, it helped me better understand how larger Unity projects organize their scenes.

Key Learning Points:

- Learned how scenes in larger projects are structured using multiple prefabs and complex prefab hierarchies.
- Understood how this structure allows multiple developers to modify prefabs without directly changing the scenes themselves.
- This approach reduces conflicts when using version control systems, particularly by avoiding unnecessary modifications to binary scene files.

3. Adding Sound to the Slime Shower

What is the slime shower?

It is exactly what it sounds like, a shower… of slime. And it needed audio!

This was my first time working with audio within the project, and it also revealed one of my weaknesses as a programmer – making audio sound good is harder than it looks.

The slime can be grabbed, slapped, and moved around, so it needed audio feedback to make interactions feel satisfying. During this task I experimented with several techniques that I would later reuse and improve upon while continuing to work on audio systems.

My first objective was to make the slime produce sound when it moved. This turned out to be more challenging than expected. Finding suitable audio clips and making them sound natural required several iterations.

I eventually used a looping sound combined with multiple thresholds to determine when the sound should start and stop. Initially I attempted to use the slime's Rigidbody velocity directly to control the audio, but this resulted in inconsistent playback. Introducing velocity thresholds helped stabilize the behaviour.

To improve the effect further, I interpolated the volume and pitch based on the slime's velocity, simulating acceleration as it moved faster.

The final touch was adding sounds on OnCollisionEnter events for impacts and slaps. With this implemented, the slime shower finally produced satisfying audio feedback during interactions.

Key Learning Points:

- Learned how to use the audio system implemented in the project.
- Gained a newfound respect for audio engineers.
- Learned several techniques for implementing responsive and satisfying sound effects.

4. Baby Telepathy Power

This was the largest task of the first week and took nearly **20 hours** to complete!

The objective was to implement a power for a specific NPC that allowed it to randomly lift nearby objects using telekinesis. To keep the explanation concise, the implementation process can be summarized in several steps.

The steps included:

- Randomly trigger the power.
- Using a sphere collision check to detect nearby objects.
- Storing relevant information about detected objects inside a struct.
- Starting a recursive coroutine that periodically inverted the power's direction.
- Stopping the power.
- Updating the forces applied to the stored objects.

While this approach initially worked, it quickly reached its limits when the design evolved. The power needed to dynamically lift objects that entered the radius and release objects that left it. Additionally, the baby needed to be able to lift other babies.

Improvements to the System

To support these new requirements, I redesigned part of the system.

First, I replaced the OverlapSphere detection with a dynamically created **Capsule Collider**. This allowed me to track objects entering and leaving the power's area using collision events.

The second challenge involved lifting other babies. Their ragdoll bodies consist of multiple rigidbodies and colliders, so I needed to determine how to select and use only one specific bone rather than the entire ragdoll structure.

After solving these challenges, I successfully created a telekinetic lifting system capable of dynamically lifting objects within a given radius.

Key Learning Points:

- Learned how to use **EventHandler** systems in Unity.
- Gained experience working with **Coroutines**.
- Learned about the different **ForceMode** options.
- Worked with **RagdollAnimator** and **RagdollHandler** components.
- Learned to use **Editor Attributes** for serialized values such as **Min**, **Max**, and **Range**.

Additions

1. AudioCue Implementations

Although this task could arguably fall under polishing, I included it in this section because most of the work was done during my second week.

The objective was to go through a list of interactable items that required audio feedback. Some of these items already had audio implemented but needed adjustments, such as increasing the volume or improving clarity. Others had no audio at all, which required me to create the audio player logic and implement the sounds from scratch.

Each completed item was then marked on a shared spreadsheet to keep track of progress and ensure that all required objects had proper audio feedback implemented.

AC_Fryer_Fries_Ingredient_Completed	PF_PROP_Ingredient_Base	Volume bad	
AC_Concert_Gesus_Harp_Floating	Concert_Job_GesusLyrePimped	No sound	
AC_Drone_Land	CV_EP3_Customer_RapperBot_4_CorrectItem	No sound	
AC_Mutator_Ambient	PF_MACHINE_MutatorRay	No sound	Should be very subtle, no-gravity sound

2. Bobblehead

My first fidget toy was a bobblehead. This was a relatively simple task and did not require scripting. However, it allowed me to experiment with Unity's physics system.

I mainly worked with **Configurable Joints** and adjusted physics parameters until the bobblehead's motion felt natural and satisfying.

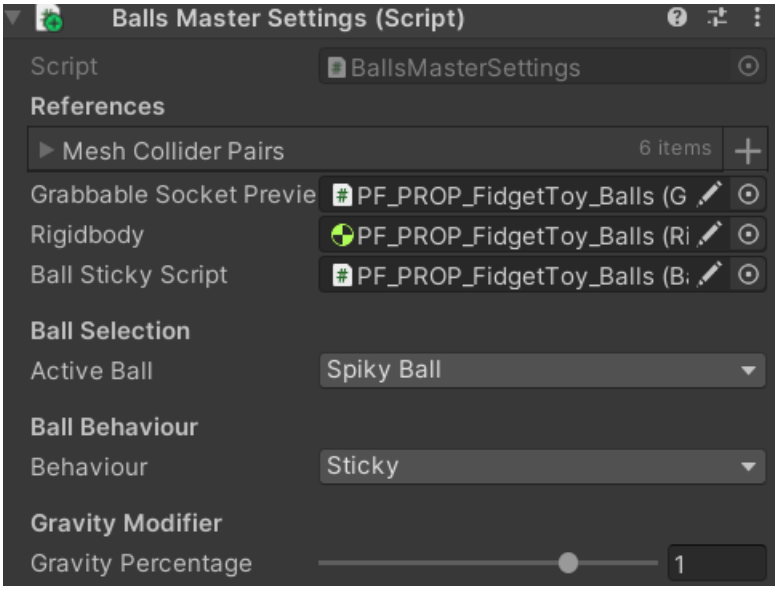
Key Learning Points:

- Configurable Joints.
- Deeper understanding of Unity's physics system.

3. Balls

The next toy I worked on involved different types of balls: sticky balls, bouncy balls, and even gravity-defying balls.

For this feature I created a **modular script** that allows different types of balls to be created by simply adjusting a few parameters.



With a small set of settings, it is possible to choose:

- The mesh of the ball.
- The ball's behaviour.
- A gravity modifier.

This approach allows the creation of multiple ball variants without rewriting code.

Key Learning Points:

- Learned that changing the physics material of a collider at runtime must be applied to **sharedMaterial** rather than **material**, despite what the Unity 2023.3 documentation suggests.
- Discovered **ConstantForce** component.

4. Tambourine & baby Shaker

Both objects are musical instruments and share similar behaviours. For their implementation, I reused techniques I had developed earlier while working on the slime shower audio system. This helped improve my workflow and implementation speed.

Despite that, making the instruments sound satisfying still required a considerable amount of iteration.

For the **baby shaker**, an additional requirement was making it interactive with the babies so that they could also pick it up and shake it!

Key Learning Points:

- Implementing convincing audio feedback is more challenging than expected.
- More natural motion in animations often requires **stacked forces or movements**. Applying motion in multiple directions with varying strengths generally produces better results than using a single force.

5. Xylophone

The The Xylophone turned out to be the most frustrating task of the week.

After working on several audio-related features and instruments, this initially seemed like a straightforward task. Conceptually, the system is simple: each key has a collider, and when that collider detects a collision, the corresponding note is played.

However, Unity had other plans.

First Issue – Collision Detection

To simplify the implementation, I created a XylophoneNote.cs script and attached it to each collider representing a note. The script listened for the OnCollisionEnter event and played the corresponding sound.

Unfortunately, this did not work. OnCollisionEnter was never called. Interestingly, OnTriggerEnter worked when the collider was set as a trigger, but the normal collision event did not.

To solve this, I changed my approach and handled the OnCollisionEnter event on the **parent object** instead. I then created a dictionary mapping each collider to its corresponding XylophoneNote script. This allowed me to determine which key had been hit.

Second Issue – Early Collision Detection

Although the previous solution worked, another problem appeared. OnCollisionEnter was being triggered **too early**, before the colliders were visibly touching. As a result, hitting one key would sometimes also trigger neighbouring keys.

To mitigate this issue, I ignored additional collisions occurring within the same physics frame. This reduced the problem, although the solution may need to be revisited later as multiple keys can be played at once.

Third Issue – Jingle Tracking

I also implemented a **jingle system** where playing a specific sequence of notes would trigger a predefined jingle.

To achieve this, I created a Jingle struct that stored the sequence of notes and tracked which note in the sequence had been played. However, the tracking system behaved unpredictably, with the internal counter producing inconsistent values.

```
private void StoreNotePlayed(object sender, XylophoneNoteID noteID)
{
    if (NotesToPlay[_currentNoteIndex] == noteID)
    {
        ++_currentNoteIndex;
        if (_currentNoteIndex >= NotesToPlay.Length)
        {
            _currentNoteIndex = 0;
            JingleToPlay.Play(new AudioManager.PlayArgs { Parent =
_playTransform });
            OnJinglePlayedEvent?.Invoke(this, EventArgs.Empty);
        }
    }
    else
    {
        _currentNoteIndex = 0;
        if (NotesToPlay[0] == noteID)
        {
            _currentNoteIndex = 1;
        }
    }
}
```

After some investigation, I discovered that the issue was related to using a **struct**. The internal counter logic would break for reasons that are still unbeknownst to me.

The problem was resolved by converting the struct into a **class**.

Key Learning Points:

- Unity physics behaviour can sometimes be unpredictable.
- Avoid implementing methods inside structs, prefer using class instead.

Conclusion

During these two weeks, I worked on a variety of programming tasks. This represents a significant improvement compared to the first two weeks, as it allowed my learning experience to shift into a higher gear. Working on different types of tasks also enabled me to develop a broader range of programming tools and skills.

The team still primarily communicates in Dutch, which sometimes makes it difficult for me to fully participate in conversations. However, they have been making a conscious effort to speak English around me, which I greatly appreciate.

Overall, I feel that my integration within the team continues to progress well. I am satisfied with the tasks I have been assigned, as their variety allows me to explore different programming

concepts and further develop my knowledge of Unity and programming in general. Having worked mostly on programming during these two weeks, I also feel better able to evaluate my own abilities. However, this has left me with a lingering feeling that I may be underperforming. As my supervisor is currently quite busy, there has been limited opportunity for detailed feedback, which makes it difficult for me to accurately judge how well I am doing as an intern.

Based on my own expectations, I sometimes feel behind in my work, as certain tasks have taken two to four times longer than I would normally expect. This is likely due to several factors, including the time spent refining audio implementation, the considerable size and complexity of the project's codebase, the need to adapt to the existing structure of the project, and occasionally unpredictable behaviour from Unity itself.

For the time being, I will need to readjust my expectations and focus on gradually overcoming these challenges as I continue learning and gaining experience in the coming weeks.

Internship Report 03

Week 05 & 06

Burgisser Dylan

Introduction

This report outlines my activities during the fifth and sixth week of my internship at MoonMonster Studios. This period was similar to the previous one, having more varied tasks to prepare the game for release on April 1st.

Task / Location	Mon, Mar 16 8h 13m	Tue, Mar 17 7h 44m	Wed, Mar 18 7h 41m	Thu, Mar 19 7h 17m	Fri, Mar 20 8h 12m	Total 39h 8m
Rocket probe should be stronger at start, but weaker over time Complete • Privat... / Priv... / Space Control - List	37m	—	—	—	—	**37m**
Make selection screens for the dispensers in the hub interactable Complete • Privat... / Priv... / Space Control - List	3h 6m	—	—	—	—	**3h 6m**
Improve ease of hitting drums Complete • Priva... / Priv... / Space Control - List	4h 30m	1h 25m	1h 44m	1h 48m	5h 24m	**14h 53m**
Implement new visuals for ReviewReward Complete • Priva... / Priv... / Space Control - List	—	2h 53m	—	—	—	**2h 53m**
Playtesting Doing • Private... / Privat... / Space Control - List	—	2h 20m	5h 56m	—	—	**8h 17m**
Fixing balls getting parented Complete • Priva... / Priv... / Space Control - List	—	1h 4m	—	—	—	**1h 4m**
Make an Android build Review • Private S... / Private ... / FamilieFit - List	—	—	—	5h 29m	42m	**6h 11m**

Task / Location	Mon, Mar 23 8h 34m	Tue, Mar 24 8h 44m	Wed, Mar 25 7h 46m	Thu, Mar 26 8h 29m	Fri, Mar 27 7h 47m	Total 41h 23m
Improve ease of hitting drums Complete • Priva... / Pri... / Space Control - List	4h 32m	—	—	—	—	**4h 32m**
Playtesting Doing • Privat... / Priva... / Space Control - List	4h 1m	—	—	2h 42m	3h 8m	**9h 52m**
Bugfixing Complete • Priva... / Pri... / Space Control - List	—	3h 8m	—	—	—	**3h 8m**
Add extra stats to monsterbuit end screen Review • Private ... / Private... / FamilieFit - List	—	1h 51m	—	—	—	**1h 51m**
Fix Fidget spinners on TOY Variants branch Complete • Priva... / Pri... / Space Control - List	—	53m	1h 35m	—	—	**2h 28m**
Fix FamiliFit bugs Doing • Private S... / Private... / FamilieFit - List	—	2h 51m	28m	—	—	**3h 19m**
Update plates in EP3 job Complete • Priva... / Pri... / Space Control - List	—	—	3h 54m	—	—	**3h 54m**

Task / Location	Mon, Mar 23 8h 34m	Tue, Mar 24 8h 44m	Wed, Mar 25 7h 46m	Thu, Mar 26 8h 29m	Fri, Mar 27 7h 47m	Total 41h 23m
xylophone stick gets stuck in xylophone Complete • Priva... / Pri... / Space Control - List	—	—	20m	—	—	**20m** ···
baby shaker for glebglub is not clear Complete • Priva... / Pri... / Space Control - List	—	—	41m	—	—	**41m** ···
Fix psychic baby affecting beds Complete • Priva... / Pri... / Space Control - List	—	—	46m	—	—	**46m** ···
make hub screen blink soda machine Complete • Priva... / Pri... / Space Control - List	—	—	1m	45m	—	**46m** ···
Fix plates tween step not correct Complete • Priva... / Pri... / Space Control - List	—	—	—	1h 19m	—	**1h 19m** ···
Update soda and nutricube dispensers in hub Complete • Priva... / Pri... / Space Control - List	—	—	—	3h 42m	—	**3h 42m** ···
Polish-VFX branch issues + implementation Doing • Privat... / Priva... / Space Control - List	—	—	—	—	4h 39m	**4h 39m** ···

Week 5

1. Probe Strength

A continuation of Week 4, where I implemented a lifetime strength system for the vibration probe. This time, I applied a similar system to the rocket probe, which uses a pushing mechanic instead of vibration.

Key Learning Points:

- Discovered and applied the inverse lerp function.
- Gained further experience working with Scriptable Objects.

2. Interactive Dispenser Screen

This task involved creating UI in VR, which differs significantly from traditional 2D UI systems. Fortunately, existing interactive screen examples in the project helped me understand the workflow more quickly.

Key Learning Points:

- Learned how to implement interactive UI systems in VR environment.

3. Improve Drumming Sequence

This was one of the more complex tasks and was originally expected to extend beyond release, which is why it was spread across multiple days.

During the final episode, there is a Guitar Hero style drumming sequence. During playtesting, I found it frustrating due to difficulty hitting the drums at the correct time.

The core issue was **visibility**:

- Looking at the note lanes made it hard to see the drums.
- Looking at the drums made it hard to see incoming notes.

This feedback was shared by others, which is likely why I was assigned to improve the system.

I explored multiple solutions:

- Adjusting drum and lane positions.
- Modifying stick handling.
- Tweaking collider sizes and detection.
- Changing note movement so notes reached the drums.

While bringing notes closer improved visibility, it negatively impacted the scene by drawing too much focus to the sequence. This approach was ultimately rejected.

At that point, improving the existing system through tweaking alone was not sufficient.

I developed a new solution:

- Added trigger colliders to each drum.
- Tracked when sticks entered and exited these zones.
- Detected whether the player intended to hit the drum.
- If a valid hit motion was detected but missed slightly, it would still count as a hit.

This solution improved responsiveness while maintaining immersion and avoiding making the system too forgiving.

Additional improvements included:

- Disabling vertical collisions on drumsticks.
- General sequence tuning and balancing.

Key Learning Points:

- Designing systems that balance player assistance with immersion.
- Working with trigger colliders and motion-based detection.
- Iterative problem-solving and prototyping.

4. Android Build

With Easter approaching, the project required an updated Android build. This was my first time creating an Android build in Unity, which differed from my previous experience.

Key Learning Points:

- Learned Android build processes in Unity.
- Gained experience with Android debugging tools.

Week 6

1. Bug Fixing

As expected, several issues arose from earlier implementations.

One major issue involved collision handling:

- I disabled collisions when sticks moved upward.
- After hitting a drum, the stick would bounce upward → disable collision → exit trigger → move slightly downward → re-enable collision.
- This loop occurred every frame, allowing up to **30 hits per second**.

I resolved this by stabilizing the collision logic and preventing rapid toggling.

Additionally, I worked on Android-related bugs, including issues caused by a poorly designed persistence system created by a former developer. This system continues to introduce complications.

Key Learning Points:

- Deeper understanding of Unity physics, triggers, and collision systems.
- Experience debugging complex legacy systems.
- Learned about saving and loading persistent data in Unity.

2. Fidget Spinners

This task involved implementing particle effects created by the art intern onto fidget spinner objects.

Key Learning Points:

- Improved understanding of Unity's particle systems.
- Learned how particle systems behave when parented to objects.

3. Plate Animations

I integrated animations created by the art intern into a plate washing station.

The original system used simple tweening to move plates through pipes. My task was to:

- Implement animations for dropping and suction.

- Trigger audio and VFX at appropriate times.

Challenges included:

- Animations created using mixed systems, including a legacy system.

After resolving these:

- I added animation events for sound and VFX.
- Synchronized animations with the tweening system.

Key Learning Points:

- Working with animations and Animator systems.
- Using animation events.
- Gaining experience with DoTween and tween paths.
- Understanding that tween paths are cached at editor time.

4. Soda and Food Dispenser

I modified the dispenser system so that items remain locked until a new one is dispensed.

An existing system already handled similar logic, so my main task was adapting it for food and soda dispensers. I also improved the structure by combining related components into a single prefab for easier workflow.

5. Polishing VFX implementation

I worked on integrating various VFX created by the art intern.

Tasks included:

- Implementing a can-opening effect.
- Fixing dart particles triggering on spawn due to object pooling.
- Adding disappearance VFX for a syringe.
- Creating a splash effect for soda streams.

The splash effect required using OnParticleCollision, which was new to me and required experimentation to implement correctly.

Key Learning Points:

- Understanding edge cases in Unity physics and particle behaviour.
- Learning when to use classes instead of structs.
- Working with particle collision events.

Conclusion

During these two weeks, I worked on a wide range of programming tasks, from UI and gameplay systems to debugging and VFX integration. Some tasks were more complex than others, but all contributed to expanding my technical knowledge and problem-solving skills. These experiences also gave me the opportunity to demonstrate my ability to tackle challenging systems and improve existing gameplay mechanics.

Internship Report 04

Week 05 & 06 & 07

Burgisser Dylan

Introduction

This report outlines my activities during the fifth, sixth and seventh week of my internship at MoonMonster Studios. With the release on April 1st, I spent most of my time playtesting and bug fixing. Thus, this report will be shorter.

While I learned new things from the code I needed to fix as well as the new systems I used to fix certain issues, it isn't particularly relevant to mention.

Task / Location	Mon, Mar 30 7h 40m	Tue, Mar 31 6h 37m	Wed, Apr 1 4h 34m	Thu, Apr 2 8h 1m	Fri, Apr 3 7h 29m	Total 34h 24m
Polish-VFX branch issues + implementation Complete • Priva... / Pri... / Space Control - List	2h 4m	—	—	—	—	**2h 4m** ···
Playtesting Doing • Privat... / Priva... / Space Control - List	5h 36m	6h 37m	4h 34m	—	—	**16h 48m** ···
Bug Fixing Space Control Doing • Privat... / Priva... / Space Control - List	—	—	—	8h 1m	7h 29m	**15h 31m** ···

Task / Location	Mon, Apr 6 0h	Tue, Apr 7 7h 21m	Wed, Apr 8 7h 16m	Thu, Apr 9 7h 21m	Fri, Apr 10 7h 9m	Total 29h 9m
Bug Fixing Space Control Doing • Privat... / Priva... / Space Control - List	—	7h 21m	7h 16m	7h 21m	1h 9m	**23h 9m** ···
Hand Tracking Doing • Privat... / Priva... / Space Control - List	—	—	—	—	6h	**6h** ···

Task / Location	Mon, Apr 13 8h 18m	Tue, Apr 14 7h 57m	Wed, Apr 15 7h 49m	Thu, Apr 16 8h 26m	Fri, Apr 17 7h 12m	Total 39h 44m
Bug Fixing Space Control Doing • Private... / Privat... / Space Control - List	8h 18m	3m	—	—	—	**8h 21m** ···
Teleport should work with both hands Review • Private... / Priva... / Space Control - List	—	7h 27m	—	—	—	**7h 27m** ···
Teleport should not disable rotation with other hand Review • Private... / Priva... / Space Control - List	—	27m	1h 10m	—	—	**1h 37m** ···
Teleportation targets Review • Private... / Priva... / Space Control - List	—	—	6h 38m	7h 48m	—	**14h 27m** ···
Playtesting Doing • Private... / Privat... / Space Control - List	—	—	—	37m	—	**37m** ···
Smaller circle for teleport location Review • Private... / Priva... / Space Control - List	—	—	—	—	5h 18m	**5h 18m** ···
Skyrim hand gesture Review • Private... / Priva... / Space Control - List	—	—	—	—	1h 53m	**1h 53m** ···

Week 6 & 7

1. Fixing Bugs

Yeah, not much to mention here. I learned about quite a few systems while fixing issues. There are only two bugs I couldn't tackle, one was caused by a particle system which I'm 90% sure is a Unity issue.

The other was a VR hand pose related issue which I couldn't pinpoint either. Other than that, the other issues taught me about user error and unity wonkiness.

Key Learning Points:

- Hurricane VR is a shit package, don't use it.
- Unity is full of bugs and weird behaviors!
- Even senior developers write bugs!
- In depth learning about Unity systems and logic.

Week 8

1. Hand Tracking

After fixing bugs, new content! Hand Tracking! Oh god... hand tracking...

While there's an overview of what I worked on in the timetable present on page 2, a lot of the work I did was merged with certain task. This is why some tasks that appear easier may be longer, it's likely because I worked on other systems on top of the one described.

On top of that, it was my first time working with Hand Tracking in VR and getting around the MASSIVE package that is Hurricane VR with documentation that rivals Unreal Engine's doc, slowed down my progress considerably.

Enum HVRTeleportCurve

Namespace: HurricaneVR.Framework.Core.Player

Assembly: HurricaneVR.Framework.dll

Syntax

```
public enum HVRTeleportCurve
```

Fields

Name	Description
Ballistic	
Bezier	

The key goal of this is to make Hand Tracking feel nice to use in game. Nothing worse than janky controls taking you out of the experience. In an attempt to make it feel better; I compiled a list of tasks that could potentially help.

TLDR; Hand Tracking is shit in literally every game it's implemented and if I succeed it will be the only game on the market with an actual viable Hand Tracking system.

I won't go into details about what I worked on so far as each task tackles an issue that is similar at its core:

- Hand Tracking technology is shit.
- Any hand poses where the palm is facing away from the player doesn't work.
- Any hand poses where the palm is facing the player will display the Meta menu Icon.
- Any Hand poses using more that the Index, Thumb and Middle finger can cover the ring and little finger when the palm is perpendicular to the player view.
- So, find a pose that works.
- Slap on 400 lines of code in an attempt to smooth out the controls.
- Fix the 150 issues that arise from it because Hurricane VR is shit.
- Voila, you have a slightly better Hand Tracking system.

Key Learning Points:

- I hate Hurricane VR.
- I hate Hand Tracking Technology.
- Editor Time coding.
- Control Design.

Conclusion

During these three weeks, I fixed a wide range of bugs and issues. Some tasks were more complex than others, but all contributed to expanding my technical knowledge and problem-solving skills.

I also started working on Hand Tracking system, and if it wasn't clear by know, I learned a deep hatred for Hand Tracking and Hurricane VR.

Internship Report 05

Week 10 & 11

Burgisser Dylan

Introduction

This report outlines my activities during the last two week of my internship at MoonMonster Studios. With the post release fixes behind us, it’s time to focus on new content.

Task / Location	Mon, Apr 20 8h 1m	Tue, Apr 21 7h 52m	Wed, Apr 22 8h 13m	Thu, Apr 23 8h 17m	Fri, Apr 24 0h	Total 32h 25m
Skyrim hand gesture Review • Space... / Inter... / Space Control - List	7h 35m	—	—	—	—	**7h 35m** ···
Rotation can be fully done with 1 hand Backlog • Spac... / Inte... / Space Control - List	25m	—	—	—	—	**25m** ···
Make wearables persist through scenes on the player and the crew Review • Space ... / Inter... / Space Control - List	—	6h 35m	—	—	—	**6h 35m** ···
unity build processor thing for saving persistant data Review • Space... / Inter... / Space Control - List	—	1h 17m	46m	—	—	**2h 3m** ···
fix the fucking poop hat Backlog • Spac... / Inte... / Space Control - List	—	—	26m	—	—	**26m** ···
episode 3 intro? Backlog • Spac... / Inte... / Space Control - List	—	—	52m	—	—	**52m** ···
Make hats machine work In Sprint • Spac... / Inte... / Space Control - List	—	—	6h 7m	7h 47m	—	**13h 54m** ···

Task / Location	Mon, Apr 27 7h 11m	Tue, Apr 28 8h 2m	Wed, Apr 29 8h 26m	Thu, Apr 30 7h 18m	Fri, May 1 0h	Total 30h 58m
Playtesting Doing • Space ... / Inter... / Space Control - List	1h	—	—	—	—	**1h** ···
Make hats machine work In Sprint • Spac... / Inte... / Space Control - List	6h 11m	5h 10m	—	—	—	**11h 21m** ···
Make wearables persist through scenes on the player and the crew Review • Space ... / Inter... / Space Control - List	—	2h 52m	2h 50m	—	—	**5h 42m** ···
Make Dorian hand skin interactable In Sprint • Spac... / Inte... / Space Control - List	—	—	2h 29m	2h 34m	—	**5h 3m** ···
Polish customization machine In Sprint • Spac... / Inte... / Space Control - List	—	—	3h 6m	4h 44m	—	**7h 50m** ···

1. Persistence

Following the previous week, I continued working on the hand tracking system. However, another pressing deadline quickly emerged: the cosmetic update.

With the art intern finishing new hats and glasses, my first task was to implement a persistence system that would allow players to give hats and glasses to NPCs and have those cosmetics remain equipped across levels and even after quitting the game.

My initial approach was to use prefab GUIDs for saving and loading. Each socket would store the GUID of its socketed item alongside the socket's name in a map. When loading a new level, the socket would check the map, spawn the corresponding item if one was saved, and automatically socket it. Finally save and load the map with persistence. Simple in theory.

Unfortunately, halfway through development I discovered that GUIDs were not reliable for this use case, especially in builds. Since the GUIDs were fetched at runtime, they were often inconsistent, different from editor values, or inaccessible altogether.

To solve this, I created a custom GUID system using a simple script that assigns IDs based on prefab names. This introduced a new issue: the script needed to be added to every "socketable" item.

Manually updating every prefab was not a practical option, so instead I created an editor utility that automatically scans all prefabs, checks whether they contain an HVRSocketable component, and adds my custom ID script where necessary.

To further improve consistency and reduce user error, I integrated this function into the build pipeline, so it runs automatically before each build. This ensures all required prefabs are properly configured before the project is compiled. The trade-off is slightly longer build times, but to keep flexibility, I also added an option to disable this build hook when needed.

Key Learning Points:

- Editor scripting and build pipeline integration.
- Persistence systems.

2. Cosmetic Dispenser

With hats, glasses, and hand skins now implemented, the next step was creating the cosmetic machine that allows players to spawn items and customize hand skins.

This feature brought together nearly everything I had learned so far, including animations, physics buttons, item dispensing systems, custom interactive screens, persistence, and more.

Because this machine touched so many systems at once, it became an excellent exercise in integrating existing mechanics into a polished and user-friendly feature.

Key Learning Points:

- Workflow efficiency.
- Production polish.
- Time management.

Conclusion

Over the course of these two weeks, it became increasingly clear that I was being entrusted with more important and technically demanding tasks.

Having demonstrated my abilities as both a programmer and a team member, my next area of growth is optimization. Not just in code, but in problem-solving. This means identifying the simplest and most effective solution for a given task, while carefully judging how complex a system truly needs to be.

The goal is to strike a strong balance between time, effort, and quality, while avoiding unnecessary complexity and diminishing returns. And so was the focus of this week and the ones to come.

Internship Report 06

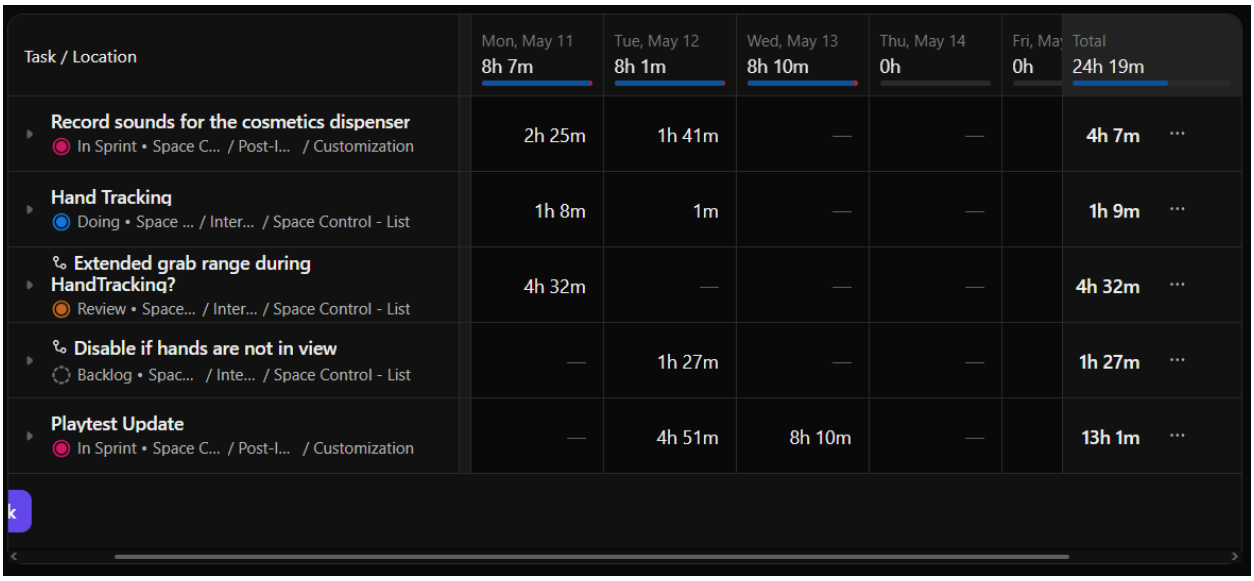
Week 12 & 13

Burgisser Dylan

Introduction

This report outlines my activities during the last two week of my internship at MoonMonster Studios. The focus being on polishing the customization updated for release, with a shorter 13th week due to the company taking Friday off on top of the Thursday.

Task / Location	Mon, May 4 7h 28m	Tue, May 5 7h 39m	Wed, May 6 7h 27m	Thu, May 7 8h 8m	Fri, May 8 7h 34m	Total 38h 18m
Polish customization machine Review • Space ... / Interns / Space Control - List	7h 28m	1h 34m	—	—	—	**9h 3m**
Implement locked hands icon. Complete • Space C... / Post-I... / Customization	—	1h 17m	—	—	—	**1h 17m**
When it opens, the hats mode needs to be selected and a hat should be spawned in... Complete • Space C... / Post-Ia... / Customization	—	13m	—	—	—	**13m**
Adjust lever to behave better. There are multiple annoyances here. Complete • Space C... / Post-Ia... / Customization	—	1h 11m	—	—	—	**1h 11m**
The sound on the hand should have some cooldown probs. Complete • Space C... / Post-I... / Customization	—	15m	—	—	—	**15m**
The hat is not socketed on the dude now. Make it so you can grab the hat and it... Complete • Space C... / Post-Ia... / Customization	—	1h 25m	—	—	—	**1h 25m**
Clarity on what is currently selected. Is it hats mode, evewe Complete • Space C... / Post-I... / Customization	—	1h 42m	54m	—	—	**2h 37m**



Task / Location	Mon, May 11 8h 7m	Tue, May 12 8h 1m	Wed, May 13 8h 10m	Thu, May 14 0h	Fri, Ma 0h	Total 24h 19m
Record sounds for the cosmetics dispenser In Sprint • Space C... / Post-I... / Customization	2h 25m	1h 41m	—	—		**4h 7m**
Hand Tracking Doing • Space ... / Inter... / Space Control - List	1h 8m	1m	—	—		**1h 9m**
Extended grab range during HandTracking? Review • Space... / Inter... / Space Control - List	4h 32m	—	—	—		**4h 32m**
Disable if hands are not in view Backlog • Spac... / Inte... / Space Control - List	—	1h 27m	—	—		**1h 27m**
Playtest Update In Sprint • Space C... / Post-I... / Customization	—	4h 51m	8h 10m	—		**13h 1m**

1. Customization Update

The primary focus during these past two weeks was polishing and finalizing the customization update in preparation for its release. My work mainly involved implementing additional features, fixing bugs and enhancing the overall user experience by adding sounds, particles, new visuals and other interactive elements to create a more polished and engaging final product.

No major new technical skills were introduced during this period as most of the work was refining existing systems and content. However, this experience provided valuable insight into how project structure and team organization directly impact development efficiency. The team is currently working across multiple projects simultaneously, often without clearly defined schedules or realistic deadlines. As a result, priorities can shift quickly, requiring rapid implementation of major systems within very limited timeframes.

This workflow can reduce overall productivity by dividing attention across multiple tasks, which affects both code quality and workflow consistency. While systems are functional, limited development time and evolving requirements can make it difficult to create reusable, scalable solutions. This highlighted the importance of clear planning, realistic timelines as well as effective leadership in maintaining both technical standards and efficient project progression.

On a personal level, this period has been useful in learning how to better balance quality expectations with time constraints. Rather than pursuing perfection in every implementation, I have focused more on delivering practical and functional solutions within the allocated timeframe. At the same time, this experience reinforced my idea about the importance of long-term sustainable workflow, as taking shortcuts early as I am tasked to do, will inevitably create technical debt and inefficiencies later in production.

While I disagree about the method, it is also clear to me that shortcuts must be taken to keep the company afloat long enough for the product to be released. Shortcuts is the lesser of two evils.

Key Learning Points:

- Improved understanding of polishing workflow.
- Better understanding of how company workflow and project structure influence productivity.
- Learned the importance of adapting code quality expectations to match development deadlines.
- Gained insight into the role of leadership, planning and task prioritization.

Conclusion

Overall, these two weeks were primarily focused on preparing the customization update for release through feature refinement, bug fixing, and overall polish improvements. While the technical challenges were less focused on learning new tools or systems, the experience provided useful exposure to real-world development constraints, team workflows, and the practical balance between quality, speed, and project expectations.

Internship Report 07

Week 14 & 15

Burgisser Dylan

Introduction

This time I will only show my second week as my first one was more of the same but separated in dozens of smaller tasks. The focus of these two weeks was on finishing up the Hand Tracking update as well as start on the Turret Minigame Update.

Task / Location	Tue, May 26 8h 4m	Wed, May 27 7h 59m	Thu, May 28 8h 51m	Fri, May 29 8h 13m	Total 33h 9m
Fix Teleportation by teleporting to closest available location instead of whatever the... Review • Space Co... / Post-la... / HandTracking	8h 4m	4h 5m	—	—	**12h 9m** ···
Improve clarity of rotation tutorial Review • Space Co... / Post-la... / HandTracking	—	3h 22m	—	—	**3h 22m** ···
Update tutorial + Range Grab visuals Review • Space Co... / Post-la... / HandTracking	—	31m	7h 9m	—	**7h 41m** ···
Set up Infinite Wave System Review • Space... / Inter... / Space Control - List	—	—	1h 41m	1h 48m	**3h 30m** ···
Cleaning up the Scene Review • Space... / Post... / Turret endless mode	—	—	—	4h 38m	**4h 38m** ···
General Turret Minigame Doing • Space... / Post... / Turret endless mode	—	—	—	1m	**1m** ···
Add Player Start/Restart Screen Doing • Space... / Post... / Turret endless mode	—	—	—	1h 46m	**1h 46m** ···

1. Hand Tracking Update

Over the past two weeks, I have been working on and off on the hand tracking update. Due to delays caused by the customization update, the focus shifted toward completing this feature as efficiently as possible. As a result, the final phase consisted mainly of last-minute polish, followed by the first playtest of the update.

The first playtest yielded a considerable amount of feedback, ranging from smaller issues that I had not yet had time to address to larger core limitations. The primary goal of the feedback was to determine whether the update was moving in the right direction:

- Did using hand tracking feel natural?
- Were players required to repeat inputs too often?
- For returning players, did the changes feel like an improvement?

The overall consensus was positive, with most testers agreeing that the update significantly improved the experience. However, the inherent limitations of hand tracking technology remain a major challenge. These limitations are not issues that can simply be solved through code.

One notable improvement was changing the rotation system to use pinching gestures between the middle and ring fingers to rotate left or right, which worked well during testing. However, these gestures would once again expose the limitations of the hand tracking technology. For example, rotating while the palm faces upward can obscure the index finger from the headset cameras. This sometimes causes the software to misidentify the middle finger as the index finger, leading to unintended actions such as opening menus instead of rotating. Unfortunately, this is a hardware and tracking limitation that cannot be fully addressed through implementation alone.

Aside from these limitations, nearly all intended features for the update are now present in the game. Following feedback and final polish, I also proposed two additional improvements that I did not initially have time to implement.

The first is a settings update aimed at maximizing player customization and accessibility for Hand Tracking. The second is a fundamental improvement to the teleportation system. Teleportation now ignores walls while still restricting movement to valid locations. This would prevent players from becoming stuck when standing too close to a wall, as the current raycast system invalidates teleportation immediately when obstructed. As a result, players currently have to awkwardly reposition or twist their arms to teleport away from walls.

Key Learning Points:

- Improved understanding of rapid prototyping and iteration under tight deadlines.
- Gained experience collecting, analysing, and applying player feedback to improve gameplay systems.

- Further developed problem-solving skills when working around hardware and software limitations. Fuck hand tracking, respectfully.

2. Turret Minigame Update

The turret minigame is a standalone experience that becomes accessible after completing the first episode, allowing players to revisit the climax sequence. This new version introduces an infinite high-score system, transforming it into a replayable arcade-style minigame.

Although hand tracking remained the priority, poor scheduling occasionally created downtime or shifted priorities. During those moments, I began work on the turret minigame update in parallel.

Over the past two weeks, my primary focus was developing a prototype of the minigame. This involved duplicating the original scene and modifying it into an infinite wave system. While the technical implementation itself was not especially difficult, it became time-consuming due to the amount of legacy content and accumulated scene complexity.

Because the scene is several years old and has gone through multiple refactors, I encountered a large amount of unused assets, inconsistent behaviours, bugs, and outdated systems. Cleaning and restructuring these elements became an important part of the process.

After successfully setting up the infinite wave functionality, I began cleaning and preparing the scene for future development. Due to the team's current workload, the prototype has not yet been playtested. As a result, I will continue developing and refining the system according to the original plan until feedback becomes available.

Key Learning Points:

- Developed a better understanding of refactoring older systems and managing technical debt.
- Gained a deeper understanding of the existing codebase and legacy implementation choices.
- Learned to manage my time to work on parallel projects more efficiently.
- Improved ability to identify inefficiencies and prepare systems for future scalability and iteration.

Conclusion

As mentioned previously, it has become increasingly clear that the team trusts my abilities, resulting in greater responsibility and more important tasks. This includes taking ownership of and leading updates such as the ones described above. Still under the supervision of my mentor of course.

I also continue to work on improving my ability to prioritize practical solutions over unnecessary complexity. While I have made good progress, I still occasionally find myself overcomplicating systems when simpler approaches would be more effective within time constraints.

Internship Report 08

Week 16 & 17

Burgisser Dylan

Introduction

The final two weeks of the internship were focused on completing the Hand Tracking update and bringing the Turret Minigame to a playable state, with the leaderboard system largely implemented. During this period, my time was divided across more than 30 tasks spanning both updates.

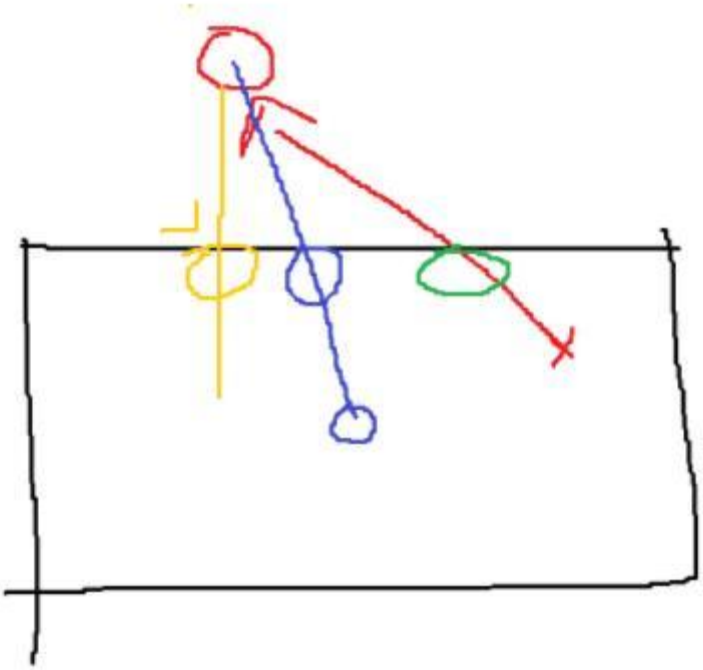
1. Hand Tracking Update

With the Hand Tracking update entering its final stretch, the first priority was extensive playtesting to identify and resolve any remaining issues.

One of the main areas that required further iteration was the teleportation system. The teleportation curve had previously been changed from a ballistic trajectory to a Bezier curve, and collision with walls had been removed. With the removal of the wall constraints, it became clear that players naturally pointed their hands further upward and forward when selecting a destination. As a result, the teleport arc often extended much farther than originally intended, causing players to frequently attempt to teleport outside the playable area.

Although the system would automatically find the nearest valid position within bounds, this often caused the teleport destination to snap aggressively to specific locations. This behaviour felt unnatural and made teleportation less predictable for players.

To address this issue, I reverted the teleportation curve back to a ballistic trajectory and corrected several related calculations. These changes immediately made the system feel more intuitive and responsive. However, another challenge remained: determining the closest valid in-bounds position when a player targeted an invalid location.



Legends for the diagram:

- Black box = Teleportable area
- Red cross = Player
- Red circle = Target teleport location
- Green circle = Current valid position found
- Blue circle = Closest position found by mapping toward the center of the area
- Yellow circle = Closest valid position to the target location

While the original system functioned correctly from a technical perspective, it often produced undesirable results. When standing near walls, players could be teleported back to their current position and smaller play areas frequently caused the system to behave unpredictably or not at all.

To improve this, I redesigned the calculation process. Using Unity's '*Closest Point'* functionality, I first determined the nearest position on the teleportable surface. However, this position was not always valid, requiring additional calculations to generate a suitable destination. After several rounds of programming, testing, and refinement, the new system consistently finds a more intuitive teleport location and provides a significantly improved user experience!

After weeks of overcoming the limitations of current hand-tracking technology, as well as adapting and extending the existing Hurricane VR implementation, the Hand Tracking update

finally reached a level of quality that exceeded the original project goals or any other game on the market offering Hand Tracking support.

Unfortunately, this success was short-lived. Just before the final round of stability testing, Meta released a platform update on June 3rd that broke hand tracking functionality across numerous applications. The issue was not limited to Space Control, it also affected other hand-tracking titles, including *First Hand*, Meta's own hand-tracking demonstration application.

The update caused the Meta system menu to open whenever users pinched their right thumb and right index finger together. Since this gesture was used extensively throughout Space Control, it interfered with core gameplay actions such as teleportation, object grabbing, and trigger interactions. As a result, final testing and deployment of the Hand Tracking update had to be postponed until a platform-level fix became available.

For the time being, development focus shifted toward completing the Turret Minigame.

Key Learning Points:

- Overcoming hardware and software limitations.
- Patience.

2. Turret Minigame Update

During the previous development period, the infinite wave system for the Turret Minigame had already been implemented. However, a considerable amount of work still remained before the minigame could be considered complete.

The remaining tasks included:

- Adding a start and restart screen
- Supporting custom wave injection for controlled progression and boss encounters
- Implementing Overcharge Mode, activated through a button press
- Restoring tower animations
- Adding conversation branching
- Creating a broken turret state
- Adding rotating information screens
- Supporting custom delays for tower events
- Implementing reverse tower animations for resets
- Developing the leaderboard system

While most of these tasks were relatively straightforward, two features proved particularly interesting from a technical perspective: the reverse tower animations and the leaderboard system.

Reverse Tower Animations

One issue discovered during testing was that tower animations would remain active after the player died. As a result, restarting the minigame on Wave 1 could leave towers in positions that belonged to much later waves, creating an inconsistent and unpolished experience.

The simplest solution would have been to replay the Wave 1 setup animation whenever the game restarted. However, this would have caused towers to instantly disappear and reappear in their starting positions, making the reset feel abrupt and unnatural.

To create a smoother transition, some form of animation blending was required. Unfortunately, the tower animations were being controlled through multiple Playable Actors and Timeline assets, making traditional blending difficult to implement.

The solution I eventually settled on was to track every tower animation that played during gameplay and replay those animations in reverse at an accelerated speed whenever the minigame was reset. While conceptually simple, implementing this required significant experimentation.

I first had to extract the relevant animation data from the Timeline and Playable systems, verify that the animations could be replayed safely, and then ensure they behaved correctly when running in reverse. These were systems I had not worked with extensively before, making the implementation a valuable learning experience.

Although I eventually succeeded, the process was not without challenges. Unity's Playable system does not particularly enjoy negative playback speeds, resulting in hundreds of assertion errors related to negative delta times. To work around this limitation, I manually evaluated the timelines, effectively bypassing the engine's internal checks and allowing the animations to play smoothly in reverse.

The final result provided a much more polished reset sequence and ensured that every restart began from a clean and consistent state.

Leaderboard System

The The leaderboard system began as what appeared to be a relatively straightforward task. However, the requirements quickly transformed it into one of the most ambitious systems developed during the internship.

Rather than creating a leaderboard exclusively for the Turret Minigame, the goal was to develop a reusable package that could be exported and integrated into future projects with minimal setup. Furthermore, the system needed to support multiple platforms and integrate with their native leaderboard solutions, including Meta, Google Play Games, iOS Game Center, and even web-based platforms such as CrazyGames. Yeah I’m still not paid for this by the way.

To achieve this, I designed a modular architecture built around a set of interfaces and service classes.

ILeaderboard<TScore, TEntry>

This serves as the primary interface for global leaderboard implementations.

Its responsibilities include:

- Submitting scores
- Retrieving leaderboard entries
- Managing communication with the underlying database service
- Providing a consistent API regardless of platform

ILeaderboardBase

A lightweight base interface that provides initialization functionality shared across all leaderboard implementations.

INativeLeaderboard

This interface is responsible for platform-specific leaderboards.

Its responsibilities are intentionally limited to:

- Initialization
- Score submission

This allows platform-specific functionality to remain separated from the global leaderboard implementation.

LeaderboardService

The central entry point for all leaderboard interactions.

Its responsibilities include:

- Registering leaderboards
- Storing leaderboard instances
- Managing score submissions
- Retrieving entries
- Synchronizing global and native leaderboard data

Internally, it maintains a collection of registered leaderboard pairs and acts as the primary access point for gameplay systems.

LeaderboardPair<TScore, TEntry>

This class combines both the global and native leaderboard implementations into a single synchronized unit.

Its responsibilities include:

- Initializing both leaderboards
- Submitting scores to each system
- Synchronizing data between implementations
- Retrieving information from the appropriate source

LeaderboardHandle<TScore, TEntry>

To simplify usage, leaderboard registration returns a strongly typed handle.

This handle:

- Stores leaderboard identifiers and associated types
- Provides direct access to leaderboard functionality
- Reduces complexity for gameplay systems

Instead of interacting with the service directly, developers can simply write:

_leaderboard.SubmitScore(score);

This abstraction significantly improves usability while maintaining type safety.

Although the leaderboard system required considerably more design and engineering effort than initially anticipated, the final result is a flexible and reusable framework capable of supporting multiple platforms and future projects with minimal modification.

With both the Hand Tracking update and the Turret Minigame reaching their final stages, this work marked the conclusion of my internship development tasks.

Conclusion

Ending Ending on a high note, this internship has been quite the adventure. From start to finish, I had the opportunity to work on increasingly complex systems while learning many different aspects of professional game development. It was an invaluable experience that helped prepare me for the challenges that lie ahead, and looking back, it is clear how much I have grown both technically and professionally.

Throughout the internship, I faced challenges that could only be experienced in a real production environment. From working with a six-year-old codebase and tight deadlines to learning new technologies and overcoming bad habits, every obstacle provided an opportunity

to improve. Along the way, I developed a wide range of new skills and gained a much deeper understanding of how game development functions within a professional studio.

At the same time, this internship showed me how much there is still to learn. While I have grown considerably as a developer, I am well aware that there are many talented developers with years of experience ahead of me. Rather than finding that discouraging, I find it motivating. My curiosity and eagerness to learn continue to drive me forward, and I look forward to further developing my skills in the years to come.

Reflecting on my personal growth, one of the biggest challenges I faced was learning when to pursue perfection and when to prioritize practical solutions. I often found myself searching for the most reusable, expandable, and elegant solution possible, even when a simpler approach would have been more appropriate given the available time and resources. Over the course of the internship, I became much better at balancing code quality with productivity, though it remains an area I intend to continue improving throughout my career.

As a passionate gamer above all else, my love for video games remains my strongest source of motivation. It drives me to continually improve my work and strive to create experiences that players genuinely enjoy. Combined with my curiosity and desire to learn, this passion pushes me to deliver work that exceeds expectations whenever possible. It is a mindset that I carried throughout my education at Digital Arts and Entertainment and one that I will continue to bring into my professional career.

Looking back on my contributions, I believe my work was genuinely valued by the company. During my internship, I contributed to every post-launch update of the game, and in some cases was responsible for implementing entire features independently. With each task, I demonstrated my ability to take ownership of complex systems and deliver solutions that met the standards of a project built over six years of development.

One example of this trust was the opportunity to design and implement a reusable leaderboard framework intended not only for the current project but also for future projects developed by the company. Being entrusted with a system of that scope demonstrated confidence in both my technical abilities and my understanding of software architecture.

Perhaps the clearest indication that my contributions were appreciated came at the conclusion of the internship, when I was offered a summer position as a student developer within the company. For me, this serves as a strong validation of the effort, dedication, and growth demonstrated throughout the internship.

However, rather than relying solely on my own perspective, I would like to conclude with feedback received from our artist:

> "For me personally, I feel like you guys were both a great help through some very turbulent periods (launch, 2 layoffs).
>
> More so than previous interns (though that wasn't necessarily their fault, we just had a bigger team and more time for onboarding with past internships)."

— Lina

Looking back, this internship was not only an opportunity to develop technical skills, but also an opportunity to prove to myself that I can contribute meaningfully within a professional development team. It has reinforced my passion for game development and given me confidence as I take the next steps in my career.

Conclusion

Ending on a high note, this internship has been quite the adventure. From start to finish, I had the opportunity to work on increasingly complex systems while learning many different aspects of professional game development. It was an invaluable experience that helped prepare me for the challenges that lie ahead, and looking back, it is clear how much I have grown both technically and professionally.

Throughout the internship, I faced challenges that could only be experienced in a real production environment. From working with a six-year-old codebase and tight deadlines to learning new technologies and overcoming bad habits, every obstacle provided an opportunity to improve. Along the way, I developed a wide range of new skills and gained a much deeper understanding of how game development functions within a professional studio.

At the same time, this internship showed me how much there is still to learn. While I have grown considerably as a developer, I am well aware that there are many talented developers with years of experience ahead of me. Rather than finding that discouraging, I find it motivating. My curiosity and eagerness to learn continue to drive me forward, and I look forward to further developing my skills in the years to come.

Reflecting on my personal growth, one of the biggest challenges I faced was learning when to pursue perfection and when to prioritize practical solutions. I often found myself searching for the most reusable, expandable, and elegant solution possible, even when a simpler approach would have been more appropriate given the available time and resources. Over the course of the internship, I became much better at balancing code quality with productivity, though it remains an area I intend to continue improving throughout my career.

As a passionate gamer above all else, my love for video games remains my strongest source of motivation. It drives me to continually improve my work and strive to create experiences that players genuinely enjoy. Combined with my curiosity and desire to learn, this passion pushes me to deliver work that exceeds expectations whenever possible. It is a mindset that I carried throughout my education at Digital Arts and Entertainment and one that I will continue to bring into my professional career.

Looking back on my contributions, I believe my work was genuinely valued by the company. During my internship, I contributed to every post-launch update of the game, and in some cases was responsible for implementing entire features independently. With each task, I demonstrated my ability to take ownership of complex systems and deliver solutions that met the standards of a project built over six years of development.

One example of this trust was the opportunity to design and implement a reusable leaderboard framework intended not only for the current project but also for future projects developed by the company. Being entrusted with a system of that scope demonstrated confidence in both my technical abilities and my understanding of software architecture. Perhaps the clearest indication that my contributions were appreciated came at the

conclusion of the internship, when I was offered a summer position as a student developer within the company. For me, this serves as a strong validation of the effort, dedication, and growth demonstrated throughout the internship.

However, rather than relying solely on my own perspective, I would like to conclude with feedback received from our artist:

> "For me personally, I feel like you guys were both a great help through some very turbulent periods (launch, 2 layoffs). More so than previous interns (though that wasn't necessarily their fault, we just had a bigger team and more time for onboarding with past internships)."

— Lina

Looking back, this internship was not only an opportunity to develop technical skills, but also an opportunity to prove to myself that I can contribute meaningfully within a professional development team. It has reinforced my passion for game development and given me confidence as I take the next steps in my career.


MOONMONSTER
studios
INTERNSHIP
DYLAN BURGISSER &
JENS DE BRABANTER


MOONMONSTER
studios
INTERNSHIP
DYLAN BURGISSER &
JENS DE BRABANTER

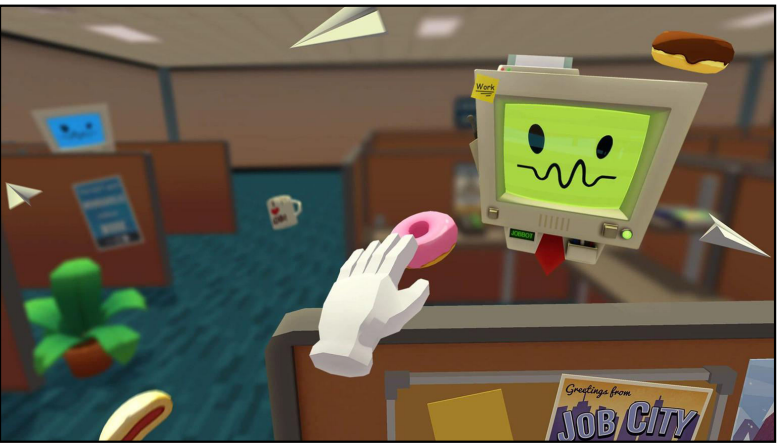

?
The Team

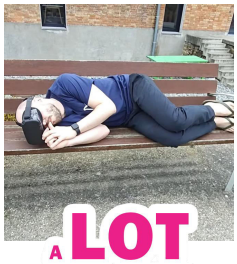

Lina
Joins the battle!


Simon
Joins the battle!

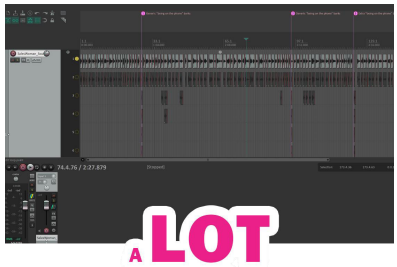

Brian & Joran
The Fallen Soldiers

MOONMONSTER studios



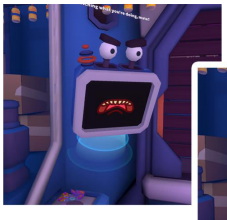


A **LOT**
of playtesting



A **LOT**
of voice lines

SPACE CONTROL
PRE-LAUNCH

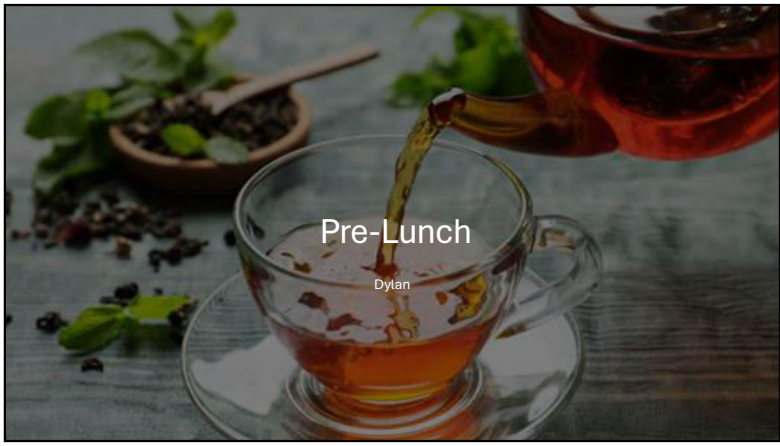


SPACE CONTROL
PRE-LAUNCH


SPACE
CONTROL
PRE-LAUNCH


SPACE
CONTROL
PRE-LAUNCH

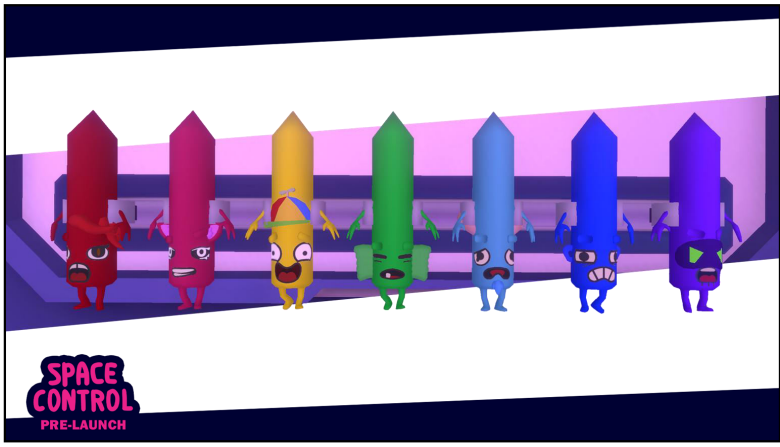

SPACE
CONTROL
PRE-LAUNCH

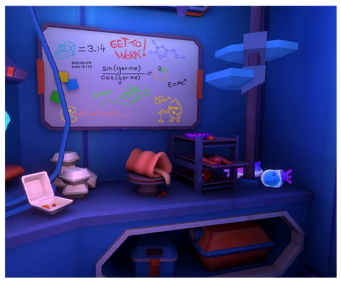

Pre-Lunch
Dylan

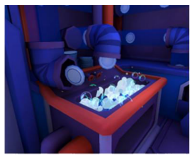

SPACE
CONTROL
PRE-LAUNCH


SPACE
CONTROL
PRE-LAUNCH

Task / Location	Mon, Mar 9 7h 42m	Tue, Mar 10 7h 40m	Wed, Mar 11 8h 21m	Thu, Mar 12 7h 42m	Fri, Mar 13 8h 19m	S C	Total 39h 47m	
AudioCue implementations episode 1 Complete • Spac... / Int... / Space Control - List	5h 52m	—	—	—	—		5h 52m	···
Vibration probe should only vibrate butt Complete • Spac... / Int... / Space Control - List	1h 22m	—	—	—	—		1h 22m	···
Bobblehead Complete • Spac... / Int... / Space Control - List	27m	1h 55m	—	—	—		2h 22m	···
Balls Complete • Spac... / Int... / Space Control - List	—	5h 45m	1h 26m	—	—		7h 11m	···
Tambourine Complete • Spac... / Int... / Space Control - List	—	—	5h 28m	—	—		5h 28m	···
Baby Shaker Complete • Spac... / Int... / Space Control - List	—	—	1h 25m	6h 12m	—		7h 37m	···
Vibration probe should be stronger at the start, but weaker over time Complete • Spac... / Int... / Space Control - List	—	—	2m	—	42m		45m	···
Xylophone Complete • Spac... / Int... / Space Control - List	—	—	—	1h 30m	7h 37m		9h 7m	···







SPACE CONTROL
PRE-LAUNCH

SPACE CONTROL
PRE-LAUNCH

SPACE CONTROL
PRE-LAUNCH

- The FOV during this sequence was frustrating. If I looked up, I couldn't see the drum set. If I looked down, I couldn't see the notes. If I stepped back to see better, I couldn't reach the drum set. I ended up missing beats simply because I couldn't properly see where I was hitting.

SPACE CONTROL
PRE-LAUNCH

SPACE CONTROL
PRE-LAUNCH

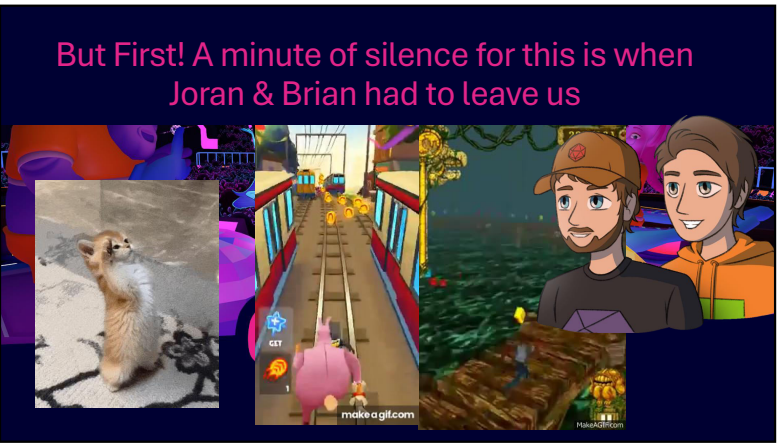
SPACE CONTROL
PRE-LAUNCH

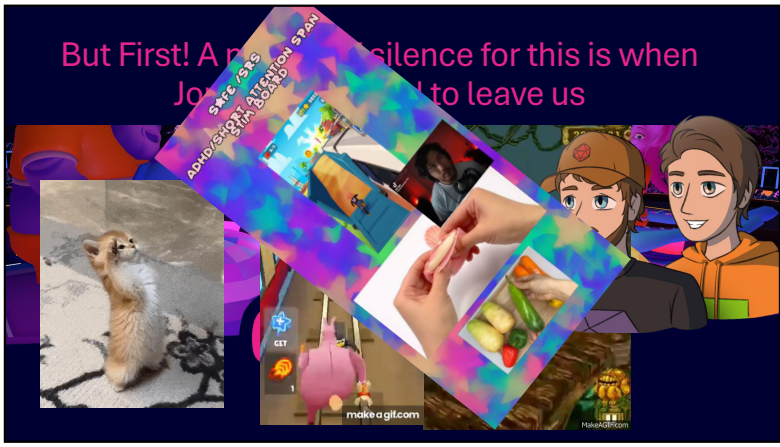
STAGES OF MALE BALDNESS PATTERN

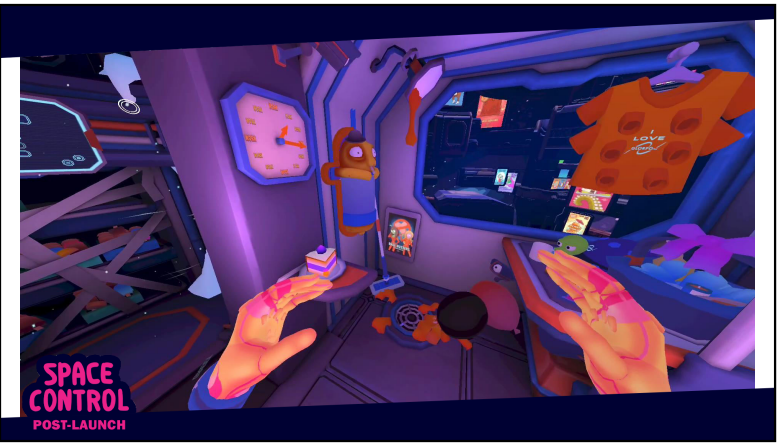
Type 1 | Type 2 | Type 3 | Type 4

But First! A minute of silence for this is when Joran & Brian had to leave us

But First! A minute of silence for this is when Joran & Brian had to leave us


But First! A minute of silence for this is when
Joran & Brian had to leave us
make a gif.com


But First! A
silence for this is when
Jo
to leave us
ADHD/SHORT ATTENTION SPAN
SMA BOARD
make a gif.com

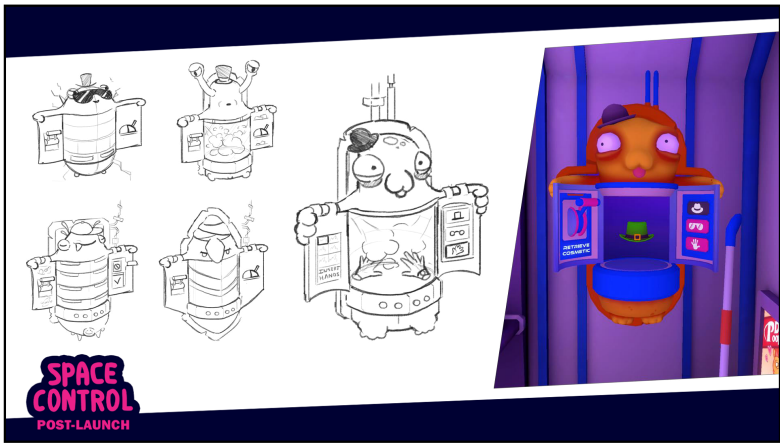

SPACE
CONTROL
POST-LAUNCH


GLORPO INC
SPACE
CONTROL
POST-LAUNCH


SPACE
CONTROL
POST-LAUNCH


BEST FLAVOR EVER
Jelly Kiss
GLORPO INC
SPACE
CONTROL
POST-LAUNCH

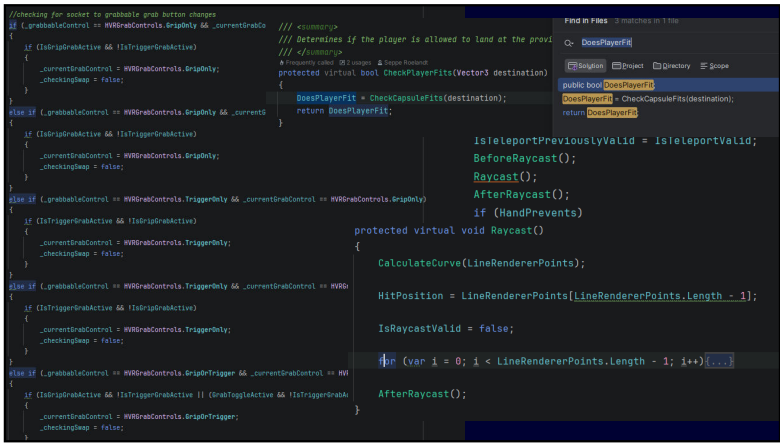

SPACE
CONTROL
POST-LAUNCH


SPACE
CONTROL
POST-LAUNCH


SPACE
CONTROL
POST-LAUNCH


SPACE
CONTROL
POST-LAUNCH

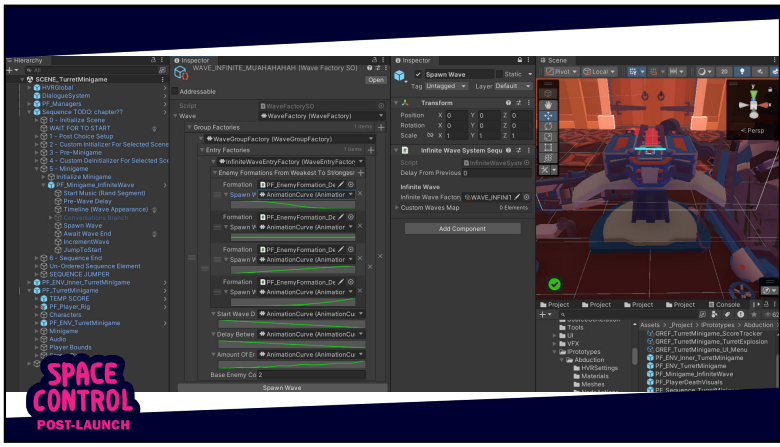

SPACE
CONTROL
POST-LAUNCH

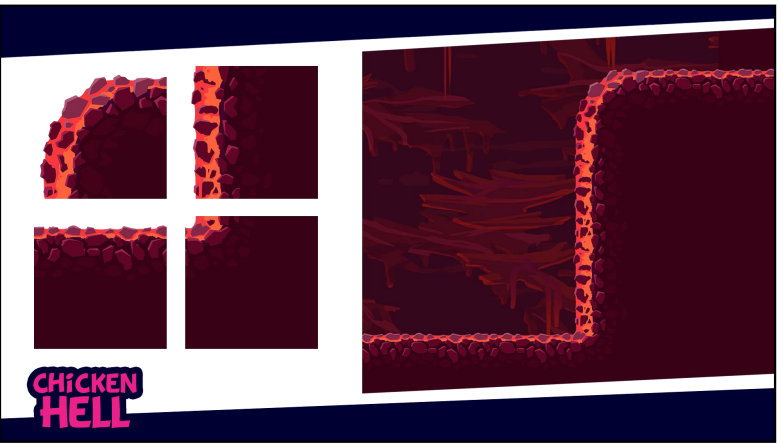

//checking for socket to grabbable grab button changes
/// <summary>
/// Determines if the player is allowed to land at the provi
/// </summary>
protected virtual bool CheckPlayerFits(Vector3 destination)
{
DoesPlayerFit = CheckCapsuleFits(destination);
return DoesPlayerFit;
}
IsTeleportPreviouslyValid = IsTeleportValid;
BeforeRaycast();
Raycast();
AfterRaycast();
if (HandPrevents)
protected virtual void Raycast()
{
CalculateCurve(LineRendererPoints);
HitPosition = LineRendererPoints[LineRendererPoints.Length - 1];
IsRaycastValid = false;
for (var i = 0; i < LineRendererPoints.Length - 1; i++){...}
AfterRaycast();
}
Find in Files
DoesPlayerFit
public bool DoesPlayerFit
DoesPlayerFit = CheckCapsuleFits(destination);
return DoesPlayerFit;

Hand Tracking!

- Teleporting with both hands
- Teleportation + rotation simultaneously
- Teleportation snap targets
- Smaller circle
- Various Hand gestures: skyrim + snaping + pistol
- Rotation fully on one hand
- Grab tweak+ grab ranges
- Force grab -> Disable hands not in view + locking
- Gradual trigger

SPACE CONTROL
POST-LAUNCH





MOONMONSTER studios

INTERNSHIP CONCLUSION

Jens De Brabanter

Technical Skills

- Unity
- VR
- C#
- Programming

Soft Skills

- Time Management
- Multi-project tasking
- Manage Complexity
- Communication

OPERATIE ZONNESLAG