
# Capstone Documentation

Nadine Carillo

Capstone Proposal

"How can I create game aesthetics to be used for a 2.5D escape room game, with purposeful environmental interactions, to reflect the experience of control?"

# Contents

**Semester 2**

# Winter Break 1: 3D modelling extra wires, lockers and extras

For the first week into winter break, my focus was finishing off the rest of the 3D models I'd have to make for the cafeteria, the showers and the infirmary. Following the proposed aesthetic for the environment, mentioned in our pitch deck, I tried to keep to the grimy feel by picking reference images that showcase abandoned or retro aesthetic.

My wire designs (Figure 1.2) are inspired by *Poppy Playtime - Chapter Two's* (2021) ceiling designs. I used this game as a reference because the game environment there is based off an underground toy factory which parallels to our game's design, an underground prison.

The wires don't need to all connect but they do need to look like they go somewhere, from what I've seen in each scene that I explored in game, they just need the illusion that they are going somewhere (Figure 1.1). I also took inspiration from the wire 3D models that had sections in between them, I assume are for holding the wires and pipes in place, for variety.

To create the loose wires, I used the EP curve tool to create wavy lines (Figure 1.4), then extruded and converted them into polygons (Figure 1.3). I did this approach to mimic the unpredictability of the wires' nature to curl at random. I think the messier the appearance, the more run down and abandoned the prison will appear to be.

Following the two reference images (Figure 1.5) of the locker I wanted to make; I used a simple cube which I scaled to make it longer and created a simple door, that's separate to the main locker if we decide to make the locker a hiding place later (Figure 1.4).

Figure 1.1 - Images of the wires in *Poppy Playtime - Chapter 2* (2021).

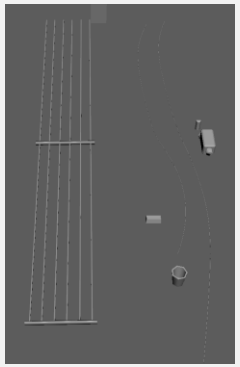


Figure 1.2 - 3D model of wires.

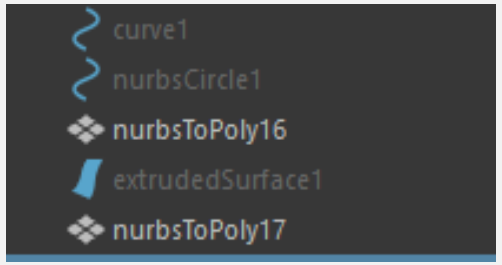



Figure 1.3 - Screenshots of extrued surface.

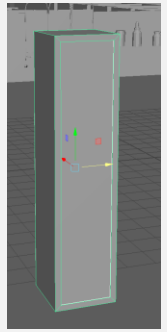


Figure 1.4 - 3D model of locker.

Figure 1.5 - Locker reference images.

# Winter Break 1: 3D modelling Cafeteria

When 3D modelling the utensils, I picked a plastic spoon as reference. Which not only shows how poorly managed the prison is but also shows control, which tackles the focus of my research question. The prisoners are given plastic utensils (Figure 2.1) to ensure that the prisoners can't attack staff, the lunch lady in this case, while they're on shift and to ensure that they don't harm themselves or others in the prison with the utensils. Safer materials like plastic and paper were encouraged instead of giving them normal utensils that have use hard materials like metal or steel. After further research, I found multiple accidents where Inmates would escape or harm themselves with a plastic spoon, so I plan to change the material to paper through Substance Painter instead of 3D modelling a new spoon, for time efficiency. It's not going to be usable anyway (Figure 2.2).

But for food trays I used references (Figure 2.3) that showcase rustic metal trays. I chose to keep it this way because I wanted the design to be cohesive to the prison looking old fashioned and uncomfortable. Same goes for the food heater (Figure 2.5), and the cafeteria rubbish bins (Figure 2.7); my reasoning for those reference images are the same. I used the symmetry tool in maya to make the vertices even on each side of the food tray.

For the cafeteria tables (Figure 2.9), I used a prison cafeteria reference image (Figure 2.10), to have room for making those rusty as well. I think this rust approach also encourages the players to know that the prison isn't cared for and is old. I plan to make the utensil holder low budget, so I chose a reference image for the rubbery utensil holders (Figure 2.11). I added a few edge loops to imitate the look of the utensil holder and soften the edges where the utensil holder holds the utensils (Figure 2.12).

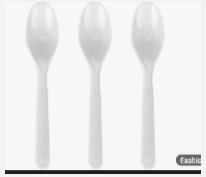


Figure 2.1 - Plastic spoon reference.

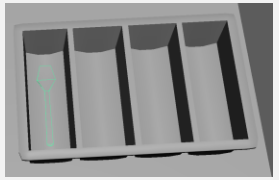


Figure 2.2 - 3D model of spoon.

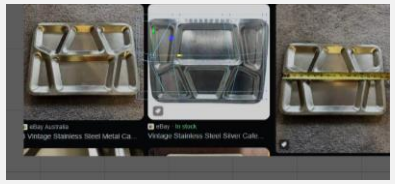


Figure 2.3 - Reference of metal tray.

Figure 2.4 - 3D model of tray.

Figure 2.5 - Food Heater Reference.

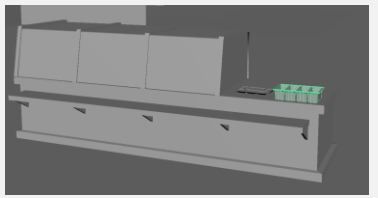


Figure 2.6 - 3D model of food heater.


Figure 2.7 - Rubbish bin reference.

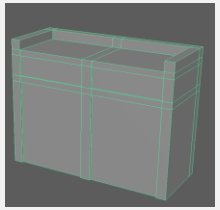


Figure 2.8 - Rubbish bin 3D model.

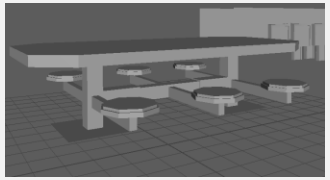


Figure 2.9 - Prison table 3D model.

Figure 2.10 - Prison table/Cafe reference

Figure 2.11 - Utensil holder reference.

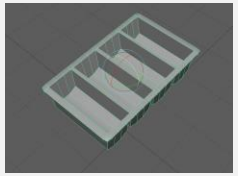


Figure 2.12 - Utensil holder 3D model.

# Winter Break 2: 3D modelling showers

Picking a reference with multiple shower drains (Figure 3.1), I went ahead and scaled a 2D plane with 1 face to act as a canvas for adding detail into substance painter later (Figure 3.2). This way, I can change the design of the shower drain if I choose to, without having to 3D model all the tiny holes which is unnecessary because this is only for decoration purposes.

I chose a reference image (Figure 3.3) of a bar of soap that looked like it was made from fat and oils, I think it's best to let the player imagine what type of fat the soap is made of.

I softened the edges of the cube to give the illusion that it's rounded on the edges (Figure 3.4). Using this old shower room reference image (Figure 3.5) as inspiration for the showers, I created a simple shower head, a soap holder and duplicated one of the buttons used for the prison sinks (Figure 3.6), to recreate the look. I chose this approach, because I already have shower walls 3D modelled from last semester. So, these pieces are the few that will bring the showers together.

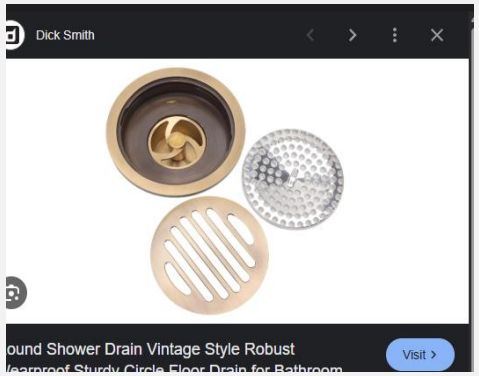



Figure 3.1 - Shower drain reference.

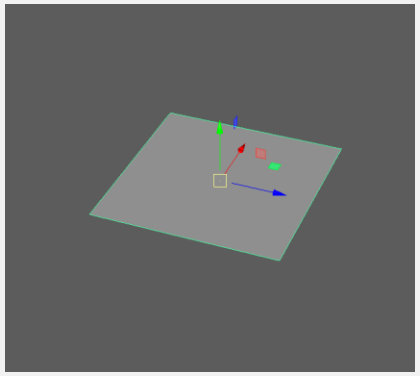


Figure 3.2 - Shower drain 3D model.

Figure 3.3 - Soap reference.

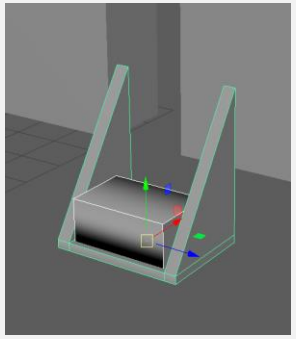


Figure 3.4 - Soap 3D model.

Figure 3.5 - Showers reference.

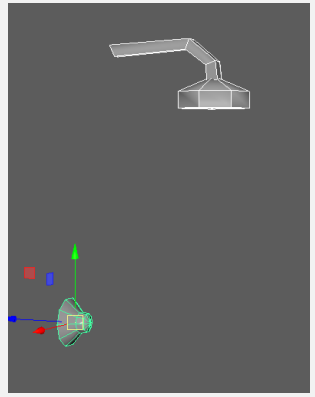


Figure 3.6 - Showers 3D model.

# Winter Break 2: 3D modelling Infirmary

The syringe 3D model (Figure 4.1) is inspired by metal vintage syringes (Figure 4.2); they fit our prison because of its aesthetically pleasing appearance that fits our already gross prison. The medicine cabinet reference image (Figure 4.3) I picked looked vintage and retro, so I used that as the main reference (Figure 4.4) before making more potential infirmary 3D models (Figure 4.5) like the sink one and a duplicatable medicine counter.

The first aid kit (Figure 4.6), curtain dividers (Figure 4.8) and stool (Figure 4.10) are reference images that show retro or vintage versions of the items.The first aid kit (Figure 4.6), red would go well with the walls of the prison because they'd match. The chosen colour, blue, of the cabinets and the floor would contrast with this so it wouldn't make it hard to find.

I chose these reference images because they all work to help create the abandoned, unsanitary, grimy look we are going for which shows on the medical dividers (Figure 4.9) and stools (Figure 4.11). With the amount of metal objects and wooden objects , there's room to add rust and mold to them to showcase deterioration.

Figure 4.1 - 3D model of Syringe.

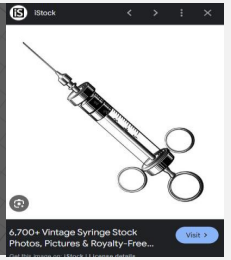



Figure 4.2 - Syringe reference.

Figure 4.3 - Medical cabinet reference.

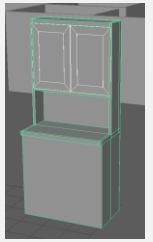


Figure 4.4 - Medical cabinet 3D model.

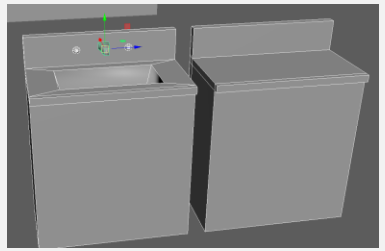


Figure 4.5 - Medical cabinet 3D model variations.


Figure 4.6 - First aid kit reference.

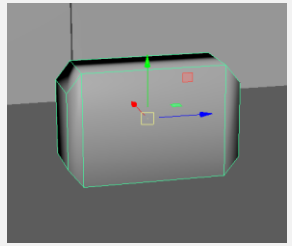


Figure 4.7 - First aid kit 3D model.

Figure 4.8 - Medical divider reference.

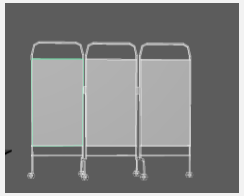


Figure 4.9 - Medical divider 3D model.

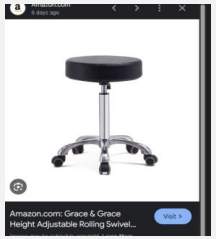



Figure 4.10 - Stool reference.

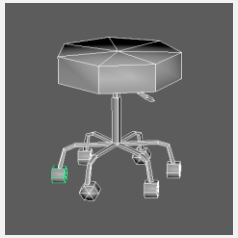


Figure 4.11 - Stool 3D model.

# Winter Break 2: UV unwrapping and texturing Cafeteria

This week, I UV unwrapped (Figure 5.1) and textured (Figure 5.2) the food heater, tables, and rubbish bins (Figure 5.3-5.4) from semester 1. For the food heater, I chose orange, one of the warmer colours from our Miro board.

After doing some research to improve my UV/Texture workflow, I set off to make the glass separate from the food heater part, so that I can add transparency to the glass later in Unity instead of making it transparent in Substance Painter because the last time I had the frame and the glass as one mesh. So, when I made it transparent in Unity, it made the whole model transparent which is not ideal because not the entire mesh is supposed to be transparent.

As pre-planned, I made the tables, rubbish bin look as if they're a part of an abandoned prison. I used an alpha shape "Grunge Stains Small Hollow" (Figure 5.5) and smudged it, to create rust and gross food stains on the 3D models, so that it would fit the grimy and grungy feel we are going for in the underground prison.

I also UV unwrapped some 2D planes (Figure 5.6), for making café and infirmary posters for decoration later. I chose to do this so that I can add more life to the cafeteria and the infirmary. It can also be a way to show control again through the information on the posters.

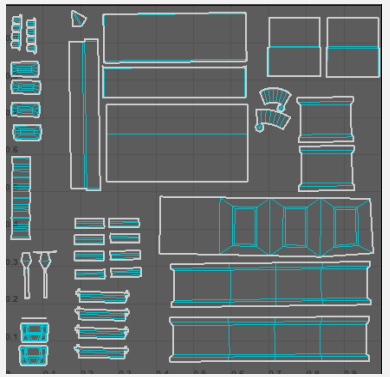


Figure 5.1 - UV map of Food heater.

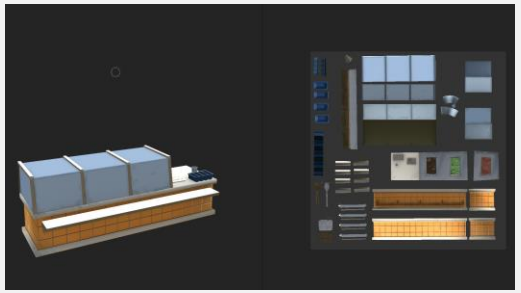


Figure 5.2 - Textured food heater.

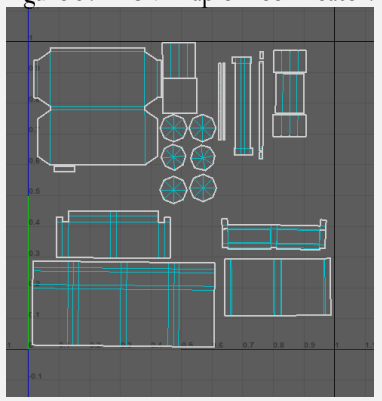


Figure 5.3 - UV map of tables and bin.

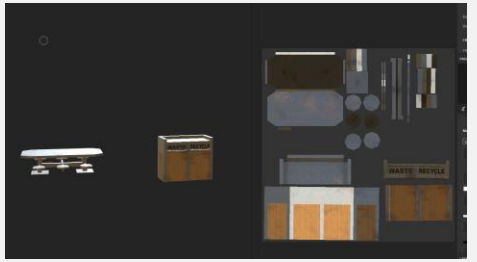


Figure 5.4 - Textured tables and bin.


Figure 5.5 - Grunge Stains Small Hollow Alpha brush.

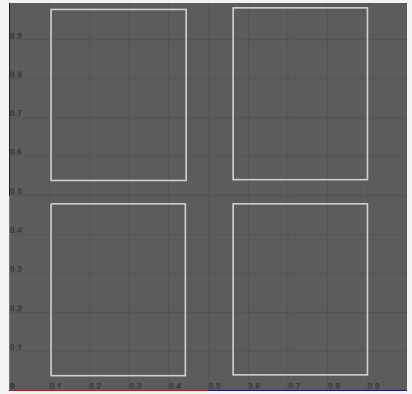


Figure 5.6 - UV map of posters.

# Winter Break 3: UV unwrapping and texturing lockers/wires

Next I unwrapped and textured the CCTV camera, wires (Figure 6.1-6.2), warden plaque, keycard scanner and lockers (Figure 6.3-6.4) from semester 1. For the food heater, I kept to the orange part of the colour palette. After doing some research to improve my UV/Texture workflow, I set off to make the glass separate from the food heater part, so that I can add transparency to the glass later in Unity instead of making it transparent in Substance Painter because the last time I had the frame and the glass as one mesh, so when I made it transparent in Unity, it made the whole model transparent which is not ideal for immersion. It can distract the player from the game.

For the CCTV camera, I used the same plastic texture I used for the computer and keyboard. It's a tiny detail but it symbolises that they're connected. I thought making the lockers blue would be a nice touch to the green and off-white background instead of a bright red. It still follows our group's colour palette and vision for the game. As pre-planned, I made the tables, rubbish bin look as if they're apart of an abandoned prison. I used a sand stamp to create rust and gross food stains on the 3D models, so that it would fit the grimy and grungy feel we are going for.

I also accidentally separated the UV maps (Figure 6.5), cause the wire and the lockers had two separate materials on them. I'm going to prevent this in the future by making sure that the group of 3D objects I export out share the same material, so there's no overlap.

The warden plaque is inspired from wooden plates with gold indents in them (Figure 6.6). It shows that the warden prioritises buying nice things for themself instead of using the money to renovate the prison.

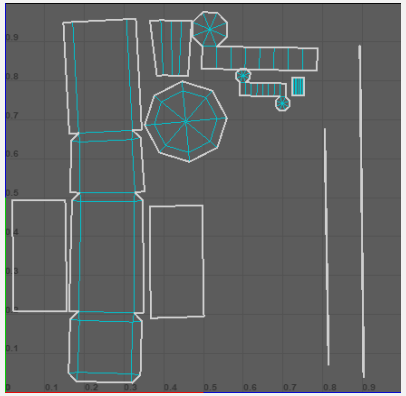


Figure 6.1 - UV map Camera and wires.

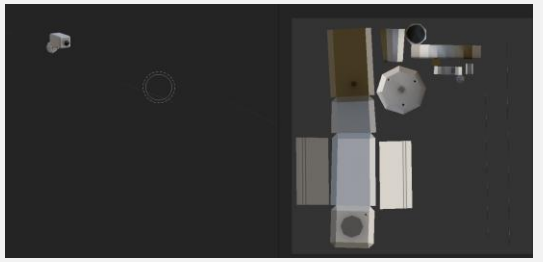


Figure 6.2 - Textured Camera and wires.

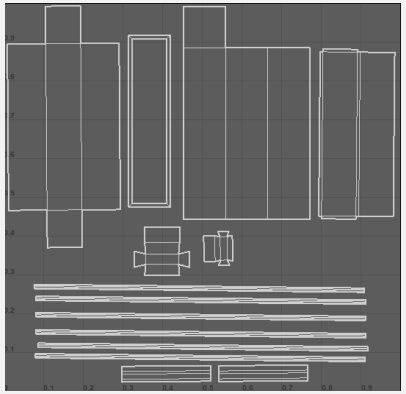


Figure 6.3 - UV map of Staff Hallway clutter.

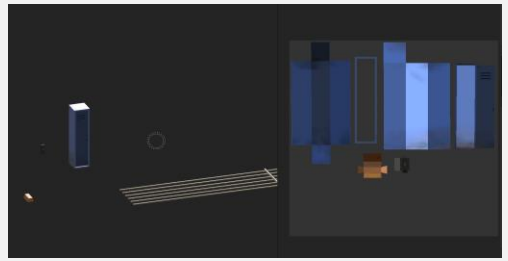


Figure 6.4 - Textured lockers and plate.

Figure 6.5 - Textured wires.

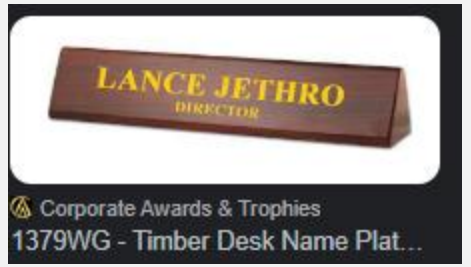



Figure 6.6 - Plate reference.

# Winter Break 4: UV unwrapping infirmary and showers

I also finished UV unwrapping the remainder 3D models that were previously assigned to our former teammate. These 3D models were part of the infirmary and the showers.

The medicine, syringe and first aid kit (Figure 7.1), are all grouped together into one UV map because they're the little details and clutter that you can find in an infirmary.

I put the stool and the medical room dividers into one UV map (Figure 7.2), because they both share the same material, which is metal. In my opinion, It's more efficient to keep 3D models that share the same materials together.

The showers also are going to have a rustic metal material on majority of the parts except for the soap (Figure 7.3), the reason why I didn't group it with the room dividers and stool is because they're from different rooms. If I had combined them together, there wouldn't have been room for the shower 3D models to have enough detail because their UV would be too small.

The last group I made was the medicine cabinets (Figure 7.4) with all their iterations, I made sure to group the areas that would share the same material as the glass being in one corner of the UV map and the metal on the other side.

I'm planning to use materials that can have mold and rust appear on them, so that's why I'm planning to use the same materials; wood, glass and metal on a lot of my 3D models.

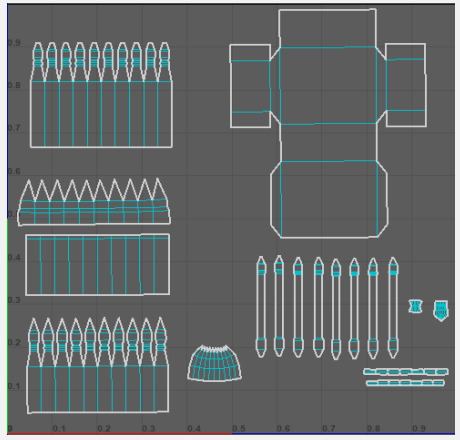


Figure 7.1 - UV map of medicine.

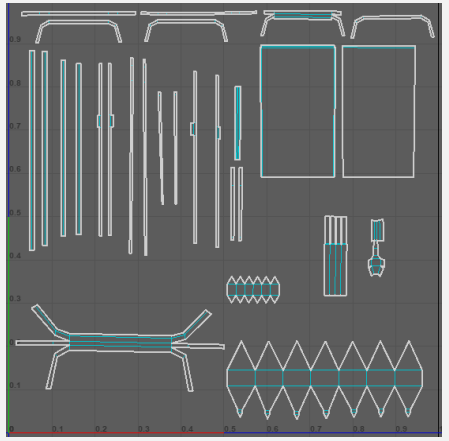


Figure 7.2 - UV map of medical dividers.

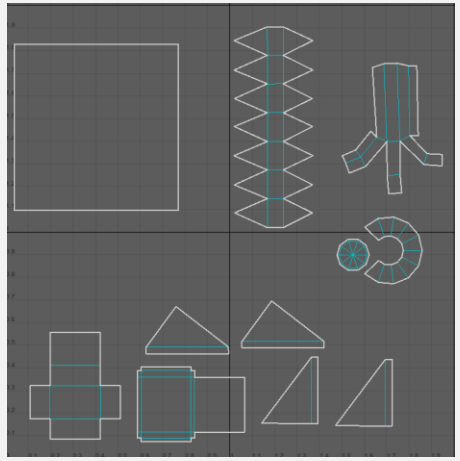


Figure 7.3 - UV map of shower essentials.

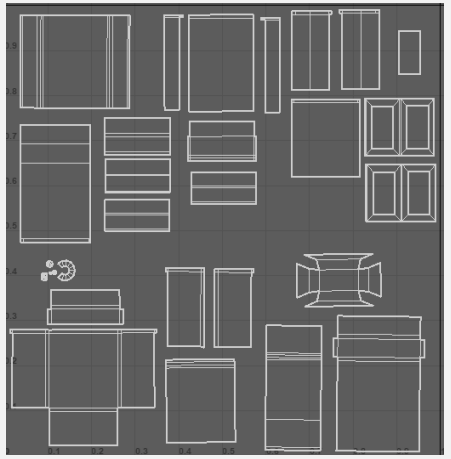


Figure 7.4 - UV map of medical dividers.

# Winter Break 5: New game plan and coding door with key

Unfortunately, one of our team members decided to leave our group, due to unexplained circumstances. 2 weeks before winter break ended our remaining team had to abruptly change the direction of the game (Figure 8.1). Everything onwards from this week will be following our new game concept.

Since it's too late to start anew, I'll be continuing to use the environment assets I've already been working on which means we'll be keeping the retro and abandoned prison setting, minus the slime and grime, and keeping our research questions the same. All the code and research I've done, will be used in this new game, wherever it can fit and if it can't it'll be scrapped.

Our back up plan and narrative can be found in this miro board: https://miro.com/app/board/uXjVIy16_Pc=/?share_link_id=608069584583

Our new chosen idea is still following an escape room concept except it follows a narrative where the player is a prisoner who needs to find a way to escape after finding out that their obsessive ex is being transferred to their prison. We wanted to choose a fun, simple and silly idea while being able to focus on our specialisations, animation and 3D modelling. We also did this to avoid the unnecessary stress and time that would go into fixing our research.

It's taken a dark comedy turn, the environment isn't controlled by the snail warden anymore, it's controlled by the obsessive ex of the prisoner, which the player is made aware of through creepy love notes scattered all over the map.

Moving forward, I worked on coding a door (Figure 8.2-8.3) to open after collecting a game object. This is a useful game mechanic that can be duplicated to control the direction of where the player can explore in later rooms. It was my first time animating doors in Unity but I found that this tutorial helped cover the basics of how to trigger an animation once a game object was collected. This tutorial uses a tag and a simple OnTriggerEnter function to make the door open.

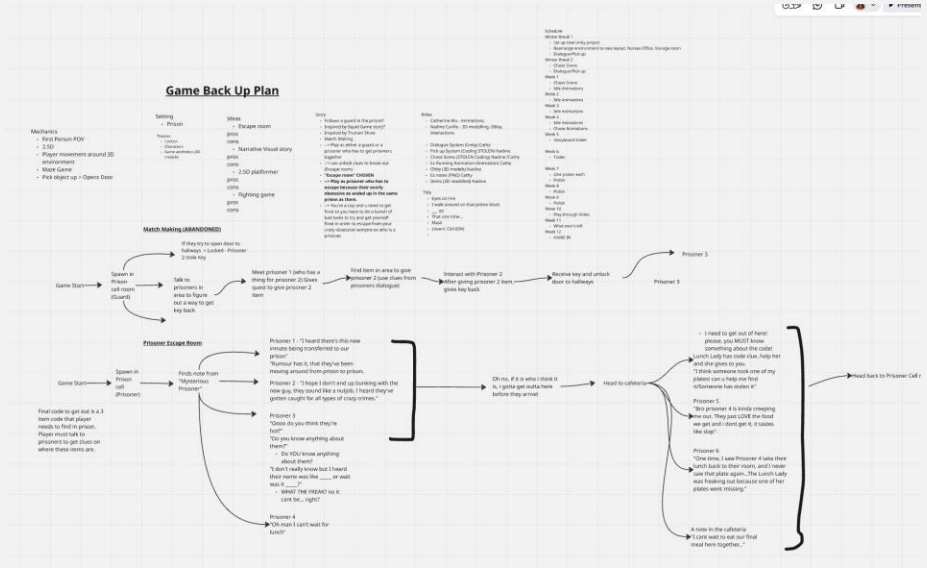



Figure 8.1 - Screenshot of game back up plan.

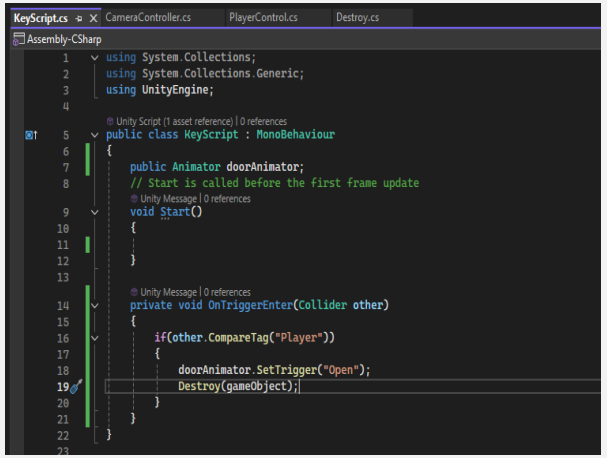



Figure 8.2 - Screenshot of Key Script.

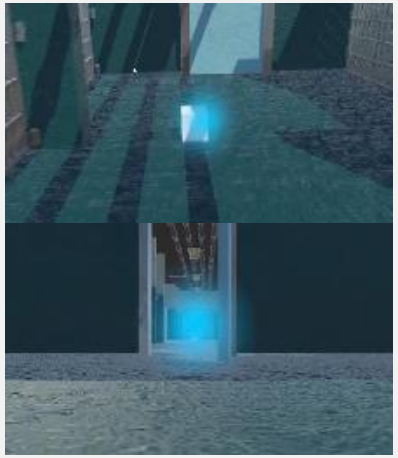


Figure 8.3 - Screenshots of key to open door gameplay.

# Winter Break 5: New Level design plan

Although our game's art direction is majorly stylised, I still wanted the opinion of someone who's been to real prisons, fortunately my family friend works in this field so they were able to tell me useful information that I could apply to my new level design. Some recommendations that she gave me to make the design (Figure 9.1) flow more was to set the prison cells in the centre of the cell room instead of it being on the sides in the last level design (Figure 9.2).

So that it looks and feels a bit more realistic even if it's a small-scale design. This recommendation I took because I want the player to be able to navigate their way around but in a way that doesn't feel super restricted but there is control because the areas that the prisoner can access are close by and anything that leads to staff areas are locked.

I noted down potential rooms that the player can run into when they get teleported into the chase scene, staff hallway, underground courtyard, reception and carpark. I thought the idea of the environment changing while the prisoner runs would make the game more interesting. Currently though, it's a bit ambitious due to the time constraints, so to downscale, I'd probably just pick one of the four ideas.

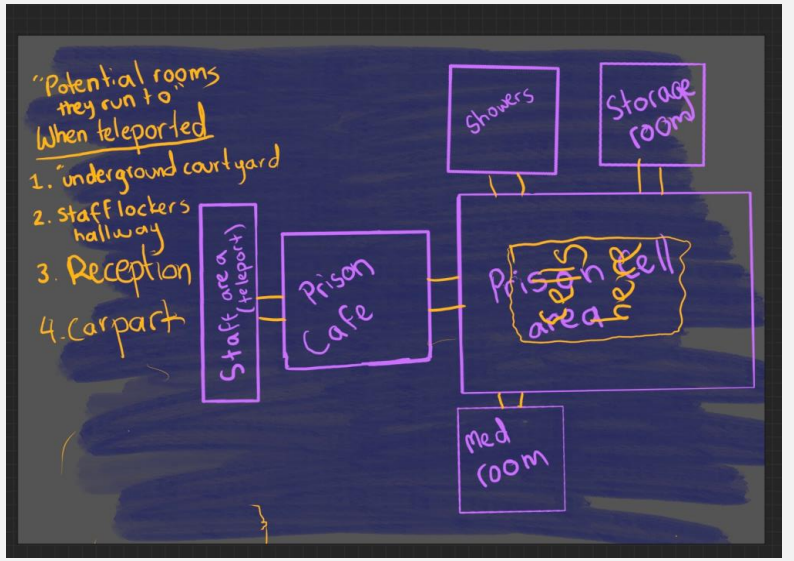



Figure 9.1 - Screenshot of escape room level design.

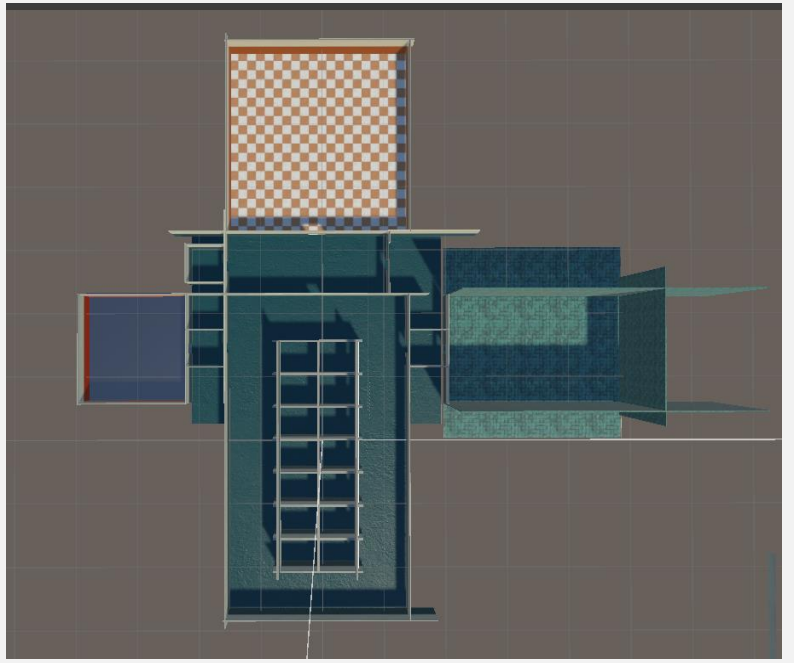


Figure 9.2 - Screenshot of escape room 3D model level design.

# Winter Break 5: Animating the doors

Next, I worked on making the cell doors animated. So that they can talk to the prisoners inside. All I did to do this was add a collider (Figure 10.1) with a trigger checked (Figure 10.2) at the doors after animating them an idle, an open and a close animation. Then I added new parameters, triggers called "DoorOpen" and "DoorClose" which was set in OnTriggerEnter and OnTriggerExit functions (Figure 10.3). Then I just had to update the old script I made for the door that's unlockable with a key only, to have different parameters to the doors that don't need keys. This parameter is called "KeyOpen." (Figure 10.4).

The reason why I made the cell doors be accessible to the player, instead of making the prisoner (Figure 10.5-10.6) must find an object for every cell door, is because it's inconvenient. The space is already small, so I want the player to feel like they're in control in the areas where the prisoner can go. It made more sense to only have the player look for items to help them escape the prison. It would make the game unenjoyable too, and feel like a chore, if they had to unlock every door to find the NPCS that will give them the information on escaping.

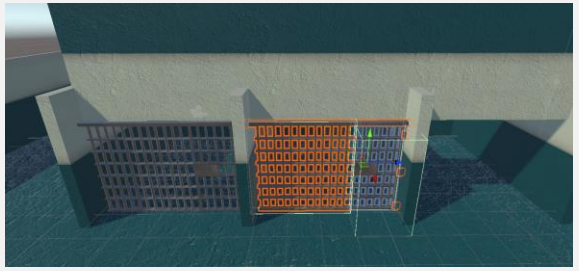


Figure 10.1 - Screenshot of door trigger.

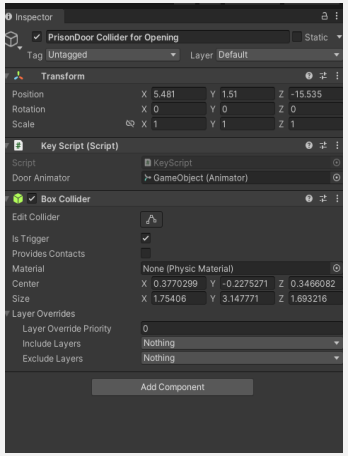



Figure 10.2 - Inspector set up screenshot for key.

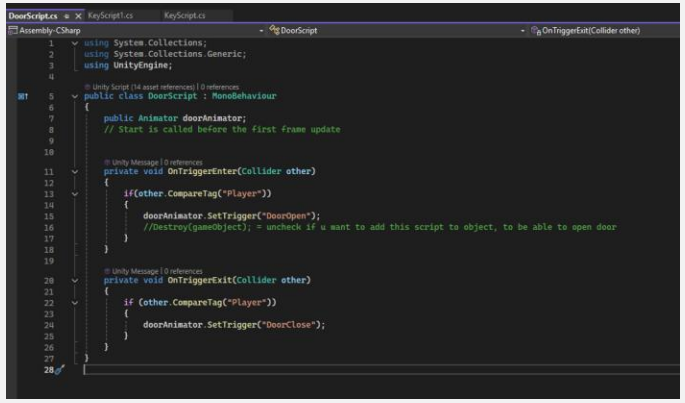



Figure 10.3 - Screenshot of Door Script.

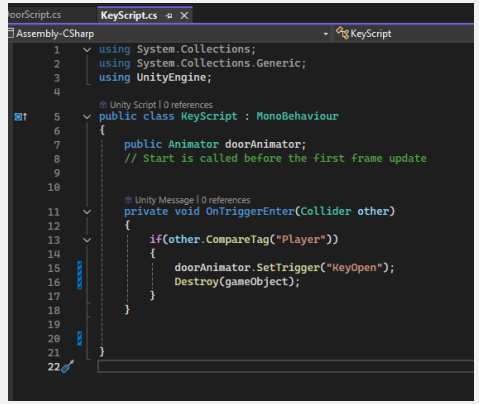



Figure 10.4 - Screenshot of updated Key Script.

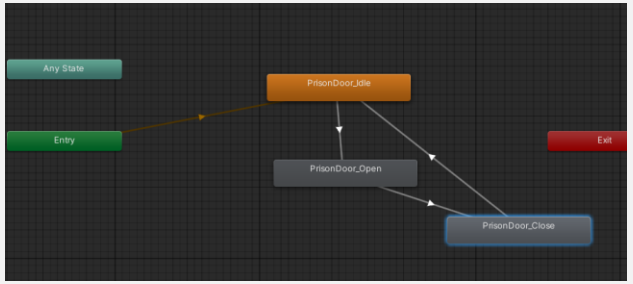



Figure 10.5 - Prison door Animator.

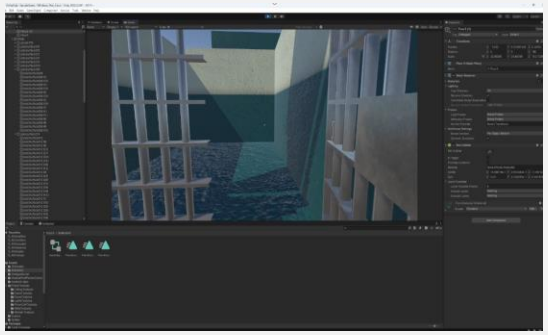


Figure 10.6 - Screenshot oif door opening gameplay.

# Winter Break 6: Chase scene set up 1

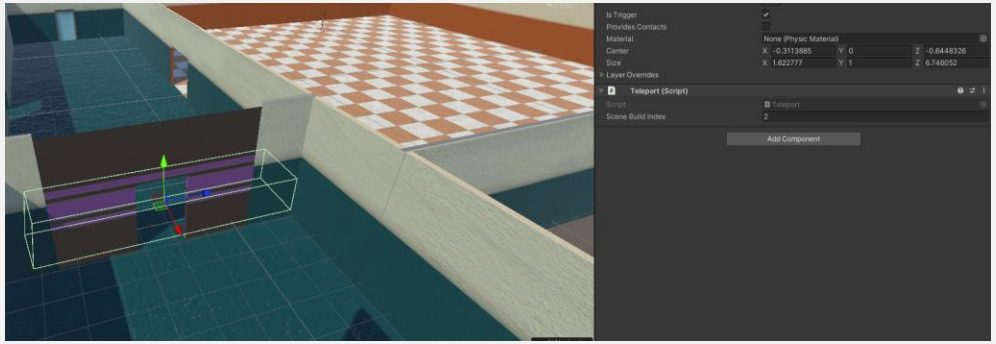


Figure 11.1 - Trigger set up for teleport to chase scene.

Mentioned in the new game plan, we have a chase scene that the player is teleported to after escaping to the staff area. This was to expand the game play for a bit if escaping the prison is way too easy. This chase scene features the reveal of who the obssessive ex is and is the final area that the player is forced to go through to escape. Who doesn't get stressed from the feeling of being chased? That's the type of feel we want to achieve here.

After setting up a collider with the trigger checked (Figure 11.1) I chose to save time with the teleport part, by recycling a script from  Year 2, that I made. "Teleport.cs" (Figure 11.2) It allows the player to be able to teleport to different scenes based off their order in the build index.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class Teleport : MonoBehaviour
{
    public int sceneBuildIndex;

    //Level move zoned enter if collider is an tagged as an ally
    //Move game to another scene


    private void OnTriggerEnter(Collider other)
    {
        print("Trigger Entered");

        if (other.CompareTag("Player"))
        {
            //player has entered so move level
            print("Switching Scene to" + sceneBuildIndex);
            SceneManager.LoadScene(sceneBuildIndex, LoadSceneMode.Single);
        }
    }
}
```

Figure 11.2 - Screenshot of Teleport Script.

The reason for using a game over screen (Figure 11.3) instead of adding checkpoints is because our map is not big enough to need check points so I decided that a simple retry button for the chase scene is the best decision so that the player doesn't have to re-escape the room to get to this part.


Figure 11.3 - Game over screen prototype.

I also recycled my script from last year "MainMenu.cs" (Figure 11.4), to handle all of the buttons on the game over panel, and this same script will be used for the win scene as well.

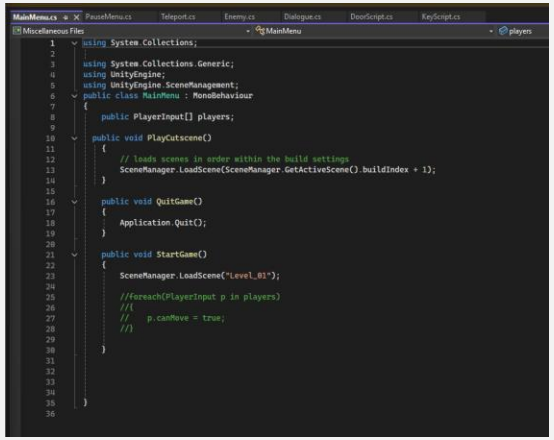


Figure 11.4 - Main Menu Script screenshot.

# Winter Break 6: Chase scene set up 2

The mechanic in the focus here is the ex-lover's chasing mechanic in the chase scene. This mechanic was a necessary environmental interaction to code because it's the foundations of polishing the gameplay and balancing the difficulty level of the chase. The chase scene is major scene that reflects the ex-lover's destructive/controlling personality within the environment and their drive for capturing Y/N.

I used a tutorial (Kozmobot Games) to help me understand Navmesh agents and surfaces for the chase scene.

This tutorial taught me how to make a game object, the enemy, follow the player using NavMesh (Figure 12.1) and one line of code in the Update function (Figure 12.2) which sets the destination of the enemy towards the player the whole time.

Then I had to make a new function called "KillPlayer" to handle code that would cover killing the player and popping up the game over panel I made (Figure 12.3). There was an issue with the cursor disappearing or getting stuck in the center of the screen.  Since the player's controller is in first person, the cursor gets destroyed. So,  all I had to do to make it usable was add in "Cursor.lockState = CursorLockMode.None;" and "Cursor.visible = true;" (Figure 12.3).

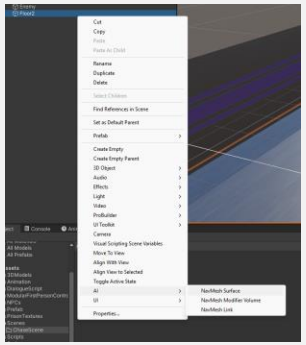


Figure 12.1 - Screenshot of NavMesh.

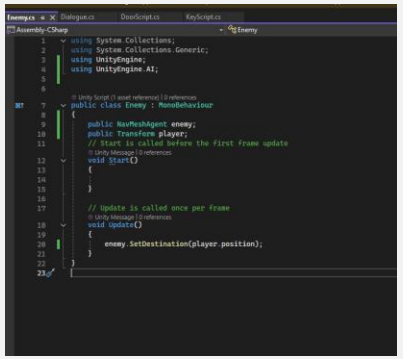


Figure 12.2 - Enemy Script screenshot.

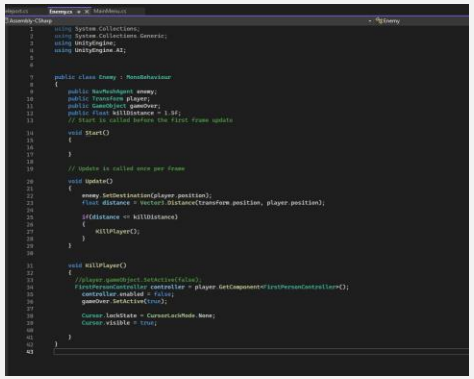


Figure 12.3 - Updated Enemy Script screenshot.

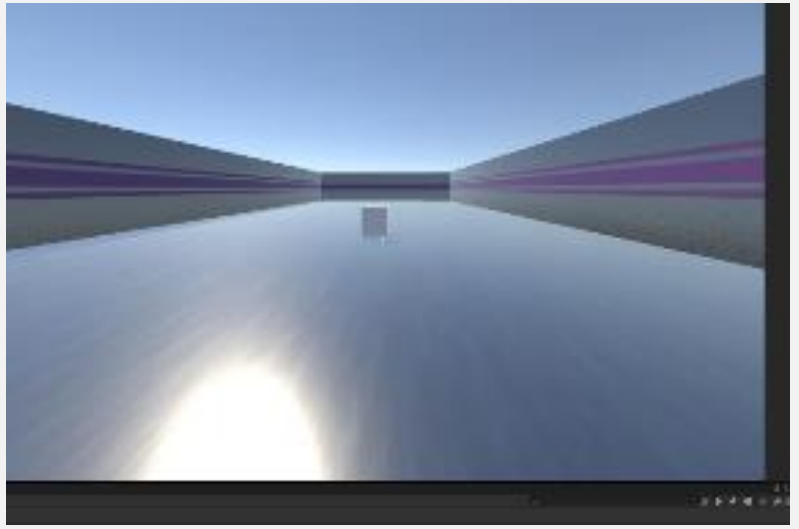


Figure 12.4 - Screenshot of enemy chasing player gameplay.

# Winter Break 6: Coding a keypad

For the last week of break, I focused on getting the keypad working, initially, our keypad was going to be colour coded but since our game took a new direction, I decided to go from the basics because it's my first time making a working keypad to be able to unlock a door. I used a simple keypad reference image to follow (Figure 13.1).

The tutorial I followed was *Unity – Open Door, with Safe Code* (Clip Collection Vault, 2022).

This tutorial was easy to follow as it covered a weaved combination of functions and variables that I am already aware of (Figure 13.2). It's possible to change the numbers to letters that the player will need to scramble to form a word or be colour coded. This is changeable through Unity's OnClick() functions and how the code set it up so that we can change the field from numbers to letters.

The reason why we need a safe code pass (Figure 13.3-13.4) to unlock the last door is to control the player's gameplay in the game and to prevent them from escaping straight away. They're being controlled to stay within the escape room first because they cannot get information on what the code is unless they do tasks for the prisoners and explore for items that they may need.

The player themself has control over their decisions; by being able to decide and start with which ever prisoner they want or they could just try to guess the code too which they can if they choose to. If there's a keypad on the last door that requires a multi-numbered or multi-worded code, the player is controlled to either guess or explore within the space. We want the player to think that the last door is where the game finishes, this is achievable through the keypad.


Figure 13.1 - Keypad Script screenshot.

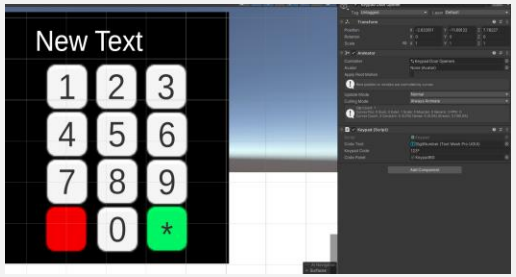



Figure 13.3 - Screenshot of Keypad script set up.

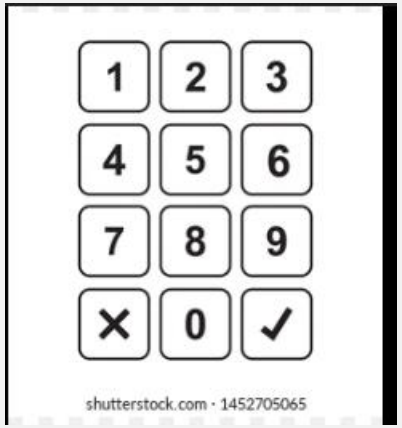



Figure 13.2 - Keypad reference.


Figure 13.4 - Screenshot of keypad gameplay.

# [Semester 2] Week 1: Adding lighting, fixing floors

For the first week back, I focused on adding lighting (Figure 14.1-14.2). Taking the same reference image, I used when 3D modelling the wires (Figure 1, p.3), I added in the wired ceiling (Figure 14.3) within the escape room scene too. Before I could do this, I fixed the floor flickering problem, where the floor meshes would intersect with each other. The same issue was happening with the ceiling planes. After research ("Floor flickering", 2016), it's recommended that your floor meshes, don't intersect to prevent z-fighting. An issue that occurs when two planes are in same space causing the rendering to flicker between the two planes.

I chose to fix the flooring (Figure 14.4) when I noticed that lighting on the floor planes were uneven due to the z-fighting floors. I tried to manually line them up at first, but the planes just wouldn't line up properly. So, after further investigation on the issue, I tried to find a way to snap the planes together. I found a tool (Brain Fail Productions, 2021) in the Unity asset store and decided to try it out. This tool allowed meshes to snap together by using the keybind "S" to snap their vertices together. It worked at first but the scripts were outdated and they kept causing "Object at index 0 is null"(Figure 14.5) errors whenever I would play the game. Knowing that easy snap was made to snap vertexes together, I researched for a tool that would allow me to snap vertexes in Unity and apparently there was according to another tutorial I found on YouTube (Code Master, 2022). I learned that you must hold "V" which will snap to wherever your cursor is. Clicking and dragging to the vertex of the mesh you want to snap to will allow your selected mesh to snap to there.

If I didn't fix this issue, it would have been extremely distracting away from the gameplay if the flooring were left as is. Little issues like this would ruin the atmosphere of any game not just ours.

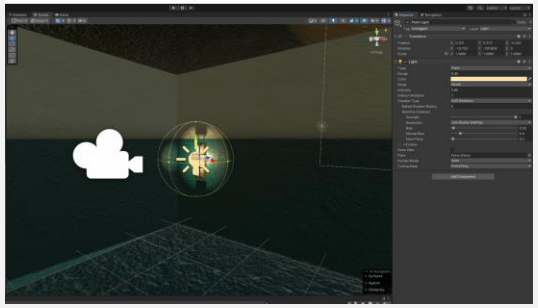


Figure 14.1 - Light test.

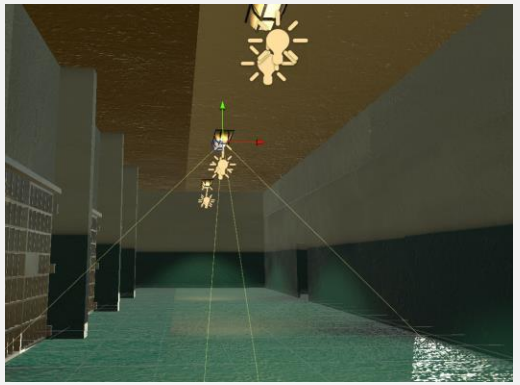


Figure 14.2 - Prison light set up.

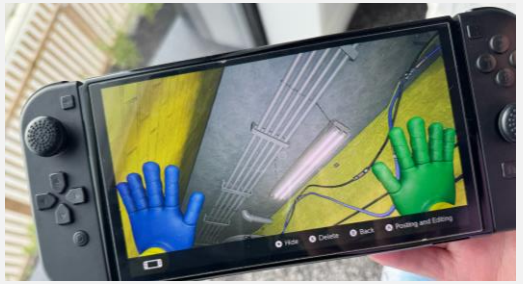


Figure 1, p. 3

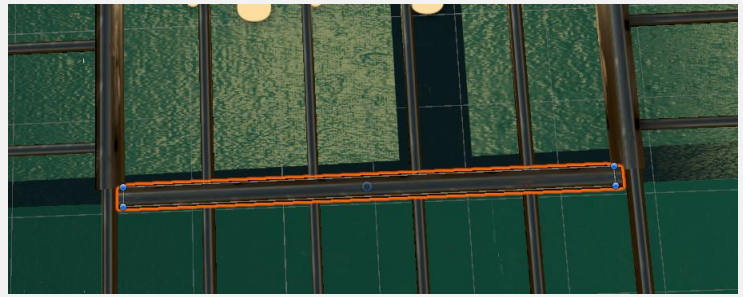


Figure 14.3 - Adding wires.

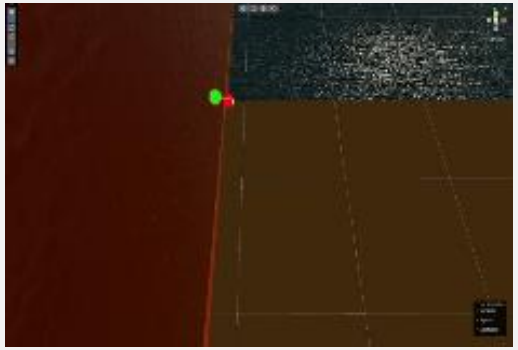


Figure 14.4 - Using easy snap to snap vertices.

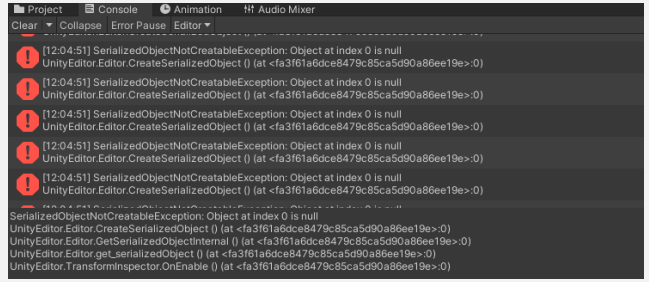



Figure 14.5 - The error that came with easy snap.

# [Semester 2] Week 1: Retexturing pipes, adding win screen

Moving forward, I added a win game panel (Figure 15.1), set up the same way as the game over panel, the difference is that it's not triggered in the chase scene, the exit teleports you to the win screen. The reason we did this is to prevent errors that come with changing the enemy script, which I did not want to do since coding is not my focus.

We also have more space to create a potential cutscene by making a new scene for the win screen. The only changes I had to make was to add the cursor.lock lines of code (Figure 15.2) that I did for the keypad last week so that the player has control over the cursor when they get teleported to the next scene.

I also had time to retexture the pipes from its original red with the slime on them (Figure 15.3) to a turquoise version of it (Figure 15.4). I wanted the colour of the pipes to look cohesive with the prison walls. From research, green pipes (Figure 15.5) can be known for wastage/sewage pipes (McGarry & Madsen Inspection, 2018)and blue pipes are known for water pipes (UtilityFinder, 2024), since the prison is supposed to have a grungy look and be in gross conditions, I thought picking a mixture between green and blue would be good for the pipes, to kind of imply that the water and waste pipe are the same. This small detail, enhances the idea of the authorities controlling the inmates through mistreatment in the underground prison.

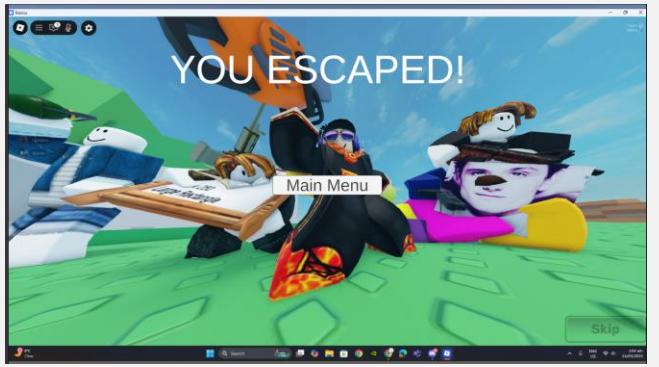
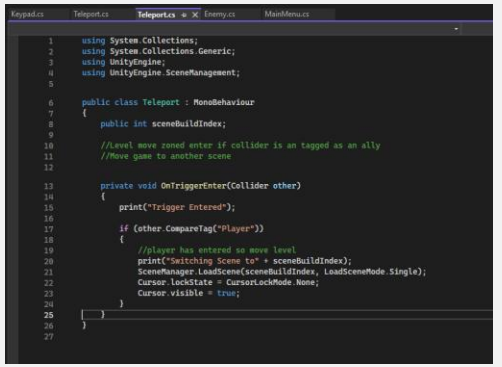



Figure 15.1 - Win screen prototype.



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class Teleport : MonoBehaviour
{
    public int sceneBuildIndex;

    //Level move zoned enter if collider is an tagged as an ally
    //Move game to another scene

    private void OnTriggerEnter(Collider other)
    {
        print("Trigger Entered");

        if (other.CompareTag("Player"))
        {
            //player has entered so move level
            print("Switching Scene to" + sceneBuildIndex);
            SceneManager.LoadScene(sceneBuildIndex, LoadSceneMode.Single);
            Cursor.lockState = CursorLockMode.None;
            Cursor.visible = true;
        }
    }
}
```

Figure 15.2 - Teleport Script screenshot.

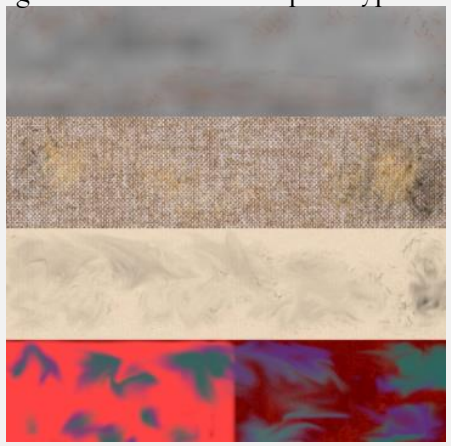


Figure 15.3 - Old prison trim sheet.

Figure 15.4 - New prison trim sheet.

Figure 15.5 - Pipes reference.

# [Semester 2] Week 1: Creating a Sound Manager

I worked on importing an audio manager into our game. It's the same as the one I used for the multiplayer game we made last year, *Hallowed Three*. This audio manager (Figure 16.1) will oversee managing the sound effects we plan to add into our game.

An SFX prefab (Figure 16.2) was made to spawn and be destroyed after playing a sound effect.

Sound is so important, as mentioned in my annotation of Sinclair's *Principles of Game Audio and Sound Design: Sound Design and Audio Implementation for Interactive and Immersive Media,* I want the player to be able to feel the music as they explore. Sound can either make or completely destroy the aesthetic within a game. So, it's crucial that we pick sound effects and background music that match and enhance elements within our narrative.

Issues that I ran into while implementing in the audio manager, was that the audio wouldn't work when I attached it to the parent game object of the player, I had to put it on the game object which had the camera attached on it (Figure 16.3), the current key bind to test the audio is "1".

<u>Notes for the Sound Manager:</u>

- works when player audio is placed on game object with camera

- works when placed on an object near the player too

- Audio Manager is on its own game object

- SFX Prefab is called to Audio Manager and Destroyed after full audio clip has played

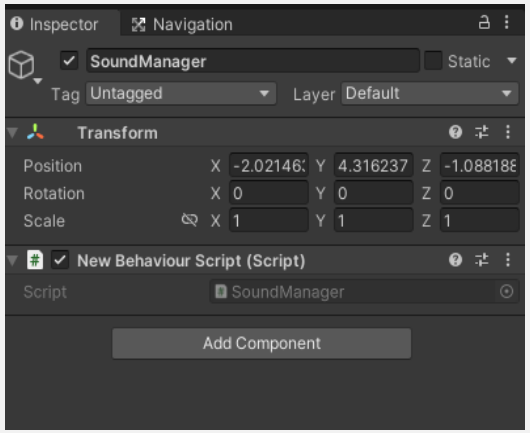



Figure 16.1 - Sound manager gameobject.

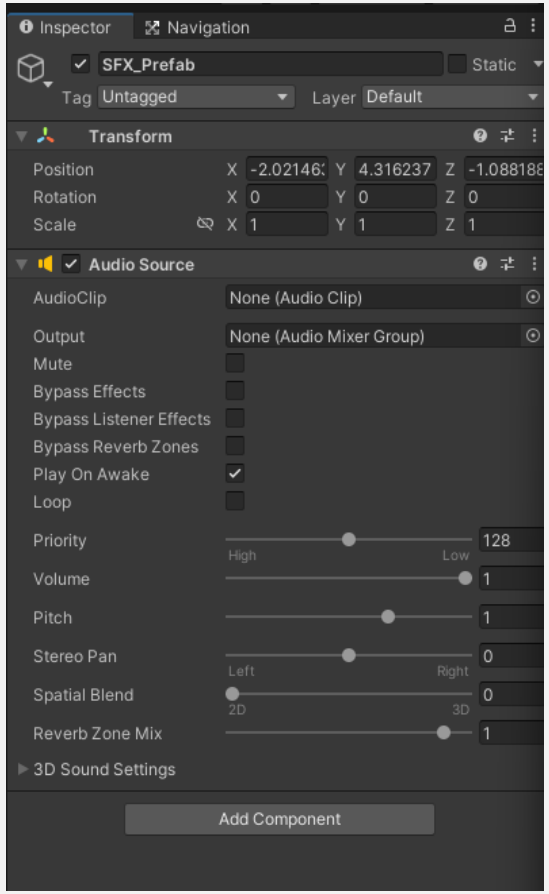



Figure 16.2 - Screenshot of SFX prefab on Inspector.

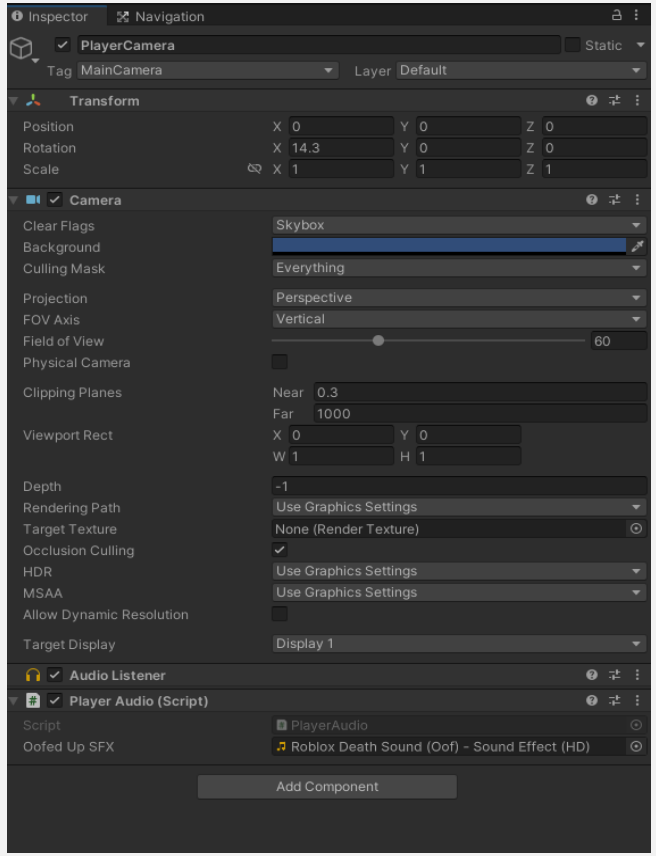



Figure 16.3 - Player Camera Inspector set up.

# [Semester 2] Week 2:Chase scene Obstacle Course Level Design (Obby)

Before setting up, I created iterations (Figure 17.4) to map out the pathways in the chase scene. Purple is the pathway; orange is the locked doors and yellow are the obstacles.

My chosen iteration is the sketch (Figure 17.3) that didn't have extra pathways leading to the same exit. The reason for this is because I want the player to feel powerless and forced to find the one path to make the exit.

Taking inspiration from *DOORS* (LSPLASH, 2021) on Roblox (Figure 17.2) level design of the chase scene, I chose to add hallways that split in two and doors that can lead to dead ends to give the players to allow for some decision-making. This was intended to give the player's a sense of decision-making, which increases player agency and provides a feeling of control. In *DOORS*, obstacles are created by cluttered 3D models that block the player's path. Which forces the player to move forward or through the obstacles.

I also took inspiration from *Poppy Playtime: Chapter 3 (*Mob Entertainment, 2023), features a shorter duration (Figure 17.1) of their chase scene. It's designed so the player cannot escape their death, so they can wake up from the nightmare sequence. From this I took inspiration of the tight one-way hallway they had, to take control from the player completely, leaving them powerless.

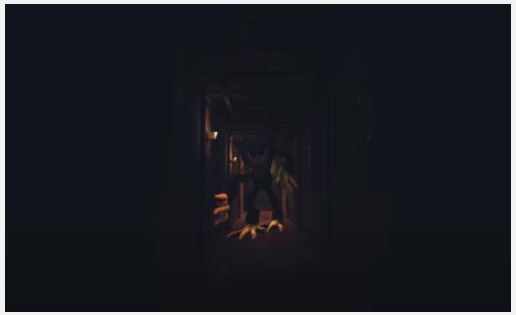


Figure 17.1 - Screenshot of P*oppy Playtime - Chapter 3* (Mob Entertainment, 2023) chase scene.

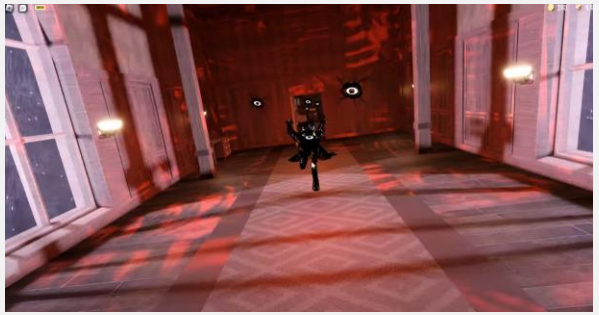


Figure 17.2 - Screenshot of *DOORS* (LSPLASH, 2021) chase scene.

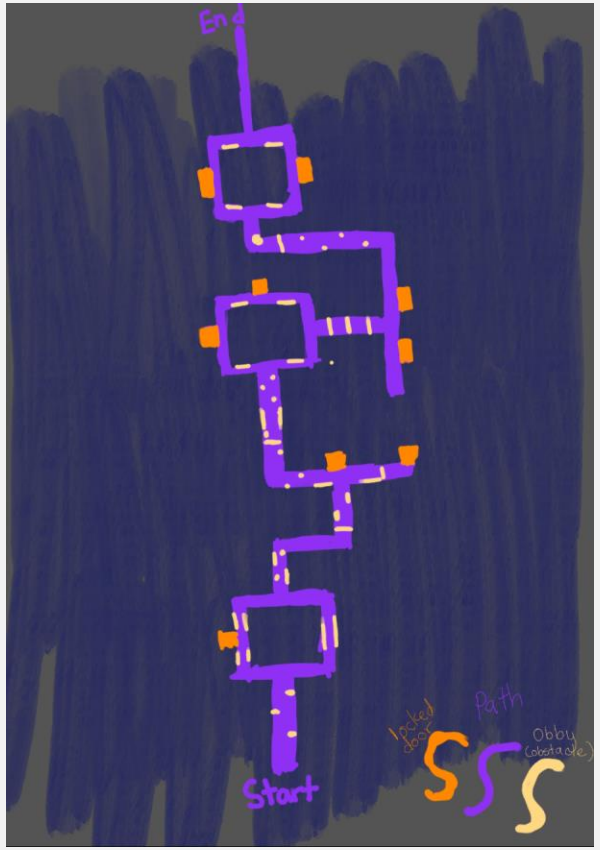



Figure 17.3 - Chosen final chase scene level design.


Figure 17.4 - Chase scene level design variations.

# [Semester 2] Week 2: Chase scene Obstacle Course Set up (Obby)

Using my chosen level design sketch (Figure 17.3, p.19) as inspiration to map out the pathway towards the exit. Once I finished building the map, I re-baked the NavMesh but ran into issues where the NavMesh wouldn't bake evenly (Figure 18.1), so I went back and made sure that the floor meshes were on the same depth then re-baked again with the walls elevated (Figure 18.2), this seemed to fix the issue. Elevating the walls before baking caused the NavMesh agent to not detect the walls as an obstacle, making it spiral whenever it hit a wall, so I re-baked the floor mesh again (Figure 18.3) with the walls and doors on it.

I chose to make the exit door and the openable doors red (Figure 18.4) to contrast with the colour of the closed doors (Figure 18.3) this way I can give the player a visual hint of which door to interact with.

Originally, I wanted the ceilings to be curved (Figure 18.5)like how underground tunnels are shown (Figure 18.9) but because of the square design of the corridors, it made more sense to keep the ceiling design flat (Figure 18.6). To remove the noticeable seam between the walls, I removed the height map off "WallMaterial 1" (Figure 18.6). The obstacles made were the lockers I 3D modelled from over the winter break (Figure 18.7) and the paper clutter from the old warden's office (Figure 18.8) that I 3D-modelled last semester. The reason for the clutter and lockers being ripped from the walls is to reflect the enemy's personality in the environment because their purpose is to prevent the player from leaving the prison so that they can be together.

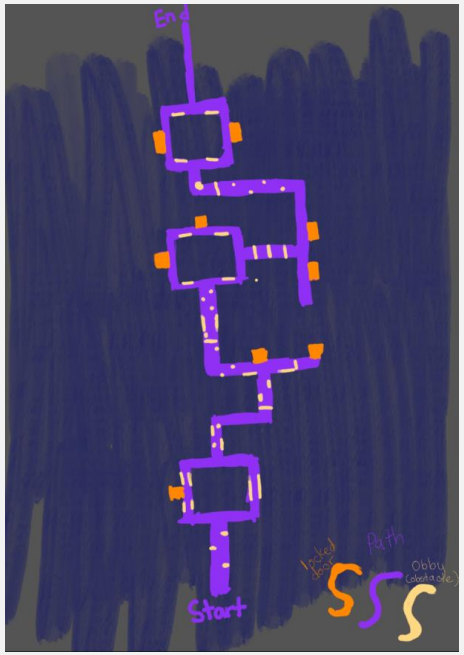



Figure 17.3, p.19

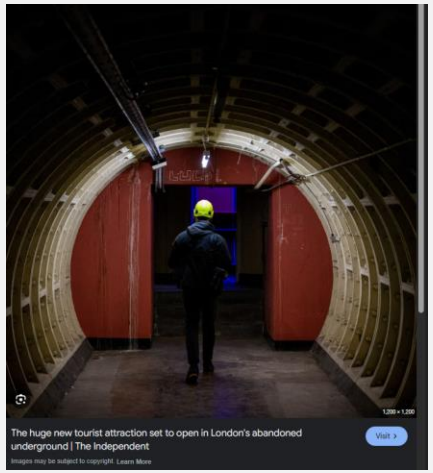



Figure 18.9 - curved ceiling example.

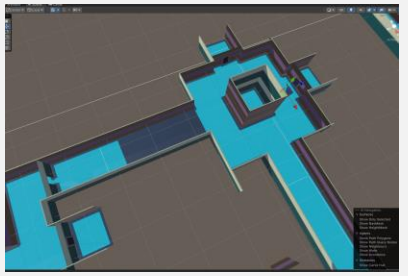


Figure 18.1 - Screenshot of baking error.

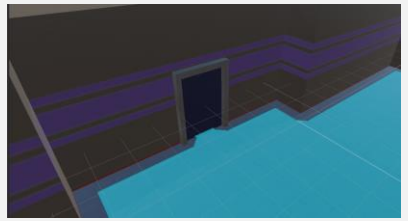


Figure 18.3 - Closed door example.

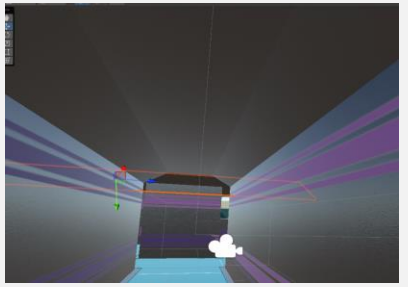


Figure 18.5 - Curved ceiling example.

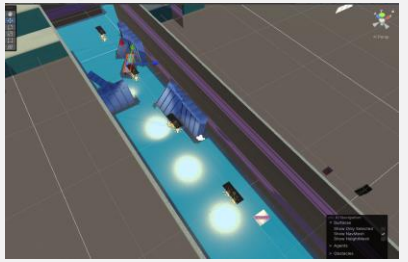


Figure 18.7 - Obstacles example.

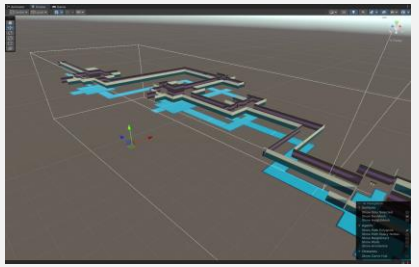


Figure 18.2 - Elevate walls screenshot.

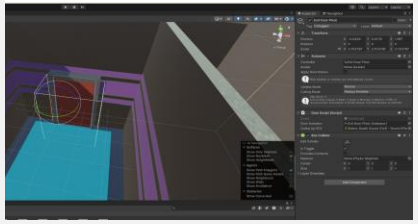


Figure 18.4 - Correct door example.

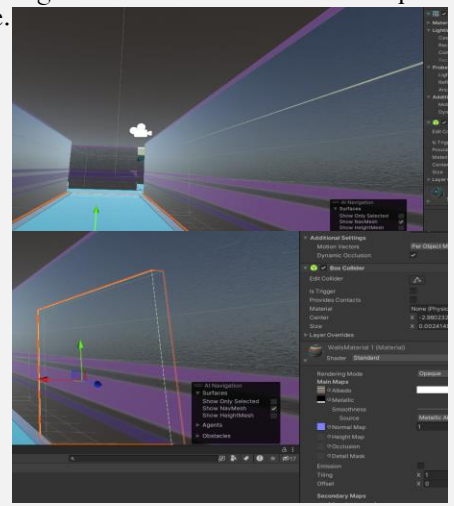


Figure 18.6 - Higher flat ceiling design.

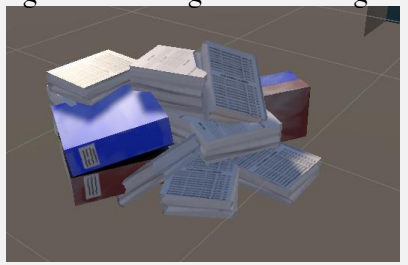


Figure 18.8 - Paper clutter.

# [Semester 2] Week 2: Chase scene Obstacle Course Navmesh

There was an issue with the NavMesh agent not being able to jump above obstacles. According to the Unity NavMesh tutorial by CodeMonkey (2021), changing the jump distance, height, step height and max slope can help the NavMesh agent to walk on top of obstacles blocking the way. So, I changed the default settings of the NavMesh agent to increased settings of their height, step height, max slope and jump distance (Figure 19.1), this way they can detect the obstacles and walk over them.

It was still not enough to just adjust the NavMesh agent's settings before baking the surface to help them jump over the obstacles (Figure 19.2) because sometimes the NavMesh agent would just slide down away from the player and glitch, so after doing research in the Unity Discussion board ("Making Agents Jump and Climb", 2022) I learned about "Off-mesh links", though (you can just auto-generate some through the NavMesh surface, it's recommended to do it manually. I just increased the width of the off-mesh link (Figure 19.3) to prevent the NavMesh agent from missing the point.

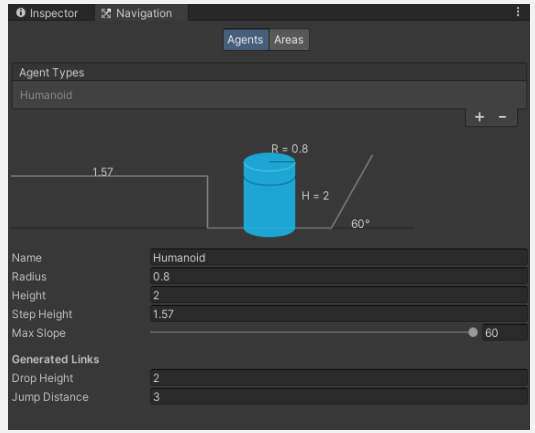



Figure 19.1 - NavMesh agent navigation settings.

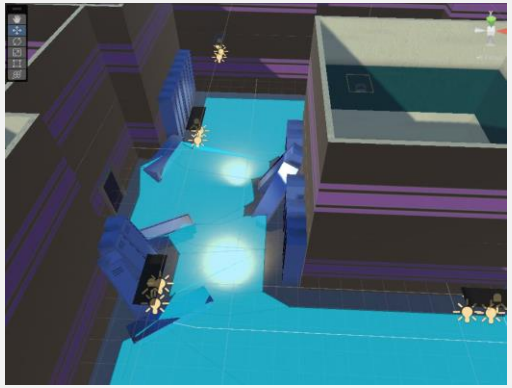


Figure 19.2 - Updated obstacle baked surface.

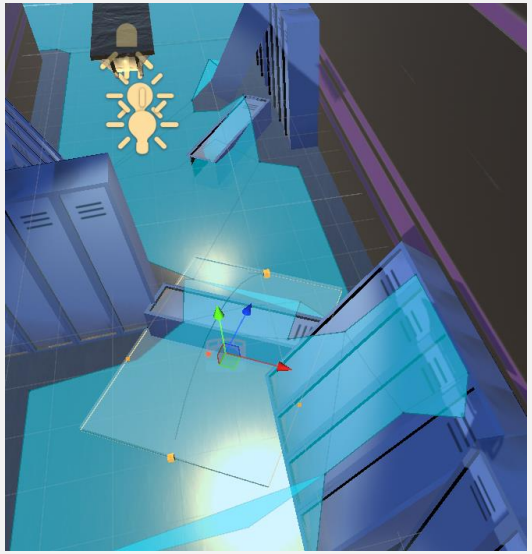


Figure 19.3 - Adding NavMesh links in.

# [Semester 2] Week 2: Showers Obstacle Course Level Design (Obby)

Here's the level design iterations I made for the obstacle course in the showers (Figure 20.3). I decided on my final design (Figure 20.4) because it has an equal balance in difficulty to getting the soap whether they go left or right. This obstacle course initiates player agency and less narrative because the player is pushed to get the bar of soap for the story to progress.

In *BARRY'S PRISON RUN! (OBBY).* (PlatinumFalls, 2022), there's a section in their obstacle course (Figure 20.1) where the player is encouraged to jump on the pipes to progress to the next level and to avoid the cop below.  Taking inspiration, I plan to reuse the pipes I made earlier in the year as a pathway to the bar of soap.

*Poppy Playtime: Chapter 2* (Mob Entertainment, 2022) has levels where the player comes across a hole (Figure 20.2) that they cannot cross without swinging across. I want to take inspiration from this and add a hole that you must avoid by jumping on pipes and other clutter in the showers.

Both inspirations encourage a forced path on the player and a controlled consequence when they fall into the hole. Pink represents the bar of soap, blue is the obstacle course and red is the death zone.


Figure 20.1 - Screenshot from *BARRY'S PRISON RUN! (OBBY)*. (PlatinumFalls, 2022).

Figure 20.2 - Screenshot from *Poppy Playtime – Chapter 2* (Mob Entertainment, 2022).

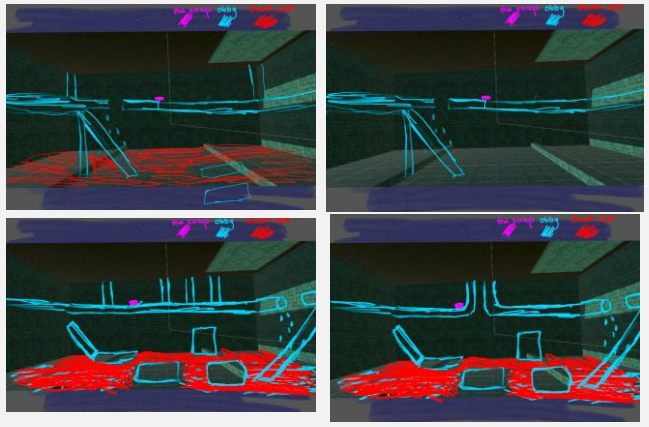


Figure 20.3 - Showers level design variations.

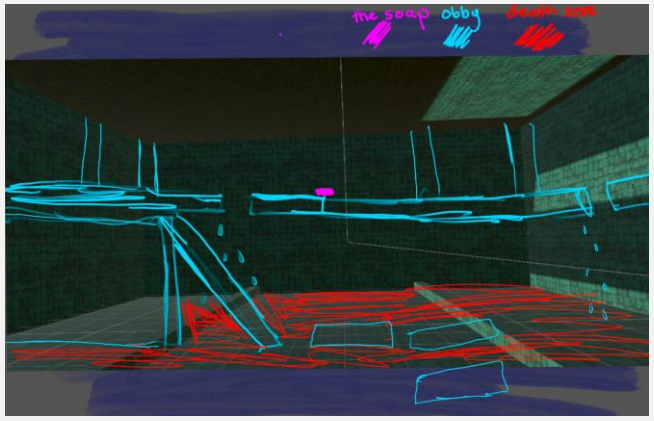



Figure 20.4 - Showers chosen final level design sketch.

# [Semester 2] Week 2: Showers Obstacle Course Set up (Obby)

Using my chosen level design (Figure 20.4, p.22), I went on Unity to set up the environment for the shower obstacle course. To utilise my assets, I reused the pipes that I had already pre-made last semester and move the soap to the area where we want it to spawn after interacting with the prisoner in the showers (Figure 21.1). This keeps the design cohesive since it's seen in other parts of the prison.

The idea for this obstacle, reflects the ex-lover's personality. They're destructive and controlling. Destruction is reflected through the shower area, and they're controlling because there's an obvious path to get to the soap. Coaxing the player to grab it. It's also the only bar of soap that's shared between the inmates, reflecting how unsanitary the place is.

Since there's a hole at the bottom of the obstacle that the player can't get out of when they fall into it. I coded a simple respawn script (Figure 21.2) featuring an OnTriggerEnter function that helps to respawn the player at the start of the obstacle course so that they can try again if they fall.

According to ChatGPT, I had the right idea on how to code it, what's missing was turning off the player controller and turning it back on, because of the player controller that we were using. "Physics.SyncTransform();" helped to synchronise the physics again when re-enabled. (Figure 21.3)

Lastly, to hide the bottom of the hole, I followed this tutorial on volumetric fog in Unity (Etredal, 2021).

Which allowed me to create volumetric fog in unity. I did this to make the hole look endless and a long drop. This can help the player feel more inclined to not fall for fear of losing progress. The main changes I made was to set the start speed to 0.1, so that the fog isn't unrealistically fast (Figure 21.4) and to change the colour to black under color and lifetime. (Figure 21.5).

The finished scene looks like there's an ominous fall and a clear pathway for the player to follow to get to the soap. (Figure 21.6).

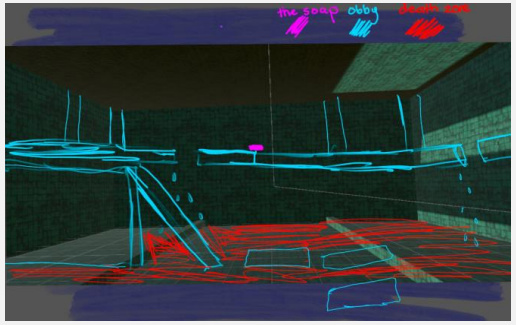



Figure 20.4, p.22)

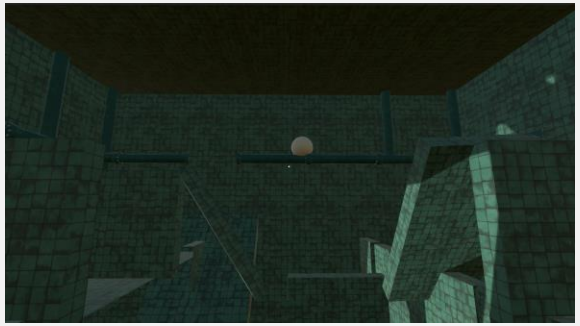


Figure 21.1 - Final Showers level design.

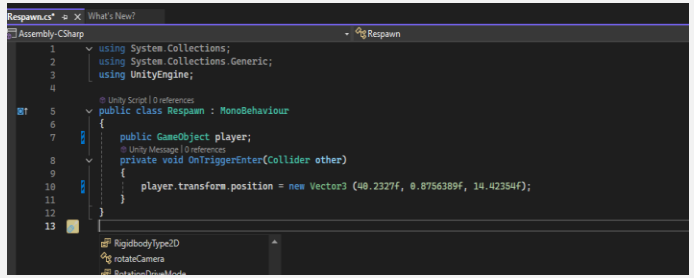



Figure 21.2 - Screenshot of Respawn Script.

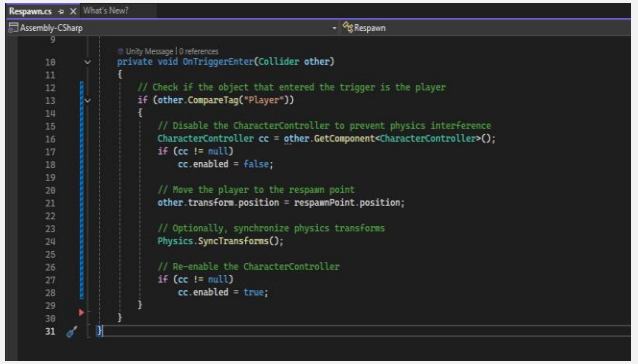



Figure 21.3 - Updated Respawn Script.

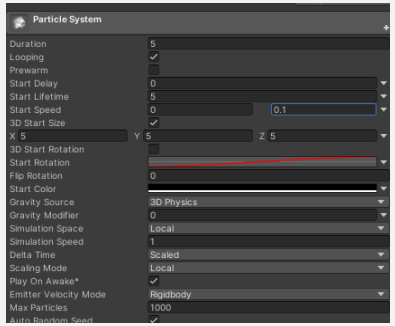



Figure 21.4 - Start speed change.

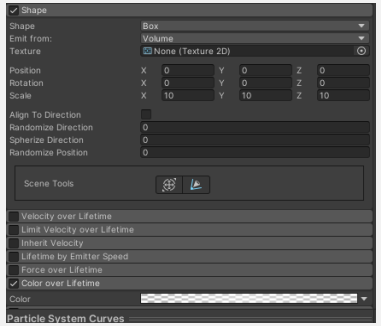



Figure 21.5 - Colour and lifetime change.

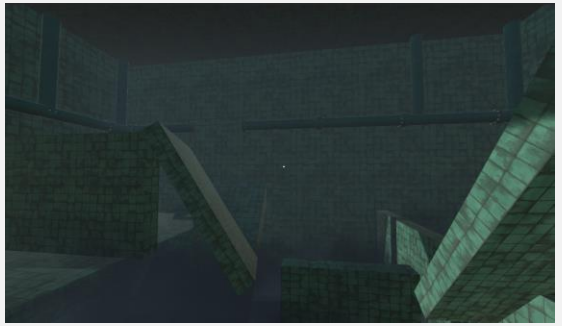


Figure 21.6 - Added fog to the Showers.

# [Semester 2] Week 3: Texturing Infirmary

Taking inspiration from old-fashioned retro aesthetic from our original game concept, I used reference images (Figure 22.3) to create a rustic, retro infirmary for our prison.

I chose this approach because I wanted to highlight that the place aged badly overtime. Reflecting the HR's carelessness towards the environment that the inmates reside in.

The stool and the medical ward dividers (Figure 22.1) I chose materials that match their reference images like "Leather Grain" for the leather seat of the stool, "Fabric linen" for the fabric used on the dividers and I chose "Metal brushed" as the base material of the supports.

After choosing the base materials, to make everything look rustic I created a new layer and used an alpha brush (Figure 22.6) with an orange base colour to create rust spots.

I took an 80's looking medicine cabinet as inspiration (Figure 22.3) for the table, sink and cabinets to fit our retro aesthetic. The final three iterations of the cabinets (Figure 22.4) were detailed by using height on the brushes (Figure 22,5) to create the illusion of doors and drawers on the bottom.

I chose this approach because the player doesn't need to open drawers to get a quest item which provides smoother gameplay because the only way to pick up an item is if you walk into it.

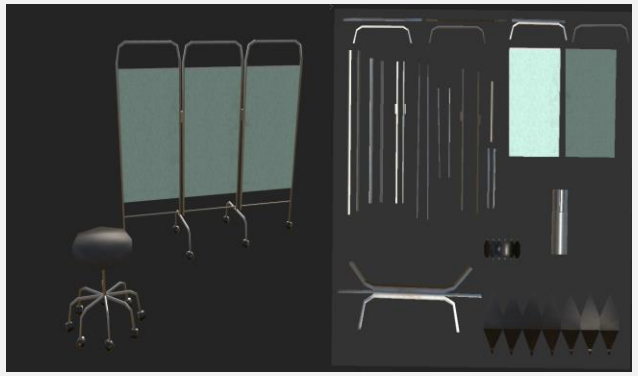


Figure 22.1 - Textured medical dividers and stool.

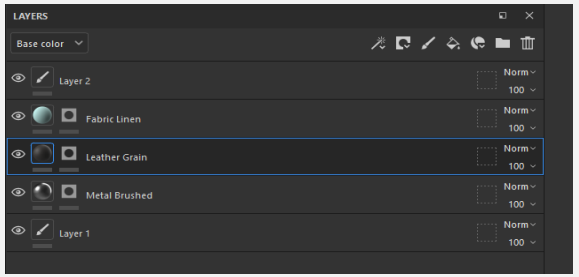



Figure 22.2 - Leather Grain Substance Painter material.

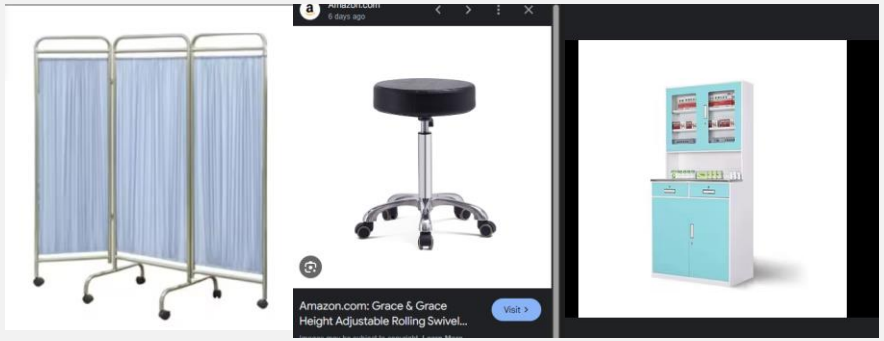



Figure 22.3 - Reference Images for Infirmary assets.

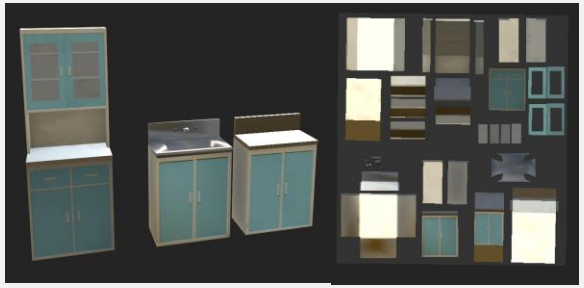


Figure 22.4 - Textured medical cabinets.

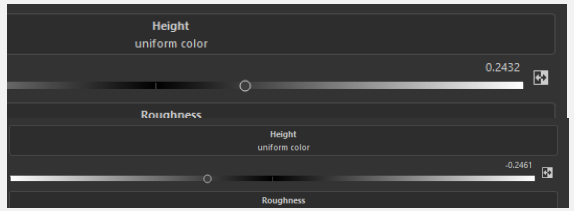



Figure 22.5 - Updated height.

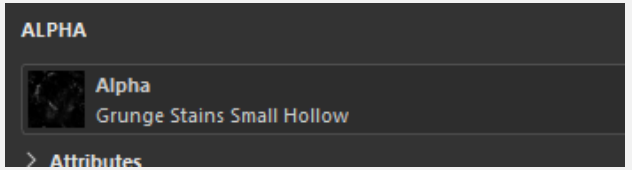



Figure 22.6 - Grunge Stains Small Hollow Alpha brush.

# [Semester 2] Week 3: Texturing Infirmary 2

Using the old medicine 3D models I made for the original game concept, the "Jelly" was the slime that the Snail warden uses to control the patients (Figure 23.2). Instead of having the jelly label on the medicines, I opted for just making the medicine look like a generic orange (Figure 23.1), still using the same reference image that I used for the jelly medicine.

Reason is that the new antagonist's personality isn't reflected through tampered medication anymore. It's reflected through the obstacles in the environment.

The rustic stains are added onto the syringe which is noticeable in the UV. The same alpha brush (Figure 22.6, p.24) was used for the rust on the stools and medical ward dividers for consistency. Keeping the old-fashioned/retro look I picked a reference image of one of those metal syringes (Figure 31, p.20). My decision of rustic metal makes it look unsanitary, making it fit the old, abandoned underground prison.

I wanted to pick a simple design (Figure 29, p.6) for the first aid kit. I decided that the bright red one fits the retro vibe the best. I hand drew the white cross (Figure 23.1) to make the first aid kit look low budget to reflect the mistreatment and carelessness of the authorities who were controlling the prison before the ex-lover's reign.

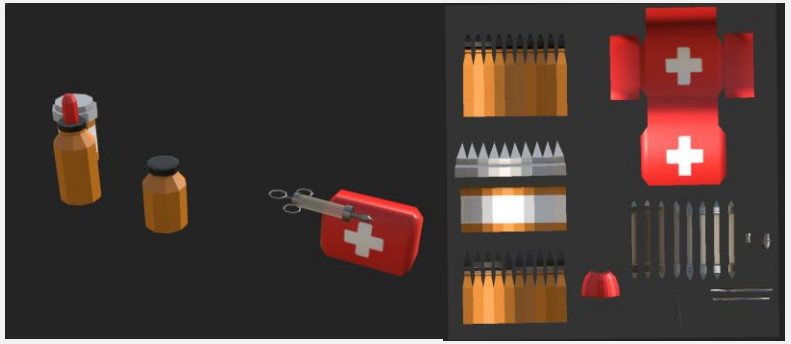


Figure 23.1 - Textured first aid kit, syringe and medicines.

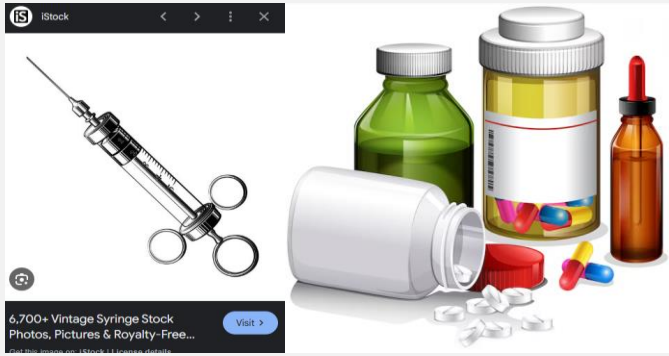



Figure 31, p.20


Figure 23.2 - Screenshot of old jelly testing notes concept.

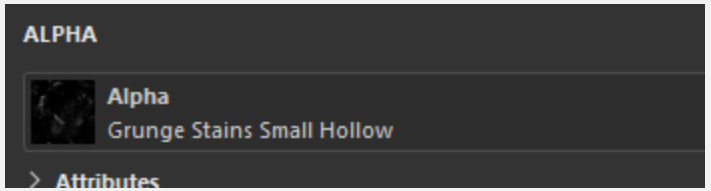



Figure 45, p. 21


Figure 29, p. 6

# [Semester 2] Week 3: Texturing Showers

For the prisoner showers, only the essentials were left to texture like soap, a drain and a shower head, I took an abandoned (Figure 24, p.19) prison shower room as the main reference image to follow off. The finished texture uses the same material "Metal Brushed" (Figure 24.7) to make the environment design cohesive with the other rooms like the infirmary. The tiling was just adjusted to make the areas with the metal material look slippery, which fits cause there's water in the Showers.

For the shower head (Figure 24.2) and the shower drain (Figure 24.3) I used a brush with height adjusted below zero, to make create the illusion that there's indents for where the water travels through. I chose the shower drain (Figure 24, p.19) with the diagonal slashes on it because it looked like those retro/vintage shower drains the best. I then used the large grunge stains alpha brush (Figure 24.7) to add rust and mold to the areas to make the area look unusable and unsanitary.

I used a gross reference image of soap (Figure 24, p.19) for reference, I tried to recreate the look of fat and oil for our bar of soap by using this material called "Paint Roll" (Figure 24.4). I just turned off the colour it was using by default and I used a similar orange to the rust (Figure 24.5) to recreate the soap. This bar of soap is going to be used as the quest item that the player will need to fetch by completing the shower obstacle course (Figure 24.8) that I made previously.

Figure 24.1 - Textured shower essentials.

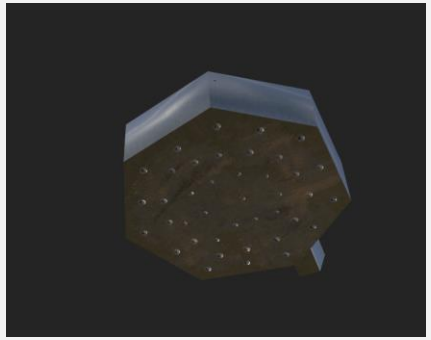


Figure 24.2 - Textured shower head.

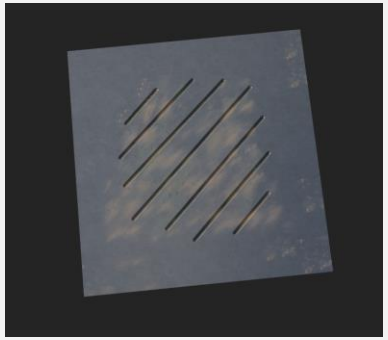


Figure 24.3 - Textured shower drain.

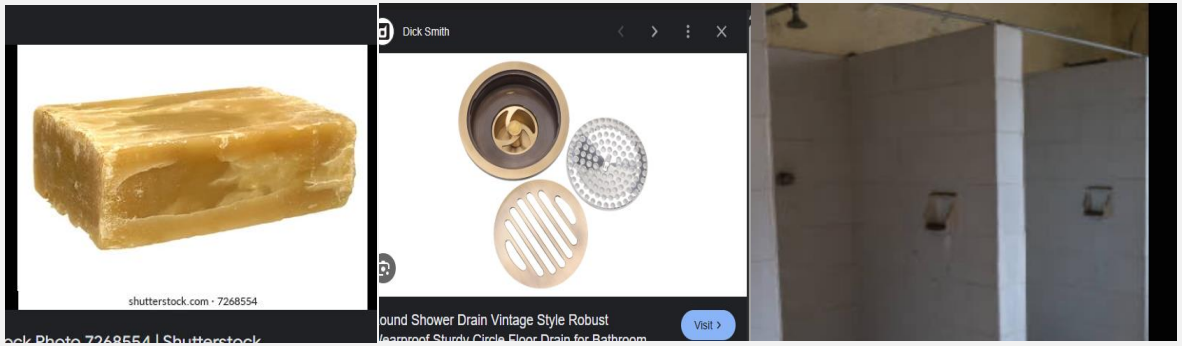



Figure 24, p.19

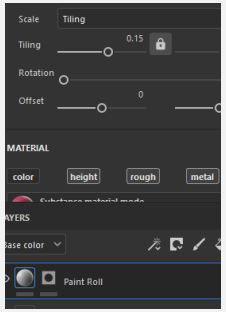



Figure 24.4 - Paint roll Substance Painter material.

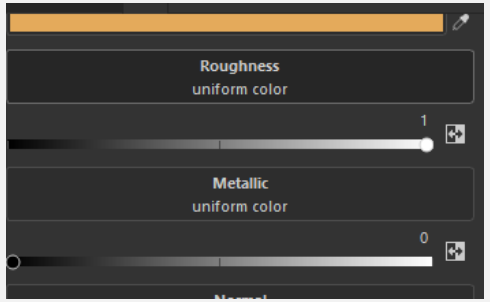



Figure 24.5 - Orange colour for rust.

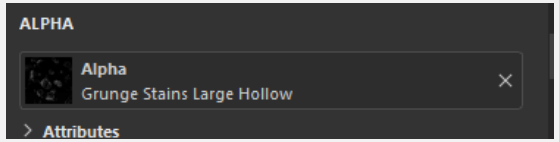



Figure 24.6 - Grunge Stains Large Hollow Alpha brush.

Figure 24.7 - Metal brushed Substance Painter material.

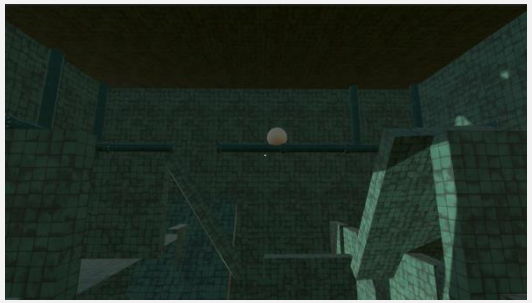


Figure 24.8 - Example of soap prototype in the Showers.

# [Semester 2] Week 3: Adding in Cafeteria 3D models into Unity

I still used the quick grey box I made last semester (Figure 25.1) to map up where I wanted to put the 3D models for the cafeteria into the game (Figure 25.2).

The only difference I made from the grey box, was adding more space between the tables so that the player can squeeze in between the tables.

I also tried to make a second material for the glass (Figure 25.3), setting it to transparent so that it simulates glass. I made it separate from the main “Food_Heater” material (Figure 25.4) which is set to opaque, so that the whole model doesn’t become see through.

Although I did this, I think that I still should have made the glass more transparent through substance painter, cause it’s still quite opaque, even after adding the transparency in it, I think that I’ll retexture the food heater so that the glass looks more transparent later when we do some polishing towards the end of the semester.

The reason for making the glass transparent is so that the player can see the food through the glass. It makes the glass look cohesive with other areas where we have glass, like the medicine cabinet made for the infirmary. It’s also good to make it transparent so that we can see the rust on the glass edges a bit more. Adding to the rustic abandoned retro look of the environment.

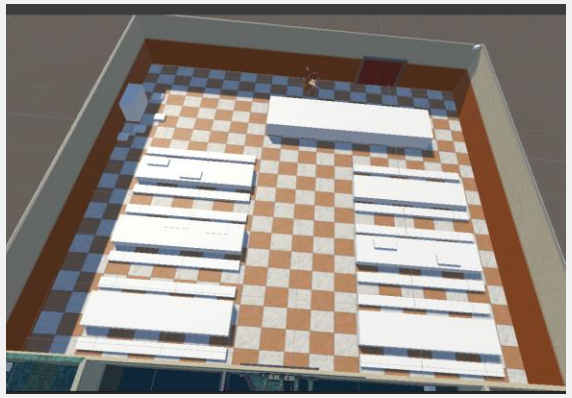


Figure 25.1 - Greybox of cafeteria.

Figure 25.2 - Updated level design of cafeteria.

Figure 25.3 - Duplicating food heater.

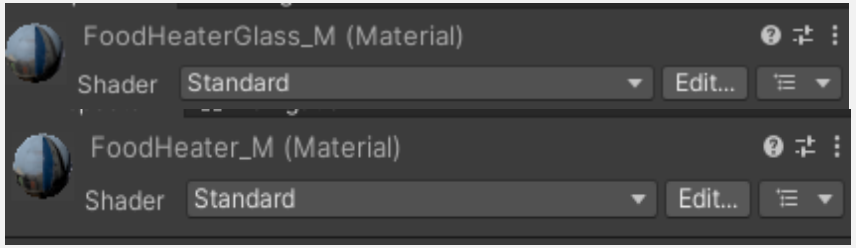



Figure 25.4 - Food heater's transparent and opaque material.

# [Semester 2] Week 3: Model and Texture New Keypad

Following the new keypad panel (Figure 13.3, p.15), it's no longer the colourful keypad because the player doesn't need to look for colours anymore, we've decided that the NPCS will tell the player a number for the code in the new game concept.

I chose to make the keypad with cubes (Figure 26.1) and just softened the edges for the buttons. Adding bevels to recreate the curved edges, of the keypad panel reference, are just unnecessary for this 3D model because it's not meant to be viewed up close, that's what the panel is for.

When I was unwrapping the UV (Figure 26.2), I made sure that the cubes representing the keys were in order, just like on the model, because a clean UV makes the texturing workflow efficient.

Using height on a brush, I created a small indent for the area where the numbers appear (Figure 26.3), to indicate to the player that there's a code that needs to be put in there. It's just an extra detail along with the number pad.

I couldn't find the exact same font that Unity used as default, so I picked a similar sans-serif font "Calibri Light" (Figure 26.4), to imitate the font used on the keypad panel (Figure 26.5).


Figure 13.3, p. 15

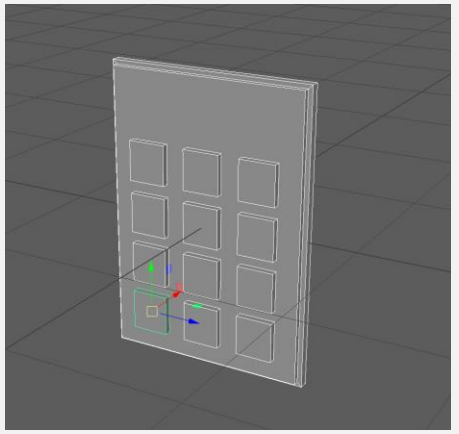


Figure 26.1 - Keypad 3D model.

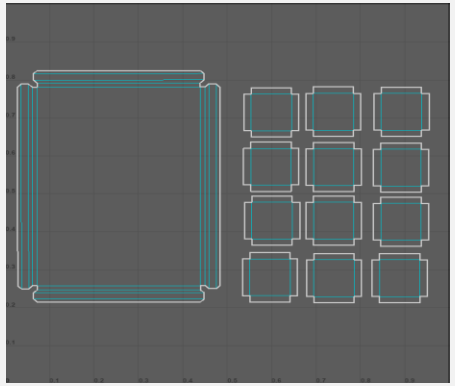


Figure 26.2 - Keypad UV map.

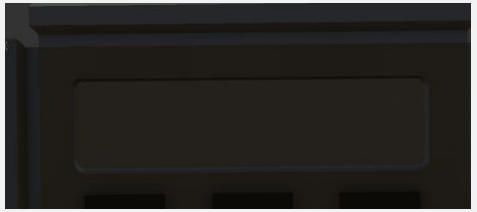


Figure 26.3 - Added height to keypad texture.

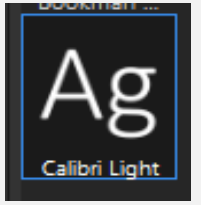



Figure 26.4 - Calibri Light font.

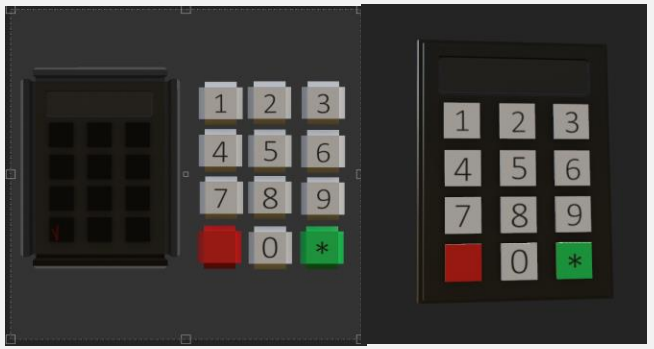



Figure 26.5 - New textured Keypad.

# [Semester 2] Week 4: 3D Modelling Janitor Keycard and Leftover tray

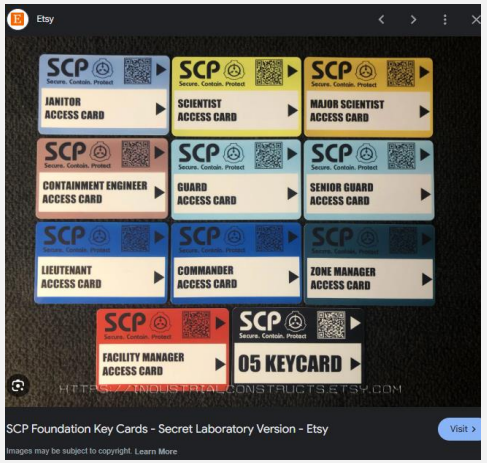



Figure 27.1 - keycard reference.

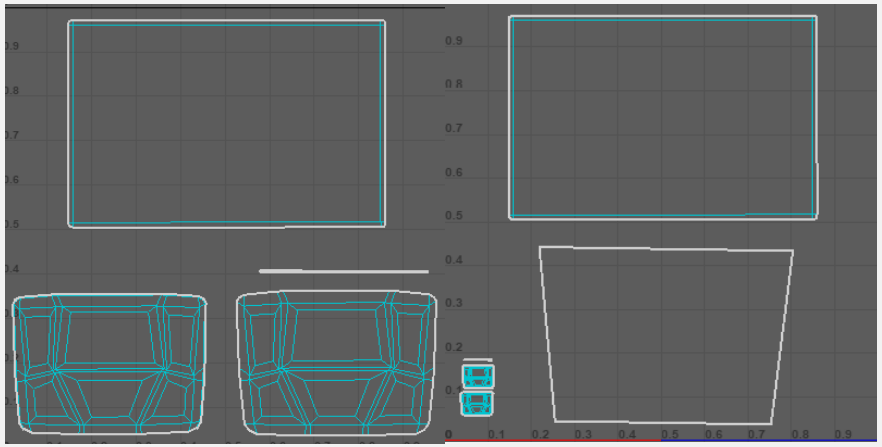


Figure 27.2 - Tray and keycard UV maps variations.

I already had a tray that I 3D modelled before, which was part of the food heater so I just had to 3D model the keycard (Figure 27.2). The keycard is a simple 2D plane where I added extra edge loops to curve the corners, to make the plane look more like the shape of the SCP keycards that I took inspiration from (Figure 27.1).

I kept the card as a plane because I plan to add details on it in substance painter to keep the poly-count low (Figure 27.3).

Since there was space on the UV, I made the tray's UVs to be larger, to make use of the space. I kept them small in the duplicated tray, because I needed more space for the plane. The plane where I plan to add more detail for the food.

By the end of this 3D modelling session, I had two different UV maps to choose from before texturing them in Substance painter.

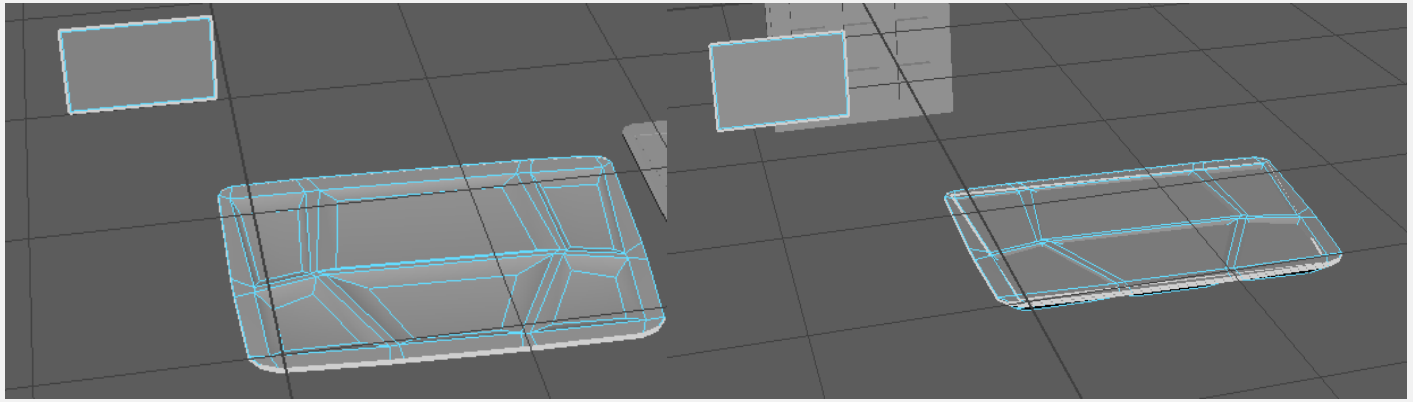


Figure 27.3 - Tray and keycard 3D model variations.

# [Semester 2] Week 4: Texturing Janitor Keycard and Leftover tray

When texturing the food tray, I decided to pick the UV map which didn't have the plane for the food, I decided that the first UV map I made gave enough space to add detail for the food without compromising the size of the tray itself (Figure 28.3).

I followed the colour palette of the food heater (Figure 28.4), to make the cafeteria grub on the tray match with it.

To add texture to the food, I chose the "Human Female 30s Face #06" material (Figure 28.1) again to imitate mysterious meat and used the alpha brush "Stain Napkin island" with increased height to add mold and imitate greens on the mesh (Figure 28.2).

I wanted to pick a sci-fi font for the keycard, cause even if the theme of the overall environment is behind it's time, it's technology is not. I wanted to show that contrast, so I chose (Figure 28.5) "Black Ops One-Regular" as the font for the keycard.

The tray and the keycard are both quest items that appear after you talk to the correct NPC for them.

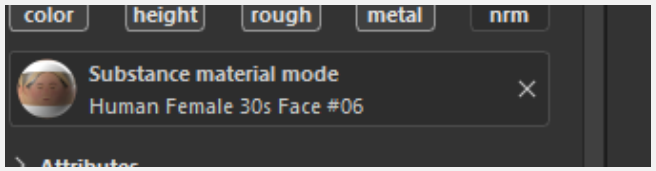



Figure 28.1 - Human Female 30s Face #06 Substance Painter material.


Figure 28.2 - Stain Napkin Island Alpha brush.

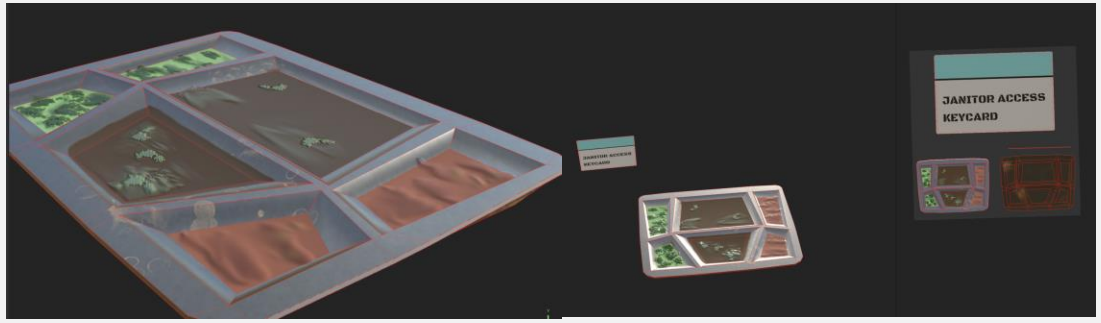



Figure 28.3 - Textured tray and Janitor's keycard.

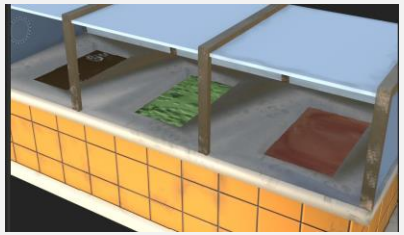


Figure 28.4 - Focus on food texture

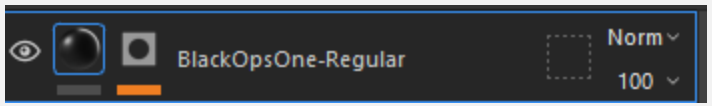



Figure 28.5 - BlackOpsOne Regular font.

# [Semester 2] Week 4: Import Items

When I imported the 3D models I made for the quest items. I had to re-set up the scripts so that they find the correct game object (Figure 29.1) not the placeholder one from before.

I also had to make sure that the soap and the missing tray for the lunch lady is a prefab with their NPCs having the updated prefabs on the "Item To Spawn Prefab"(Figure 29.2) field.

After adding colliders to the items, I placed the left-over tray next to the rubbish bin at the cafeteria (Figure 29.3). The axolotl NPC, is the one that asks for leftovers from the cafeteria. It shows a bit of narrative, letting the player know that the axolotl NPC will eat anything, even if it's rubbish scraps or lunch trays found on the floor.

I put the bar of soap at the same spot as it's placeholder (Figure 29.4), at the shower obstacle course which you give to one of the NPCs in the bathroom.

The missing tray is a lunch tray that spawns in the goat NPC's prison cell (Figure 29.5) after you get their book back from the axolotl. It currently spawns in the ground for an odd reason but it works so I decided to leave it for now.

The janitor's keycard is placed in the corner of the infirmary (Figure 29.6), I realise here that the colour of the keycard helps it stand out, so it won't be hard to find when the player goes to explore the card.

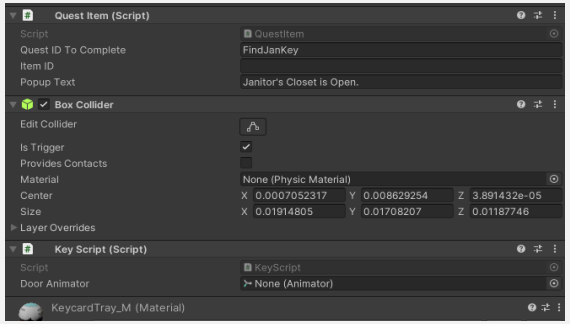



Figure 29.1 - Quest item inspector set up.

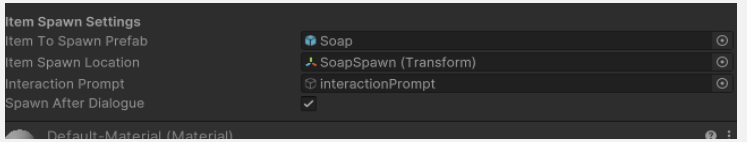



Figure 29.2 - Soap prefab set up.


Figure 29.3 - Adding in leftover food tray to cafeteria.

Figure 29.4 - Adding in soap to showers obstacle course.

Figure 29.5 - Adding in lost tray into Goat NPC's room.

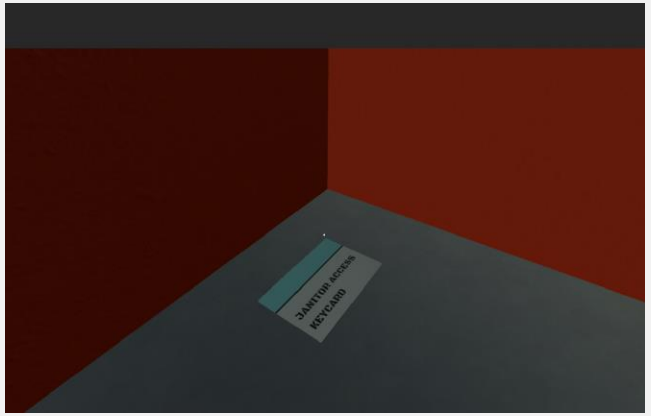



Figure 29.6 - Adding in Janitor's keycard in infirmary.

# [Semester 2] Week 4: Prison Room set up

I made all the prison cells have one bed and one toilet (Figure 30.1) and I didn't personalise them to reflect each prisoner's personality because I wanted to reflect their identities being stripped away, which relates to the theme of control.

I then researched how to add text onto meshes in Unity because I wanted to learn if it was possible to add text in without a canvas. I came across this quick tutorial (myDev learning, 2023) who covers how to add text as a game object. All I had to do was click text mesh pro under 3D objects when you right click the hierarchy in Unity. Doing this allowed me to add text on any mesh I wanted.

Using this reference image of how old prisons numbered off their cells (Figure 30.2), I began to number off the prison cell walls (Figure 30.3) accordingly. I used the default text for now, but I plan to add a sci-fi inspired text like the black ops one to the walls to make it match the prison's vibe.

For now, I've put the text at every area that needs labelling like the cafeteria (Figure 30.4) and any hallways to areas that are hard to find for the players (Figure 30.5), to prevent confusion which was reported to us the last time we had someone playtest our game.

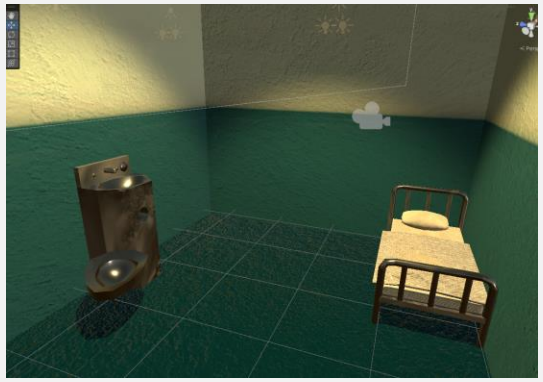


Figure 30.1 - Final prison cell room set up.

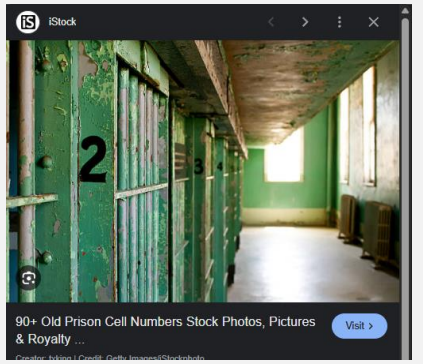



Figure 30.2 - Cell numbers reference.


Figure 30.3 - Cell numbers text on mesh prototype.

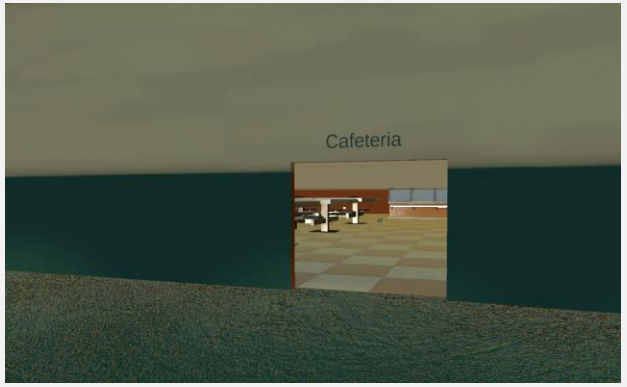



Figure 30.4 - Cafeteria sign prototype.


Figure 30.5 - Sign to showers prototype.

# [Semester 2] Week 5: Infirmary Set up

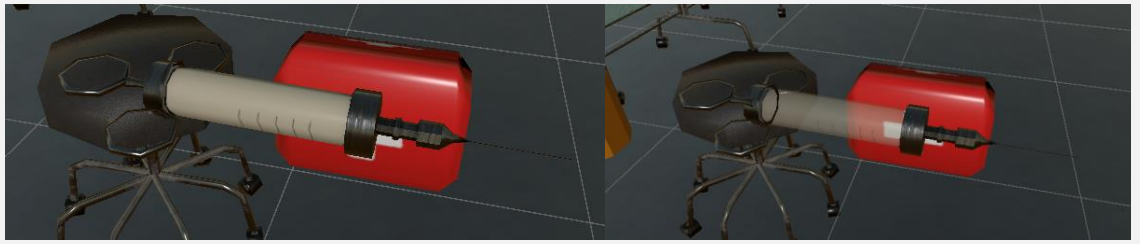


Figure 31.1 - Imported stool, syringe and first aid into infirmary.

I made two materials for the syringe (Figure 31.1) so that the glass can be see-through. I did this so the dust and rust on the syringe can be highlighted. This adds to the aesthetic of our abandoned underground prison.

I ran into an issue with the medical cabinets. The material wasn't appearing properly (Figure 31.2), the metal material was intersecting with the wooden material on the side of the cabinet.

All I had to do was re-export the medical cabinets from Maya and apply the material again, it fixed the issue (Figure 31.3).

I duplicated the medicine and placed them inside the cabinet (Figure 31.4). I chose to do this to enhance our game's theme and make the infirmary feel more alive. Also, it reflects the nurse's personality since the infirmary is the room she has control over.

The medicine being colour coded, highlights her need for cleanliness. That's just one of the many assumptions for her character.

Figure 31.2 - Texture error on medical cabinet.

Figure 31.3 - Imported medical cabinets into infirmary.

Figure 31.4 - Duplicated medicine into cabinets.

# [Semester 2] Week 5: Infirmary Set up 2

Using the collection of reference images, I found of infirmaries and medical wards (Figure 32.1) belonging to different time periods, I am planning to follow them as inspiration for setting up the infirmary.

Taking inspiration from the reference image that features beds lined up on the sides only, the one that looks like it's from the 1950s, I began to put the beds lined along the sides of our infirmary (Figure 32.2). This reflects that the infirmary has order in the space, which can also reflect control.

After that I chose to place the low budget room dividers (Figure 32.2) in-between each bed to provide a little privacy for each prisoner that must come into the ward.

Taking inspiration from the other reference image of the medical ward, the one showcasing beds lined up in between the beds on the sides, I chose to add a row of beds in the middle too (Figure 32.2).

I did this approach because I wanted it to be seen as a reflection of control through the nurse's perspective. Her obsession with cleanliness could also be an obsession of working, making space for prisoners even if it's past the capacity of the prisoners she can take care of.

Using the modern infirmary reference image, I lined up my cabinets like so on the wall. I also added more clutter by duplicating the first aid kits and syringes on the tabletops (Figure 32.3). I chose to drop some first aid kits and syringes in random places like the bed, to show that even if the nurse is a perfectionist, she still makes mistakes.

On the other end of the infirmary, where the current NPC is (Figure 32.4), I made the corner of the medical cabinets look destroyed. I chose this to reflect that something bad happened because the NPC reports that the nurse has gone missing. The destruction shows that the obsessive ex is behind the crime scene because of the broken glass scattered all over the floor (Figure 32.5).

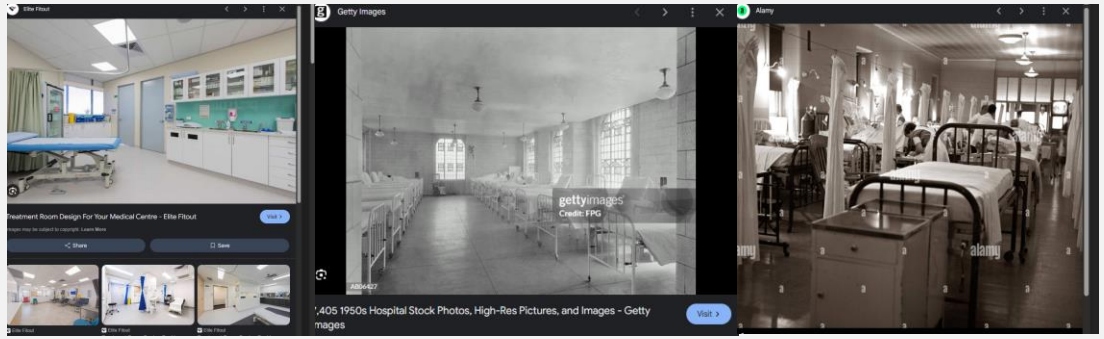



Figure 32.1 - Reference images of infirmary through different ages.

Figure 32.2 - Infirmary beds and dividers set up.

Figure 32.3 - Infirmary medical cabinets and clutter set up.

Figure 32.4 - Adding in temporary NPC into infirmary.

Figure 32.5 - Destroyed medical cabinet door on floor.

# [Semester 2] Week 5: Showers Set up

I used the same reference image that was followed for making the 3D models of the showers (Figure 28, p.19). I chose this for cohesiveness.

The current set up was quick because I only had to duplicate this set up for all the shower stalls (Figure 33.1). It helps the showers look like it's usable and in somewhat livable poor conditions.

I also stuck with the unsanitary concept, following the narrative, there's only one bar of soap that all the prisoners share in that block. Doing this encourages player to purposefully interact with the obstacle course to get the bar of soap for information.

Following a tutorial (LordEvilM44, 2022) on what settings to use to create a rain particle system for the open ends of the pipes (Figure 33.2), I went ahead and made my own adjustments when I found out what settings were mentioned in the tutorial.

I adjusted the shape of the particle system to be a box, this ensures that the particles go down vertically. I also adjusted the alpha channel to "142" because (Figure 33.3) I didn't want the sewage water to opaque. In my opinion it ruins the immersion and can be distracting to the parkour.

I turned on 3D Start Size (Figure 33.4) and adjusted the values so that they start small and end small. The in-between is set at 1 because I want the droplets to be seen as they fall.

I chose to set the start lifetime to "1" and the start speed to "0", so that it's not so fast when the particles spawn (Figure 33.5). Then I adjusted the colour to a murky turquoise to show that the water the prisoners are showering with are contaminated (Figure 33.6).

Figure 28, p. 19

Figure 33.1 - Shower stall set up.

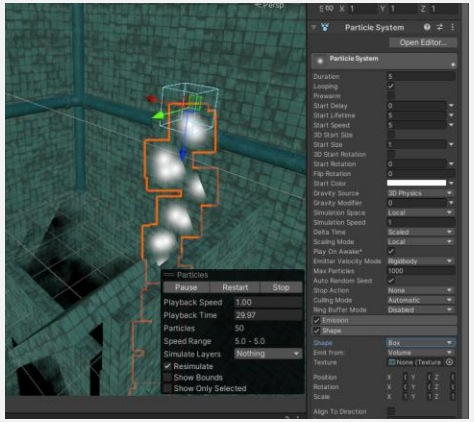



Figure 33.2 - Pipe particle system.

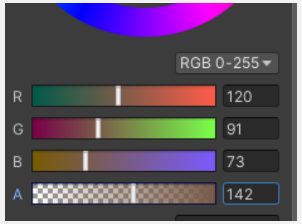



Figure 33.3 - Updated the alpha channel to 142.


Figure 33.4 - Adjusted 3D Start size values.

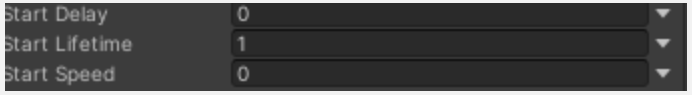



Figure 33.5 - Start speed.

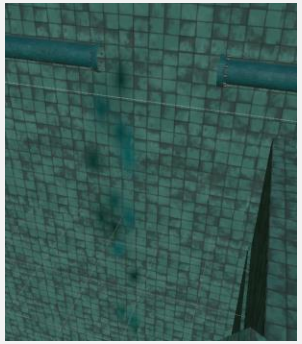


Figure 33.6 - Final pipe particle system.

# [Semester 2] Week 5: Creepy love notes 1

To add more showcase more narrative, with minimal player agency, we decided it'd be a cool idea to add little love letters scattered around the map. This is to indicate the player that their ex is stalking them before the ex gets officially transferred to the player's prison. It relates to Macklin's and Sharp's (2016) on how direct actions and movement can determine how much control a player has in an environment because I've decided to give the player a choice to have more control or less control, they can either skip these love notes which provide more narrative for the player to understand the antagonist, or they can just let the environment control their direction by focusing on the objective to escape the prison.

We wrote a list of important love notes (Figure 34.1), to help us identify each love note's purpose.

I compiled together a quick script using OnTriggerCollider() functions (Figure 34.2), to handle the logic for opening the note, interacting with the note with the keycode "F" and making sure LoveNote.cs finds the correct collider through a boolean and opens up the correct note.

I placed a 3D plane on the bed (Figure 34.3) and used this filler reference image (Figure 34.4) to test out the code to make sure everything is functioning intentionally. I also learned ("Cursor.lockState - Scripting API", n.d.) how to unlock the cursor when the image pops up and cleaned the code so that the camera stops while you're reading the note. I did this because having the camera move around while the screen would pop up, is distracting to the experience.

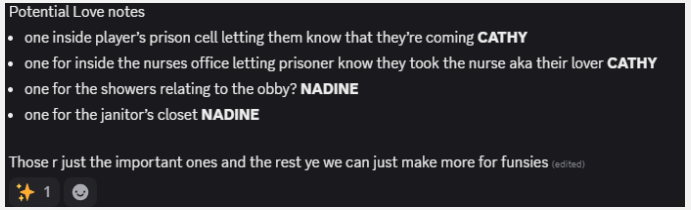
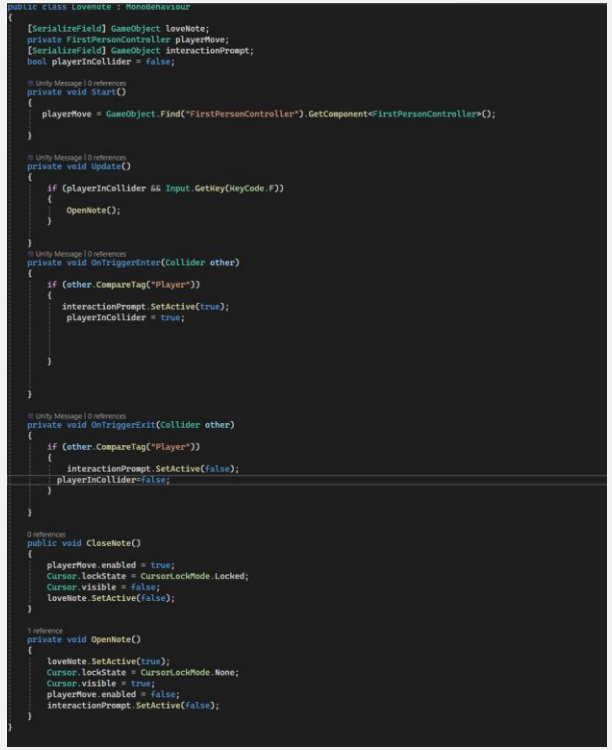



Figure 34.1 - Screenshot of love notes plan on Discord.



```
public class LoveNote : MonoBehaviour
{
    [SerializeField] GameObject loveNote;
    private FirstPersonController playerMove;
    [SerializeField] GameObject interactionPrompt;
    bool playerInCollider = false;

    // Unity Message | 0 references
    private void Start()
    {
        playerMove = GameObject.Find("FirstPersonController").GetComponent<FirstPersonController>();
    }

    // Unity Message | 0 references
    private void Update()
    {
        if (playerInCollider && Input.GetKey(KeyCode.F))
        {
            OpenNote();
        }
    }
    // Unity Message | 0 references
    private void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("Player"))
        {
            interactionPrompt.SetActive(true);
            playerInCollider = true;


        }
    }

    // Unity Message | 0 references
    private void OnTriggerExit(Collider other)
    {
        if (other.CompareTag("Player"))
        {
            interactionPrompt.SetActive(false);
            playerInCollider=false;
        }
    }

    0 references
    public void CloseNote()
    {
        playerMove.enabled = true;
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible = false;
        loveNote.SetActive(false);
    }

    1 reference
    private void OpenNote()
    {
        loveNote.SetActive(true);
        Cursor.lockState = CursorLockMode.None;
        Cursor.visible = true;
        playerMove.enabled = false;
        interactionPrompt.SetActive(false);
    }
}
```

Figure 34.2 - Screenshot of Love Note Script.

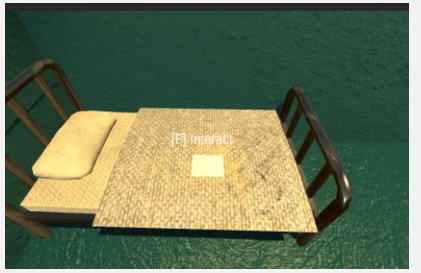



Figure 34.3 - Imported love note game object.


Figure 34.4 - Imported place holder image for love note.

# [Semester 2] Week 5: Creepy love notes 2

This is the template that I made on procreate for the love notes (Figure 35.1). Its purpose is for the player to visually see the note once it gets attached on a plane (Figure 35.2).

The brush I used for the words is called "Water Pen" under calligraphy and I used "Flat Brush" under painting (Figure 35.4) to smudge the text. I chose these brushes because I wanted the words to look like they were written messily with lip stick and the flat brush helps to smudge the water pen so that it looks like lip stick stains.

The paper I used was a combination of an A4 piece of paper at home (Figure 35.5) and a paper texture (Saporito, 2022) I found online (Figure 35.6) after applying these on the canvas, I changed the blending mode to hard light so that I can see the details of the paper on the note underneath the writing. It adds to the gross feelings and makes the paper look old; the messy handwriting reflects the antagonist's destructive personality as well.

For the note that the player finds in the janitor's closet, there's blood on the edges, to show that the antagonist, the snake, has already harmed the nurse. The other notes just have silly messages (Figure 35.2) to spook the player but no blood is evident here because the antagonist hasn't crashed out because the player has a relationship with the nurse. I also explored with making polaroids of the player being watched by their ex (antagonist) to show their stalker behaviour (Figure 198). The player is also a grey figure named "Y/N" which stands for "Your name" ; I took inspiration from a video showcasing Y/N (Angelina Cinema, 2025) in an animation compilation (Figure 35.7).


Figure 35.1 - Love note game object design.

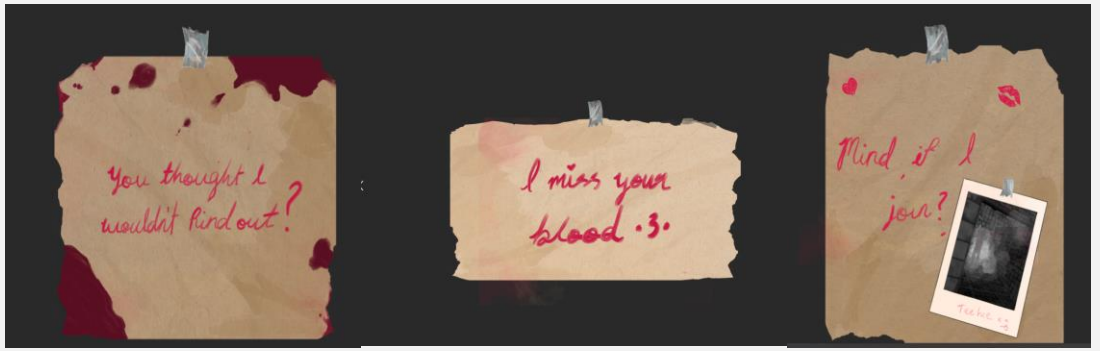



Figure 35.2 - love note variations for janitor's closet, prison area and showers.

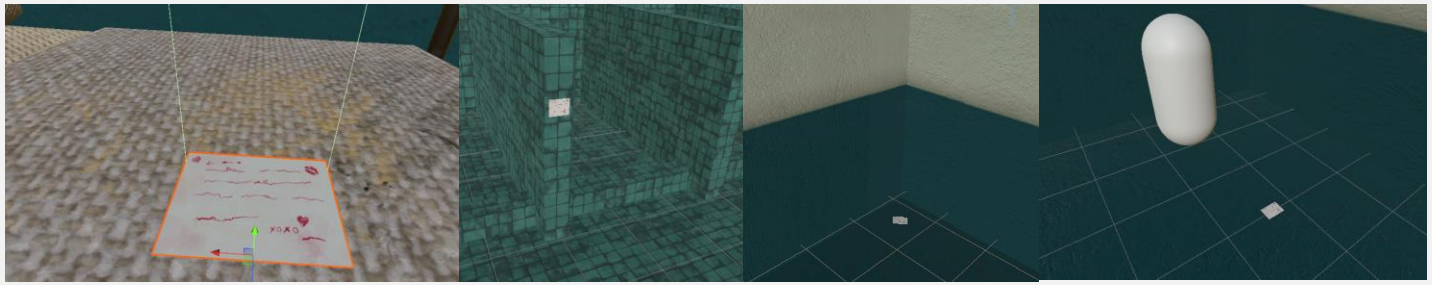


Figure 35.3 - Screenshots of where the love notes are placed in the environment.


Figure 35.4 - Procreate's Water Pen and Flat Brush.

Figure 35.5 - Image of crumpled paper from home.

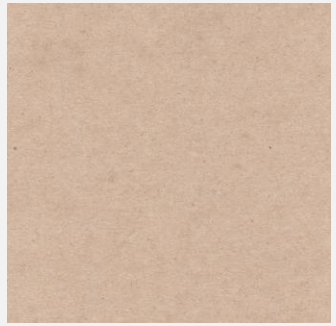


Figure 35.6 - Paper texture. (Saporito, 2022).


Figure 35.7 - Y/N reference. (Angelina Cinema, 2025).

# [Semester 2] Week 6: Audio Mixer and Quest SFX

This week my focus was adding the audio into our game. I followed my lecturer's tutorial from Year 2, on audio mixers to manage my sound effects and music separately (Figure 36.1). This ensures that if any of the audio is too loud, the player can adjust the audio to their liking, this helps them stay focused on the game. Since I had already created an audio manager, the only line of code I needed to add was an UpdateVolume() function to handle the logic for updating the volume using the slider (Figure 36.2). Having sound in our game, helps make the game more immersive which aligns with Sinclair's research (Sinclair, 2020) theory on audio, how the soundtrack can make or break a game.

I added sound effects to the quest item too; it's the sound of someone putting something into their pocket and the logic (Figure 36.4) was just enabling the pick-up SFX. I chose this to indicate to then player that they've collected an item.

Then I just polished the door script by adding the new sound effect I edited (Figure 36.5). I did this so that the sound effect of the door wouldn't be the "Oof" sound effect anymore. It was only a test sound effect anyway.

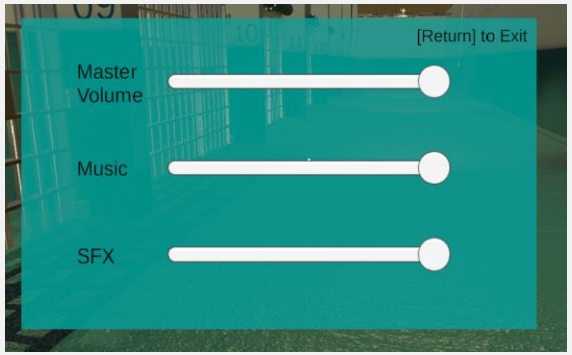



Figure 36.1 - Screenshot of volume slider panel.

```
0 references
public void UpdateVolume(Slider volumeSlider)
{
    audioMixer.SetFloat(volumeSlider.name, volumeSlider.value);
}
```

Figure 36.2 - Screenshot of Update Volume function.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class QuestItem : MonoBehaviour
{
    public string questIDToComplete;
    public string itemID;
    public AudioSource pickUpSFX;
    public string popupText;

    private void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("Player"))
        {
            if (!string.IsNullOrEmpty(itemID))
            {
                Inventory.Instance.AddItem(itemID);
                Debug.Log("Picked up item: " + itemID);
                pickUpSFX.enabled = true;

            }

            if (!string.IsNullOrEmpty(questIDToComplete))
            {
                QuestManager.Instance.CompleteQuest(questIDToComplete);

                Debug.Log("quest complete");
                UIManager.Instance.ShowPopup(popupText);
                pickUpSFX.enabled = false;
            }

            Destroy(gameObject);
        }
    }
}
```

Figure 36.3 - Updated Quest Item script.

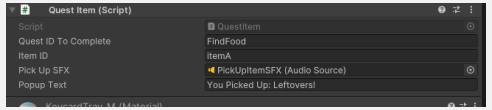



Figure 36.4 - Quest item inspector set up.

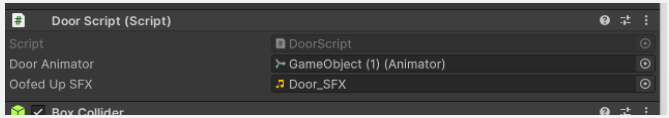



Figure 36.5 - Door script inspector set up.

# [Semester 2] Week 6: Audio Room Triggers Escape Room Scene Player Footsteps 1

My goal here was to have different background music in each room. I set up the audio sources that I needed for each room (Figure 37.1). Then the approach I took to make this happen was to add a box collider trigger in each room (Figure 37.2) and attached a new script called "Room audio" on it, this script has OnTriggerEnter/Exit functions that let you enable your chosen audio source when you enter the collider (Figure 37.3) and it stops when you leave.

I chose this approach, the cafeteria is loud with chatter and the prison cell is an area where the prisoners chill or want to be alone, the janitor's closet has intense boss music because it's the reveal of something bad happening in the prison, the infirmary has ambient noise to add suspense when the player finds out that the nurse is missing and the showers has ambient sounds like water dripping to show that the area is not managed. Since each room has a different feel to it, I chose to give them different background music.

To add more immersion for the player, I implemented footstepsSFX logic into the audio script on the player (Figure 37.4) and made an audio source with the spatial blend at 2D so that the player can always hear their footsteps. (Figure 37.5)


Figure 37.1 - Audio game objects.

```
private void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("Player"))
    {
        roomAudio.Play();
        secondroomAudio.Play();
        playerInAudioCollider = true;
    }

}

private void OnTriggerExit(Collider other)
{
    if (other.CompareTag("Player"))
    {
        roomAudio.Stop();
        secondroomAudio.Stop();
        playerInAudioCollider = false;
    }
```

Figure 37.2 - Room Audio Script screenshot.

Figure 37.3 - Room audio trigger set up.

```
void Update()
{
    if (Input.GetKeyDown(KeyCode.Escape))
    {
        volumePanel.SetActive(true);
        Cursor.lockState = CursorLockMode.None;
        Cursor.visible = true;
        playerMove.enabled = false;
        escapeButton.SetActive(false);
    }
    else if (Input.GetKeyDown(KeyCode.Return))
    {
        volumePanel.SetActive(false);
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible = false;
        playerMove.enabled = true;
        escapeButton.SetActive(true);
    }

    if(Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.S) || Input.GetKey(KeyCode.D))
    {
        footstepsSFX.enabled = true;
    }
    else
    {
        footstepsSFX.enabled = false;
    }
```

Figure 37.4 - Implemented footsteps SFX logic into audio script of player.

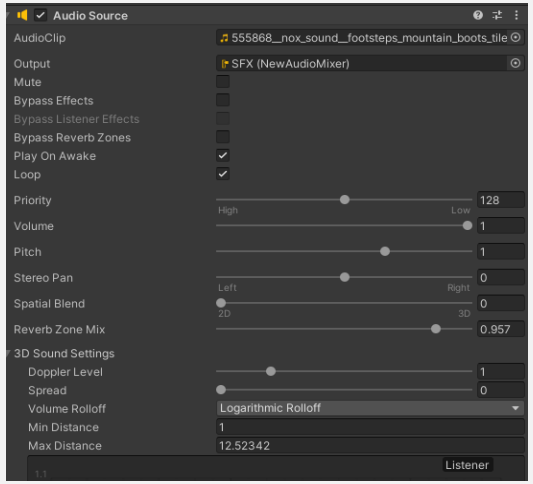



Figure 37.5 - Audio source footsteps set up.

# [Semester 2] Week 6: Escape Room Scene Player Footsteps 2

I went on Wondershare Filmora and edited the sound effect for the footsteps (Figure 38.1) so that the walk, jump, and sprint audio is split up (Figure 38.2).

The audio source is set up without loop (Figure 38.3) so that it doesn't keep playing after the player hits "space".

The player audio script got updated based off of Omogonix's tutorial (Omogonix, 2022a) on how to set up footsteps in Unity. ChatGPT helped add the jump logic, saying to put "PlayOneShot(jump.SFX.clip)" instead of enabling the audio source like I did for the sprint logic and the walk logic (Figure 38.4).

This ensures that the sound effects match with the keybinds, it breaks the immersion when the footsteps don't match with the keybinds.

Following another tutorial by Omogonix on footsteps on water (Omogonix, 2022b) I created a new script called Water Audio which holds the logic (Figure 38.5) for turning on the water footsteps sound effects audio when they enter the trigger (Figure 38.6).

I did this to so that the stone walking SFX and the water walking SFX overlap, making it sound like the player is walking through water. It adds to the grimy environment of the showers.

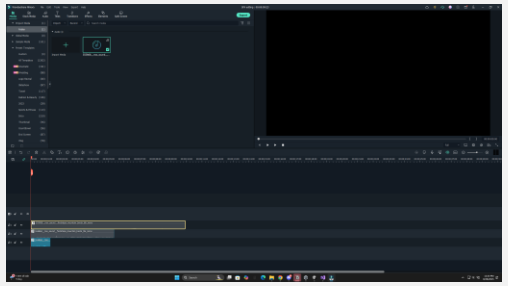


Figure 38.1 - Screenshot of filmora.

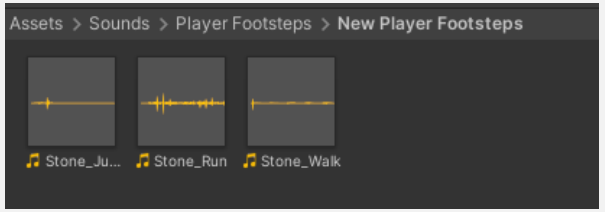



Figure 38.2 - Footstep audio variations.

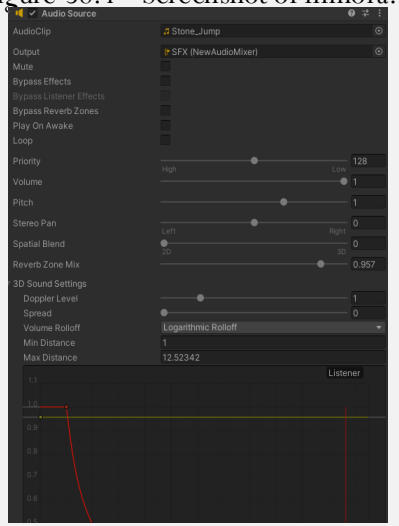
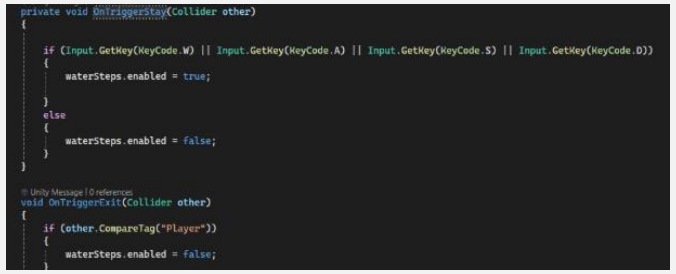



Figure 38.3 - Jump audio source set up.

```
        footstepsSFX.enabled = false;
        sprintSFX.enabled = false;
    }

    if (Input.GetKeyDown(KeyCode.Space))
    {
        jumpSFX.PlayOneShot(jumpSFX.clip);
    }
```

Figure 38.4 - added jumpSFX in player audio script.



```
private void OnTriggerStay(Collider other)
{
    if (Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.S) || Input.GetKey(KeyCode.D))
    {
        waterSteps.enabled = true;
    }
    else
    {
        waterSteps.enabled = false;
    }
}

Unity Message | 0 references
void OnTriggerExit(Collider other)
{
    if (other.CompareTag("Player"))
    {
        waterSteps.enabled = false;
    }
```

Figure 38.5 - Water Audio script.

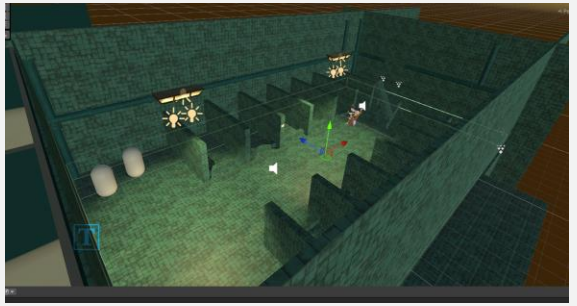


Figure 38.6 - Showers audio room trigger.

# [Semester 2] Week 7: Drawing Game Over Image draft 1

I took a screenshot of the prison cell rooms (Figure 39.1) to draw a draft of the game over screen. This image is the one that pops up after the player dies to the snake, the obsessive ex, in the chase scene. I chose this photo because we decided to make the game over screen show the player being dragged into a prison cell by their ex. It symbolises that the player didn't escape the ex's control.

Using my teammate's character design of the ex (Figure 39.2), I began sketching a draft of the game over screen, beginning with tracing the outlines of the environment (Figure 39.3). I chose to follow my team mates's design because our art needs to be cohesive, it's the same reason why I chose to trace the environment that I made, because I wanted the place to be recognisable.

I used the flat brush to do the line art which is the same brush that my teammate used for designing the characters (Figure 35.4, p. 37). It's a versatile brush that blends easily and gives us sharp edges for line art; I found it good to use for the hazy lighting that's present within our prison environment.

Again, I blended the light areas to make the colours look hazy under the industrial vintage underground facility lighting. (Figure 39.4).

The first version shows the full body of the serpent dragging the player towards the cell. This symbolises the loss of control as the serpent is seen towering over the player.

The second version only shows the arm and the tail of the serpent entering the cell while the player is still dragged. This sketch also symbolises the loss of control but it focuses on the vulnerability of the player because it's the main subject seen in the screen.

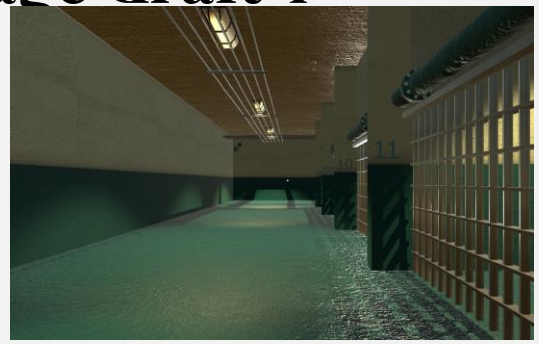


Figure 39.1 - Prison room reference.

Figure 39.2 - Ex-lover reference.

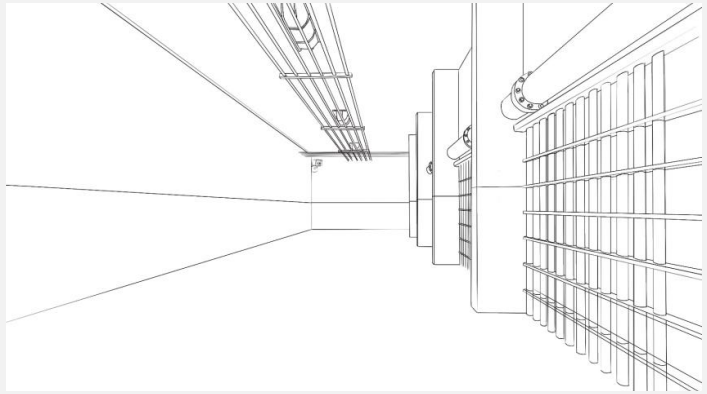


Figure 39.3 - Prison room line art.


Figure 35.4, p. 37

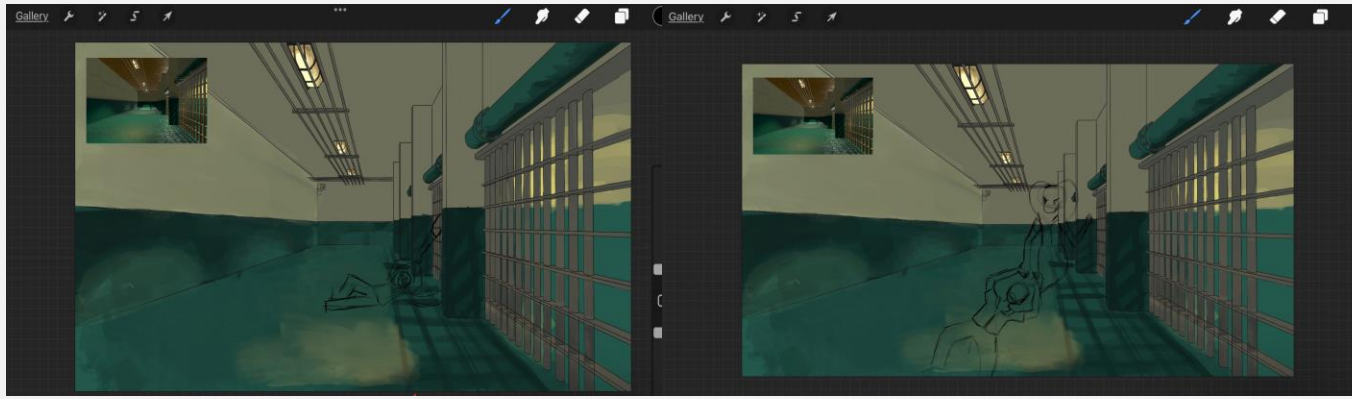



Figure 39.4 - Game over scene variation sketches.

# [Semester 2] Week 7: Drawing Win draft

Figure 40.5

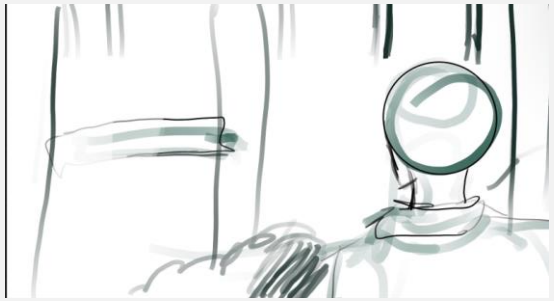


Figure 40.1 - Win sketch option 1.

Figure 40.2 - Win sketch option 2.

Figure 40.3 - Win sketch option 3.

Focusing on the win screen background draft, I drew sketches of potential good endings for the player. (Figure 40.1-40.3). The focus is making the good ending symbolise the feeling of freedom after surviving from the evil ex-lover who's been watching and plotting against Y/N the entire game. Procreate has a library of brushes for drawing nature like "Wild grass" (Figure 40.4) to implement natural grass textures and I used a warm colour palette and drew tiny daisies on a summer's day (Figure 40.5), which contrasts to the cold hard environment of the underground prison shown in the game over screen draft. Once I begin finalising this illustration, I plan to add more tall grass and flowers scattered on the floor to show the beauty of the outside world. This win screen is intended to make the player feel accomplished for finally escaping from the prison.

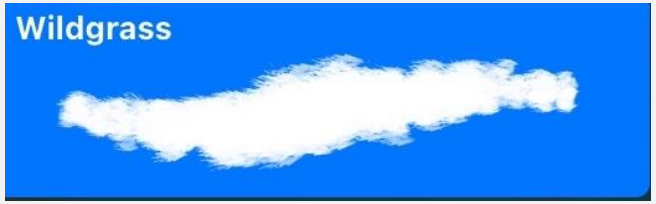



Figure 40.4 - Procreate Wild grass brush.

# [Semester 2] Semester Break: 3D modelling and UV unwrapping Ventilation

Using this image (Figure 41.1) of the vents from *Poppy Playtime - Chapter 2* (Mob Entertainment, 2022) as inspiration for the vents. These vents is going to be placed around the escape room scene and the chase scene. Upon interaction with the correct NPCs, some tell the player that there's strange happenings coming from the vents and the pipes it's a hint that the ex-lover has been stalking and hiding the player through the vents and manoeuvring through the pipes. The implementation of the vents in all areas of the game, gives the player information that there's something in the vents. I 3D modelled 2 variations of the vents in Maya (Figure 41.3-41.4) and a vent lid (Figure 41.2) that's to be animated in the chase cutscene. When I UV unwrapped the vents, I put each vent on a separate UV map, because I wanted to maximise the space, so that the details of the vent that I add in Substance Painter will be noticeable (Figure 41.5).

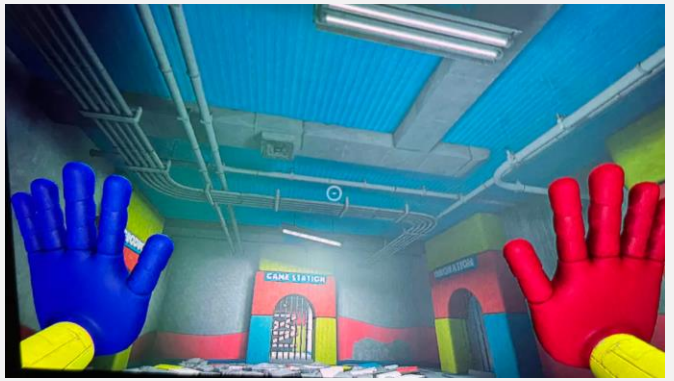


Figure 41.1 - Poppy Playtime Chapter 2 (Mob Entertainment, 2022). Vents.

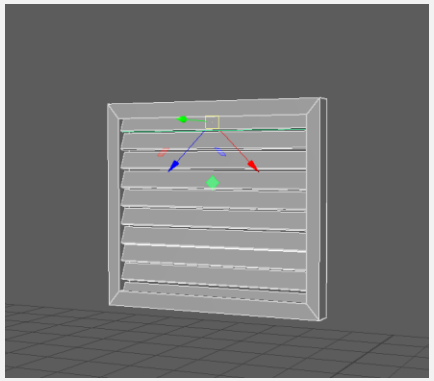


Figure 41.2 - Vent cover 3D model.

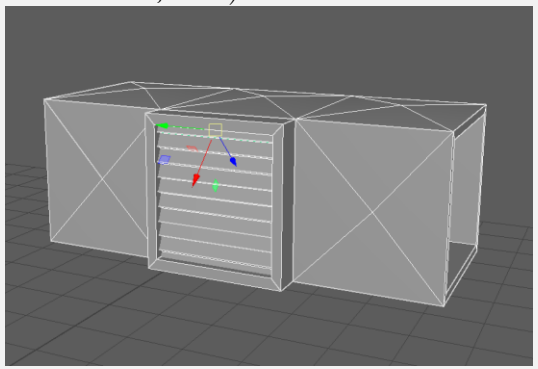


Figure 41.3 - Vent variation 1.

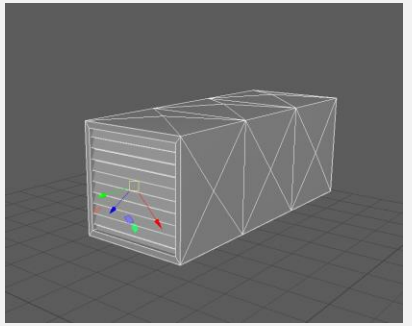


Figure 41.4 - Vent Variation 2.

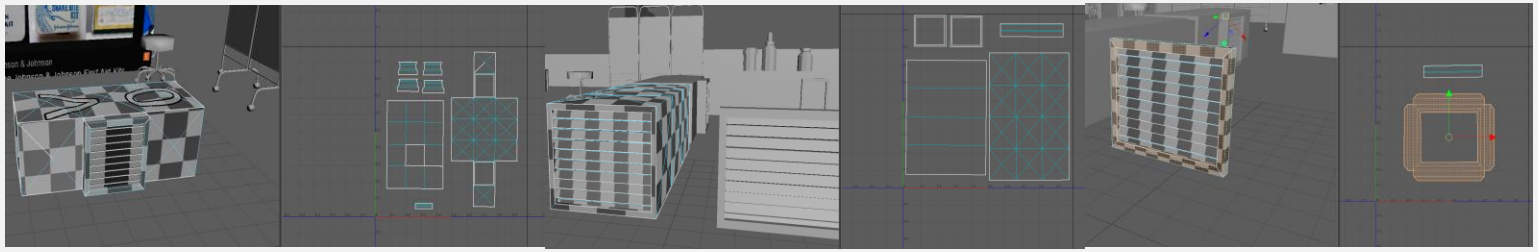


Figure 41.5 - UV maps of all vents.

# [Semester 2] Semester Break: Ventilation Textures an Importing into Unity

When I textured the vents, I used the same method to find an orange colour (Figure 42.1) and material (Figure 22.2, p.24) I've done for the previous 3D models, to imitate rust on the surfaces to showcase deterioration through rust and mold. I've chosen to use the same method so that the vent texture is cohesive with the surrounding vintage/retro abandoned prison environment. To imitate the dent details of a vent, I made two layers with the metal brushed material on them in Substance Painter and changed the rotation of one of them so that the light shines differently (Figure 42.2) on the model. If the material was all facing the same way, the illusion of detail wouldn't be present (Figure 42.3). By doing this I was able to create an illusion of the vent details (Figure 42.7).

The vents are the main hiding spot for the ex-lover, and they're a sign that they're always watching because they're present in every single room which relates to the theme of control and reflects their behaviour towards Y/N. I took heavy inspiration from *Poppy Playtime - Chapter 3* (Mob Entertainment, 2024). How the antagonist, "Catnap" is spying on the player through the vents and shadows in Playcare, the orphanage, to hunt the player down. I believe with this connection, it helps to enhance the experience of control through interactions within the environment.

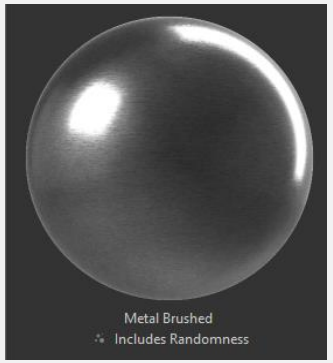



Figure 22.2, p. 24

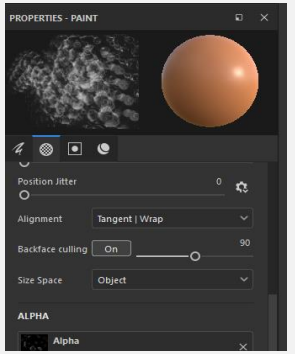



Figure 42.1 - Rust orange.

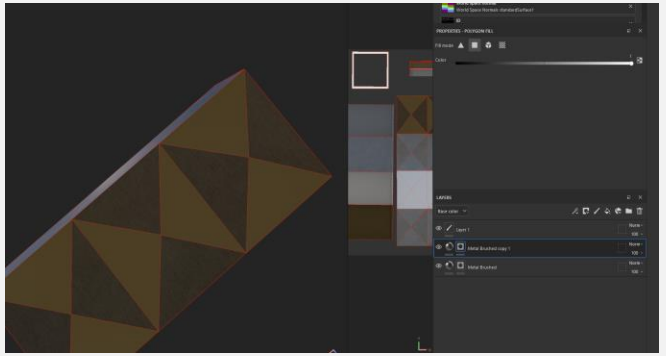


Figure 42.2 - Varying rotation of metal brushed material example.

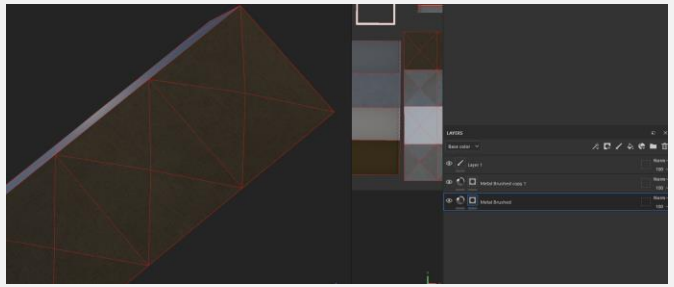


Figure 42.3 - Metal brushed material with no varying rotation.

Figure 42.4 - Vent cover textured.

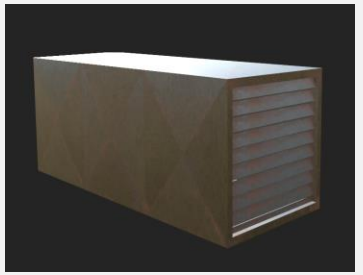


Figure 42.5 - Vent variation 2 textured.

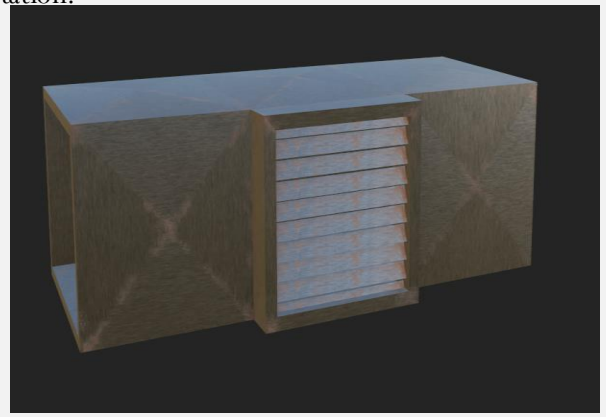


Figure 42.6 - Vent variation 1 textured.

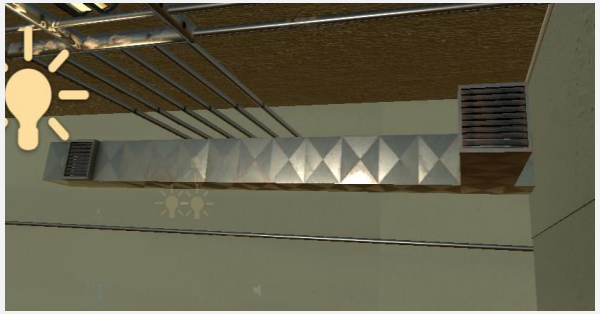


Figure 42.7 - Importing vents into the game.

# [Semester 2] Week 8: Inspiration for Making Posters for Infirmary and Cafeteria

Here is a collation of images (Figure 43.1) I've used as inspiration for the posters, I've illustrated to decorate the environment. This was done in regards to the playtest feedback we got. The environment needed more clutter and I picked posters, this way I can give the player more information about each room that they enter in and it was used to hint at the player of the mistreatment and horrible living conditions of the prison. I made sure to pick vintage/retro references of the cafeteria menu so that the menu will match with the grimy cafeteria.


Figure 43.1 - Inspiration board for the posters.

# [Semester 2] Week 8:Making Posters for Cafeteria

I wanted the posters to be silly, to contrast with the grungy abandoned prison aesthetic. I started off with making a simple menu, this is to be implemented on a black board 3D model I made to recreate the cafeteria menu (Figure 44.1) reference image, I decided that naming the red meat "Mysterious Meat" to symbolise the mistreatment. It shares information to the player about the narrative, that the lunch lady is secretive. I made a "No Junk food" poster (Figure 44.2-44.4) to symbolise that the prison is all about being "healthy" as well as creating a "eat healthy" poster but having the words say "Eat Me." To symbolise (Figure 44.5-44.7) the authority that the staff have over the prisoners, which also reflects control. Outlining the text with black made the message look more bold. I thought dark humour was best to showcase some themes of control through the posters. The software I used was Procreate, and the font I used for these posters was "DIN Condensed." (Figure 44.1).


Figure 44.1 - Menu poster process.

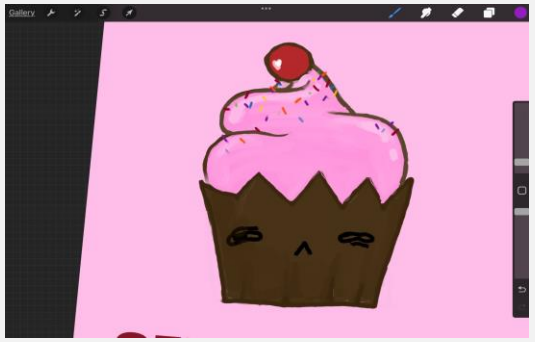


Figure 44.2 - Cupcake illustration.


Figure 44.3 - Adding the red crossed out sign.


Figure 44.4 - Final "No junk food" poster.

Figure 44.5 - Rendering the carrot.

Figure 44.6 - Outlining the text.

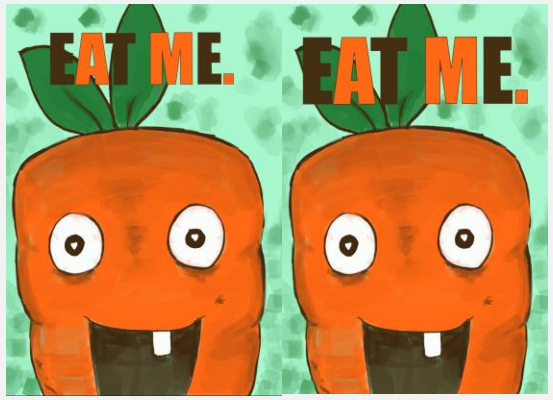



Figure 44.7 - Final "Eat Healthy" poster.

# [Semester 2] Week 8:Making Posters for Infirmary

Keeping the silly theme, I made the Infirmary rules cater to the characters that live in the prison (Figure 45.1), it reflects the strictness of the nurse and highlights her controlling behaviour and obsession for cleanliness and order. I thought that using the font Impact (Figure 45.3) can emphasise her dominance in the space. I used "Jack Armstrong BB" as the body text in the "Wash your hands" (Figure 45.4) because its design has a comedic (Figure 45.2), comic style which contrasts with the harness of the rules, maintaining the game's dark humour tone and while still reflecting control. Same goes for the donate blood poster(Figure 45.5), the text saying it's compulsory hints that the inmates don't have control or the right to say no to donating blood, which adds to the narrative of their mistreatment. Adding these posters enhance Lover's Cell's storyline by reflecting the significant characters' personalities in the environment. Adding these posters in as clutter can help the player come to their own conclusions leaving them to decide how they want to engage with the environment.


Figure 45.1 - Final "Infirmary Rules" poster.


Figure 45.2 - Using Jack Armstrong BB font.


Figure 45.3 - Using Impact font.


Figure 45.4 - Final "Wash your hands poster."


Figure 45.5 - Final "Donate blood" poster.


Figure 45.6 - Importing the posters into the game.

# [Semester 2] Week 8: Janitor's Closet 3D modelling, UV unwrapping and Texturing

Within Maya, I created two groups of models (46.1-46.2), to provide clutter in the janitor's closet. I moved them in the spot I wanted them to be, so that it'll be easy to implement into Unity later. Using the reference images I googled (Figure 46.3), I drew an easter egg design to go on the caution wet floor sign (Figure 46.4) in Procreate. If viewed, it shows a teaser of what animal the ex-lover is, a snake, slipping on a floor. The materials I used (Figure 46.5) are for making the mop (Figure 46.6)and the bucket to create a worn-out look to fit the poor quality of objects in the environment.

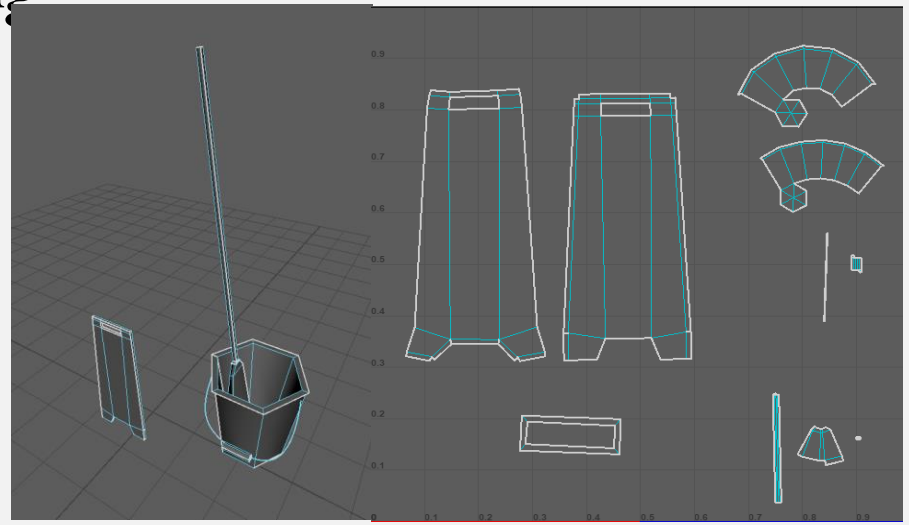


Figure 46.1 - 3D model and UV map of janitor's mop bucket and sign.

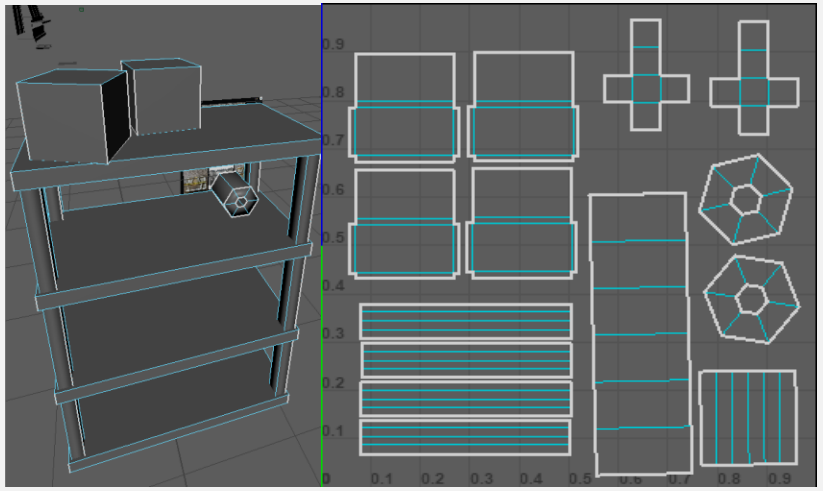


Figure 46.2 - 3D model and UV map of janitor's shelf.


Figure 46.3 - Caution wet floor reference.

Figure 46.4 - Final design for caution wet floor.

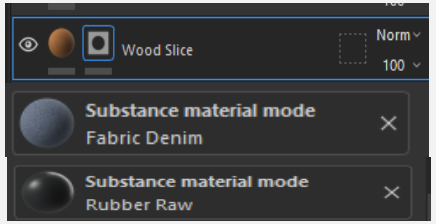



Figure 46.5 - Wood Slice, Fabric Denim and Rubber Raw Substance Painter materials.


Figure 46.6 - Textured mop bucket and sign.

# [Semester 2] Week 8: Janitor's Closet Texturing 2

Figure 47.1 - Box reference image and textured box.

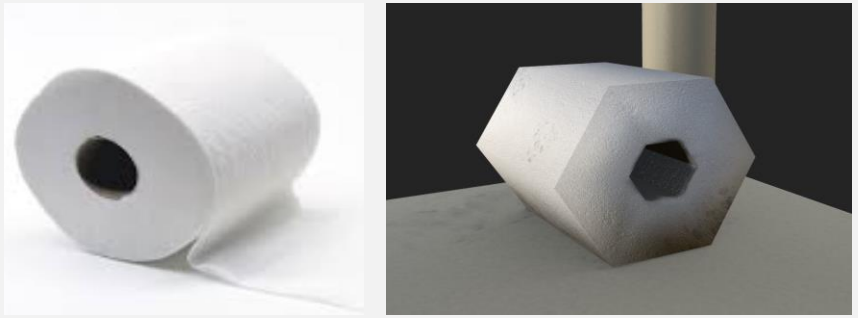


Figure 47.2 - Toilet paper reference and textured toilet paper.

Using the reference images, I found for the cardboard box and toilet roll (Figure 47.1-47.2), I collated materials (Figure 47.4) to recreate the look of these mundane objects you can find in the janitor's closet. I coloured these objects with an alpha stains brush to imitate (Figure 24.6, p. 26) mold spots on the boxes, toilet roll and shelf. This adds to the poor quality (Figure 47.3) of items given to the inmates which reflects the abandoned aesthetic of the environment. I also think that adding clutter in the janitor's closet will improve the simulated space and enhance the game feel.

Figure 47.3 - Textured Janitor's shelf.

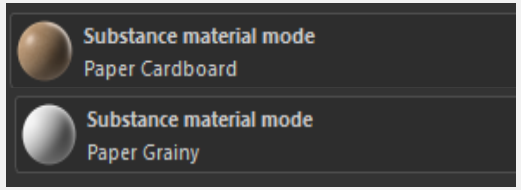



Figure 47.4 - Paper cardboard and Paper Grainy Substance Painter materials.

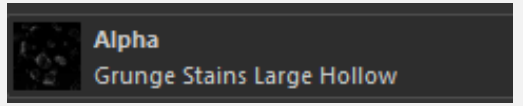



Figure 24.6, p. 26

# [Semester 2] Week 8: Janitor's Closet Set Up

In addition to the assets I 3D modelled for this room, I duplicated stacks of the soap 3D model I made for the showers (Figure 48.2). Previously I mentioned that there's one old bar of soap in the showers so that the player is pushed to complete the NPC's quest to find the only bar of soap that everyone shares, but here there's stacks of new bars of soap that are kept from the prisoners' knowledge. This adds to the controlled and poor living conditions that the inmates experience within the underground prison. I created a new material and made it bright pink (Figure 48.1) in contrast to the waxy colour of the soap the player must collect in the showers. Using inspiration from this janitor closet image I found (Figure 48.4) I set the janitor's closet  (Figure 48.3) to look like everything is messy, and that there isn't a lot of stuff in storage apart from the soap which implies that shipment of new necessities doesn't happen in this environment.

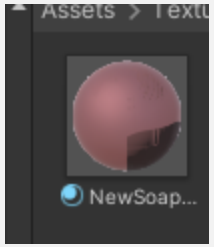



Figure 48.1 - New soap material in Unity.

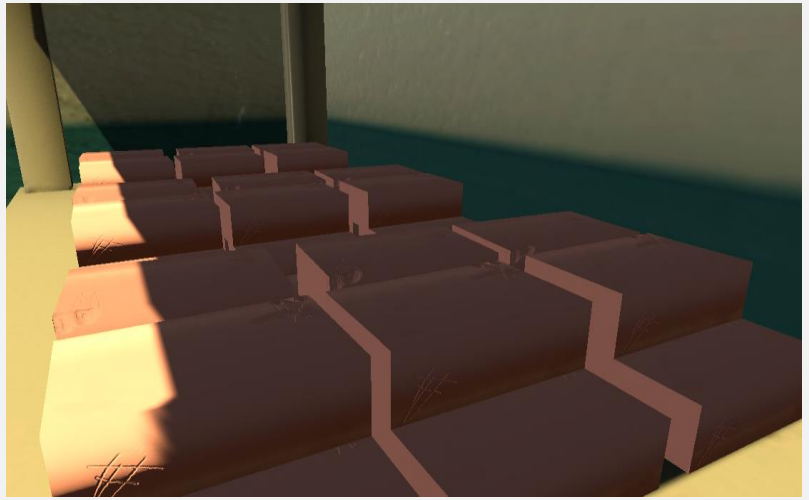


Figure 48.2 - Soap clutter in janitor's closet.

Figure 48.4 - Janitor's closet reference.

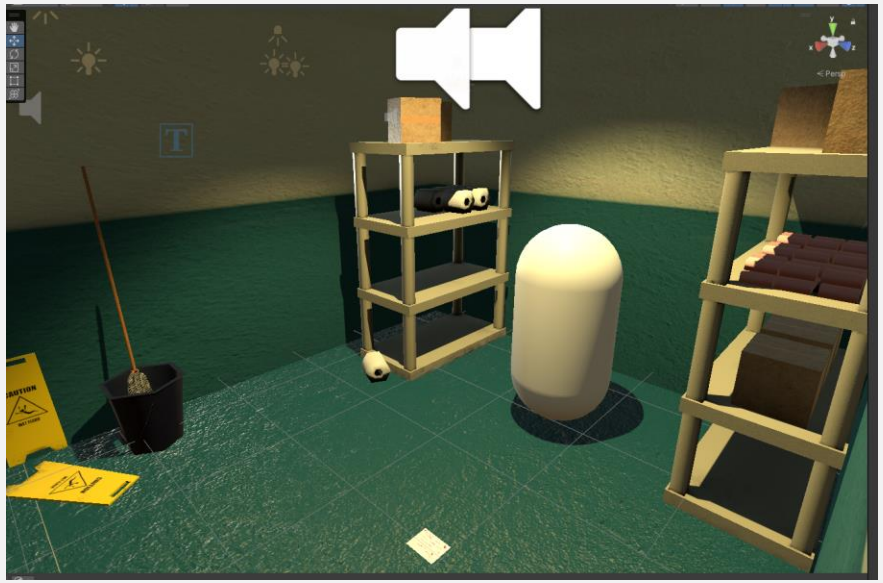


Figure 48.3 - FInal level design of Janitor's closet.

# [Semester 2] Week 8: Adding 3D environment and animating in chase cut scene

As a duo, we created a mock-up of the chase cut scene, this is the scene that happens when you unlock the door from the main prison escape room to enter the staff hallways maze to find the exit. I oversaw importing in the environment objects. The narrative here is that it's the big reveal on Y/N's ex-lover. It's also a scene where the player doesn't experience much control on their end and can't stop the natural narrative from playing in front of them, we only give them time to prepare themselves to run in the next scene. We thought the best way was to showcase them hopping out of the vents (Figure 49.1-49.2) and some text (Figure 49.3) to indicate to the player that the ex-lover is kind of like the final boss of this escape room. To make this we used Unity's timeline system (Figure 49.4).

Figure 49.1 - Screenshot of vent in timeline sequence.

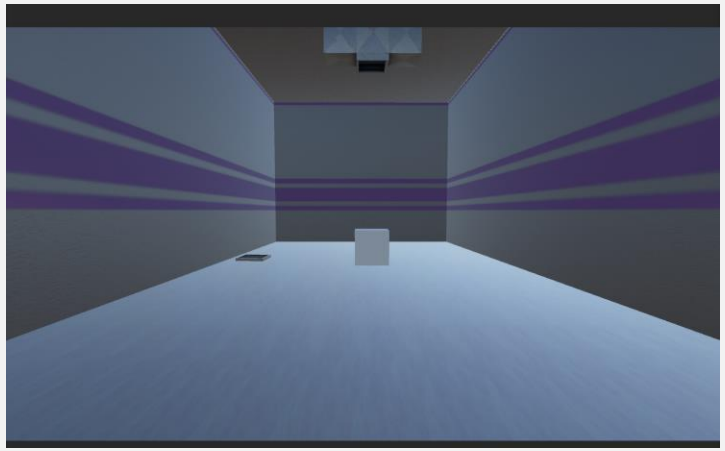


Figure 49.2 - Screenshot of ex-lover placeholder falling.


Figure 49.3 - Screenshot of text appearing in sequence.

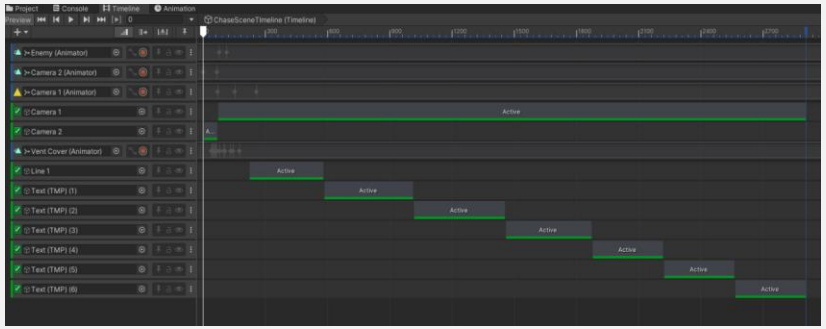


Figure 49.4 - Screenshot of current timeline.

# [Semester 2] Week 9: 3D modelling, UV Unwrapping and Texturing new doors

Implementing the feedback, I got on the doors being too simple, I recreated them (Figure 50.2) in Maya to look like the reference image I found of a prison door (Figure 50.1). In substance painter, I used a concrete like material (Figure 50.3) and "Metal Brushed" (Figure 22.2, p. 24) to imitate the look of heavy doors. This fence material I chose is to show the player that the prisoners can't just break the glass, which enhances the idea that the prison is controlled. I adjusted its channels mapping opacity setting to normal (Figure 50.5) in hopes to try to obscure the glass (Figure 50.4) so that the player can't peek through the glass especially during the chase scene.

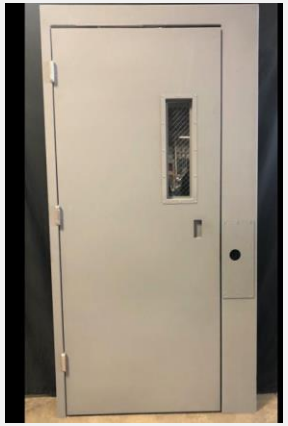


Figure 50.1 - New door reference.

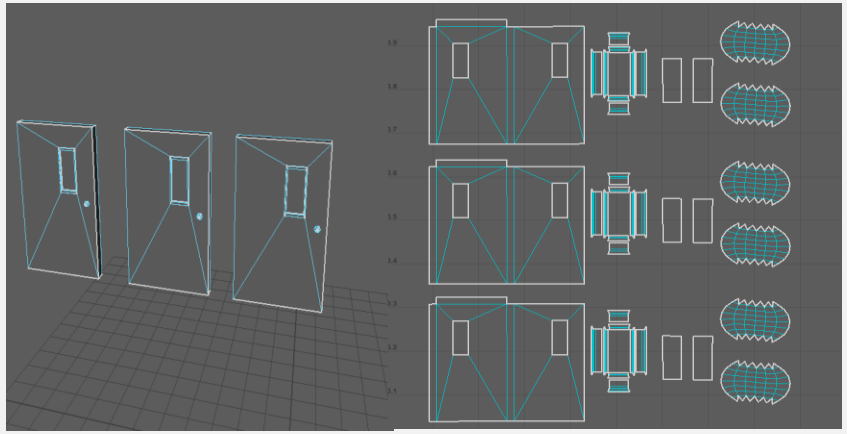


Figure 50.2 - New door 3D models and UV maps.

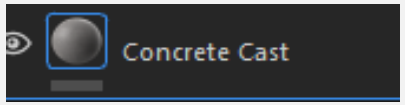



Figure 50.3 - Concrete cast Substance Painter material.

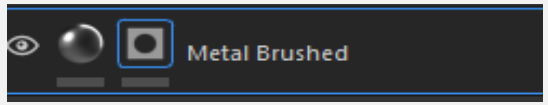



Figure 22.2, p. 24

Figure 50.4 - Changing opacity of window.

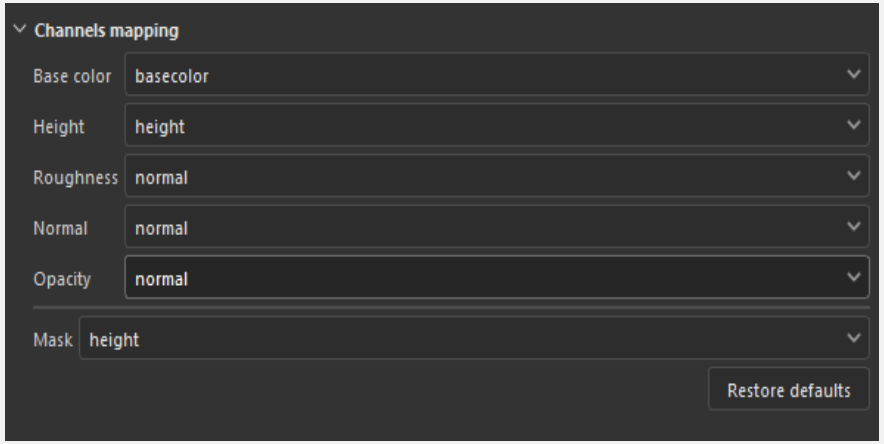
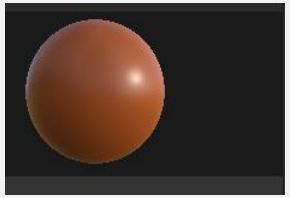



Figure 50.5 - Channels mapping iteration.

# [Semester 2] Week 9: Texturing new doors 2

I used orange (Figure 51.1) and grunge stains large hollow brush (Figure 24.6, p. 26) again to make the rusty surface to match the doors with the old retro/vintage abandoned prison environment. This also helps to keep the rust look consistent throughout the whole prison, same with the mold. I chose red, grey and pale green (Figure 51.2) as the door colour options. I chose red because it's the complimentary colour of green, I chose grey because I wanted a door with a rustic appearance just in case the red door was too bright for the environment and I chose a pale green door to enhance the grimy feel of the prison environment. I was able to adjust the opacity of the glass in Unity (Figure 51.3), this was a necessary step because these doors are used in the chase scene environment, if the player can see through the glass of the doors, they'll be able to tell which door is the wrong door, which is not what we want because that's what the lights-out animation is for.



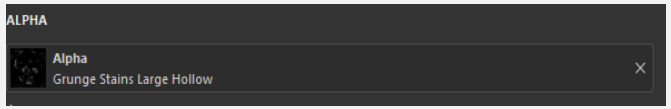



Figure 24.6, p. 26

Figure 51.1 - Orange for rust colour.

Figure 51.2 - Textured new doors.

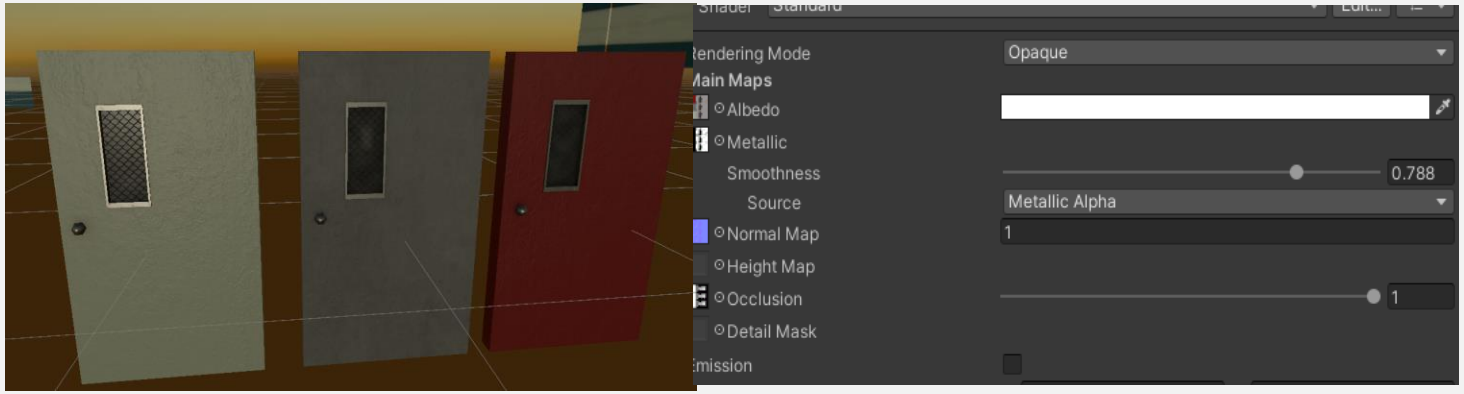



Figure 51.3 - Making the renderinf mode opaque so that you can't see through the windows.

# [Semester 2] Week 9: Adding the Jump scare animation into the Chase Scene

As a duo, we needed help with implementing the jump scare that my teammate created so we asked ChatGPT for help after getting help through our lecturers and researching, they just instructed to create a boolean and a SetActive(true/false) lines for the jump scare (Figure 52.1). We did come across an issue that the jump scare wouldn't play its animation but the image would pop up. It was a minor fix; After getting help from the teaching assistants, we just didn't put the animator on the correct game object. We added a black background to appear at the same time as the animation (Figure 52.2-52.4) because we wanted to make sure that the ex-lover was the focal point, without the black background, the player could still see the environment of the chase scene, which removes the focus of the jump scare animation.

This jump scare is related to "The Chime and Buzzer Principle" that Swink (2008) discusses in their research. *In Lover's Cell* the jump scare is a buzzer because it's negative feedback to the player signaling that they've gotten caught by the ex-lover.

```
public class Enemy : MonoBehaviour
{
    public NavMeshAgent enemy;
    public Transform player;
    public GameObject gameOver;
    public float killDistance = 1.5f;
    //public Animator exAnimator;
    public GameObject jumpscareUI;
    public GameObject jumpscareBG;
    public AudioClip jumpscareAudio;
    // Start is called before the first frame update
    private bool hasKilledPlayer = false;

    void Start()
    {

    }

    // Update is called once per frame

    void Update()
    {
        if (hasKilledPlayer) return;
        enemy.SetDestination(player.position);
        float distance = Vector3.Distance(transform.position, player.position);

        if(distance <= killDistance)
        {
            //exAnimator.SetTrigger("Jumpscare");
            KillPlayer();
        }
    }


    void KillPlayer()
    {
        hasKilledPlayer=true;
       //player.gameObject.SetActive(false);
       FirstPersonController controller = player.GetComponent<FirstPersonController>();
        controller.enabled = false;


        jumpscareUI.SetActive(true);
        jumpscareBG.SetActive(true);
        jumpscareSFX();
        //exAnimator.Play("jumpscareanimation", -1, 0f);
        StartCoroutine(ShowGameOverAfterJumpscare());



        // reset animation back to idle
    }


    IEnumerator ShowGameOverAfterJumpscare()
    {
        yield return new WaitForSeconds(1f);
        gameOver.SetActive(true);
        Cursor.lockState = CursorLockMode.None;
        Cursor.visible = true;


    }

    public void jumpscareSFX()
    {
        AudioManager.instance.PlayAudio(jumpscareAudio, 1f);
    }
}
```

Figure 52.1 - Updated Enemy Script.

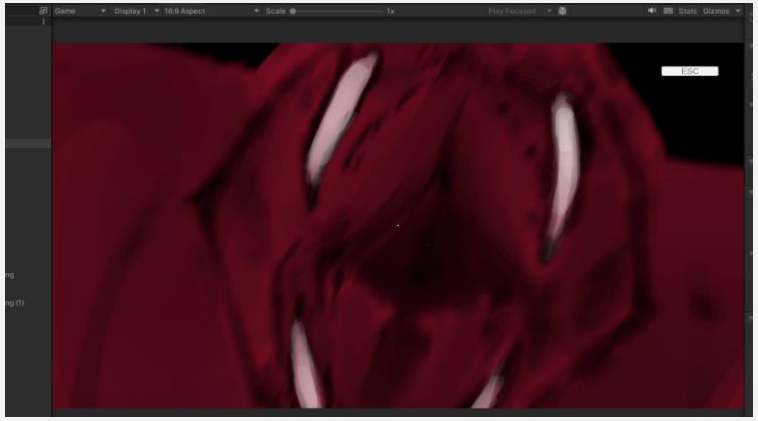



Figure 52.2 - Black background behind jumpscare.


Figure 52.3 - Updated OnClick() function for the game over canvas.

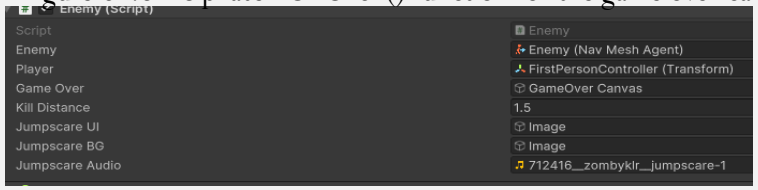



Figure 52.4 - Enemy Inspector set up.

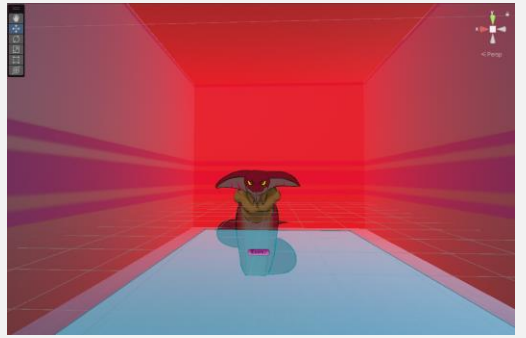
# [Semester 2] Week 9: Polishing the lighting in Chase Scene

I played around with the colour settings of the fog lighting within the chase scene. These iterations range from red, pink, green or black. I picked bright red to symbolise the urgency of needing to escape, I picked pink to symbolise love, I picked green to explore with the grimy theme and I picked black to make everything hard to see and appear more spooky. In the end, I ended up picking this dark red colour (Figure 53.1) because it ended up looking the best visually, and it being the same shade of red the ex-lover is, implies that the whole environment of the staff hallways is reflection of the control the ex-lover has within the prison.

Peach pink Lighting

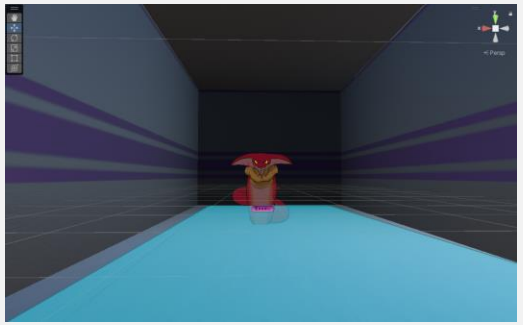
Bright pink Lighting

Neon pink Lighting



Bright red Lighting

Murky green Lighting



Black Lighting

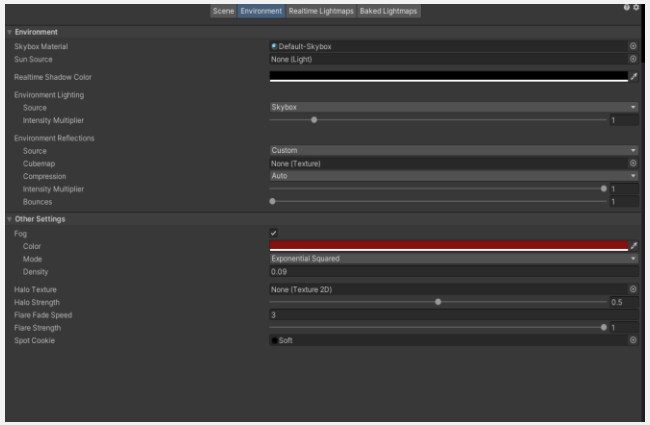
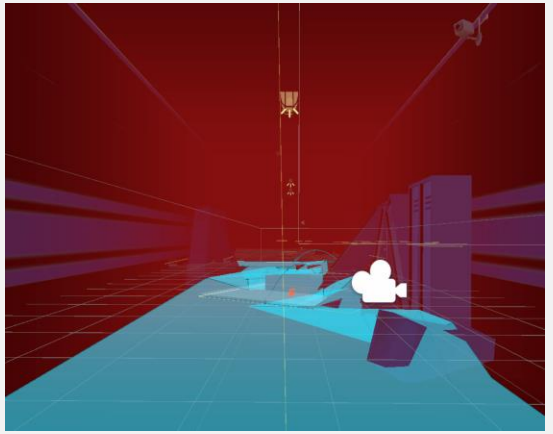



Figure 53.1 - Fog light set up.



Dark Red Lighting

# [Semester 2] Week 9: Polishing the lighting in Escape Room Scene

I experienced some trouble baking my lighting in the escape room scene (Figure 54.1), it made the environment go from the low lighting in an underground prison, to a radioactive take on the lighting. So, I decided to keep the lighting mode of all the lights in real time, taking the advice of my lecturer (Figure 54.3). To reduce the amount of lighting in the environment, I created a new light material (Figure 54.2) with high emission to create the glow of the light bulb. So instead of having three lights to one model, there's only one spotlight to imitate the light washing over the surface (Figure 54.5). Both light versions, express the liminal space feel in the abandoned prison but the new change will help with more efficient rendering and have better optimisation for our game to run. I also think that the lighting helps to highlight our horror predominant lens because of the fogginess mixed with the industrial lighting.

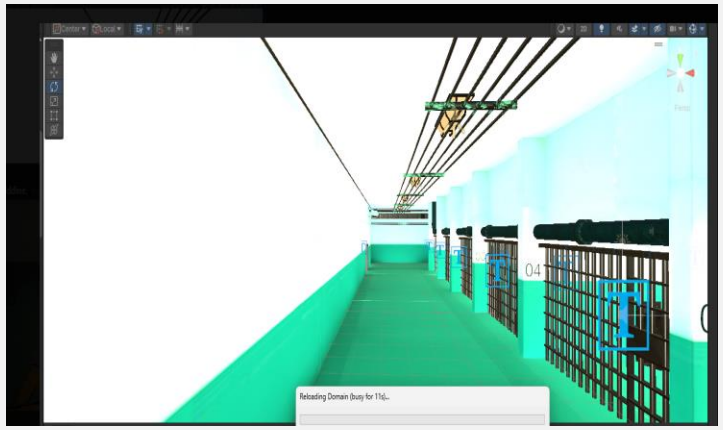


Figure 54.1 - Radioactive baked lighting.


Figure 54.2 - New emmisive light material.

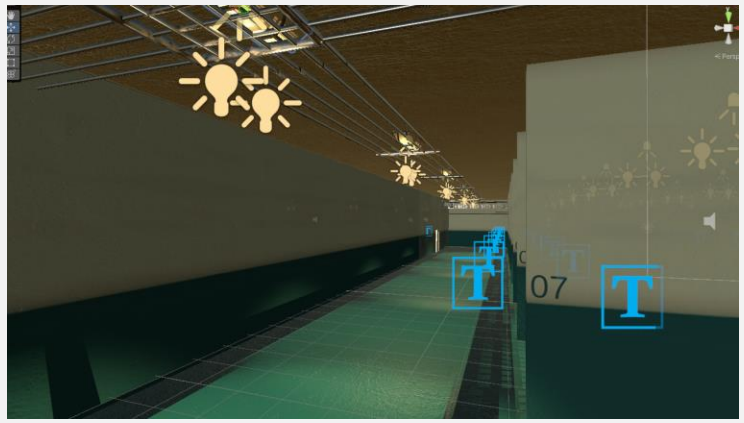


Figure 54.3 - Old lighting prison room set up.

Figure 54.4 - New lighting cafeteria set up.

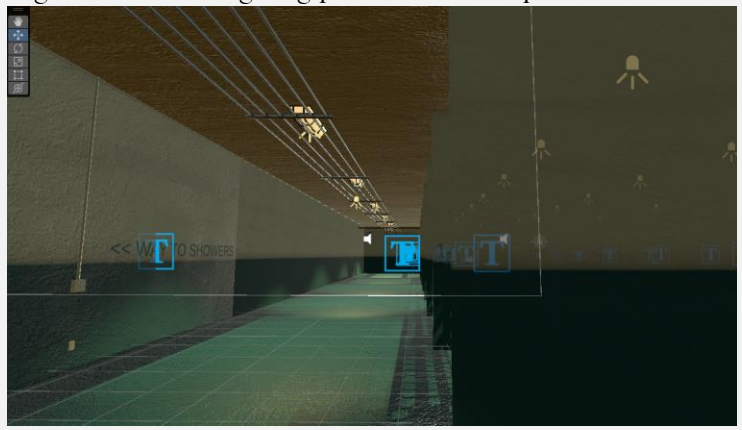


Figure 54.5 - New lighting prison room set up.

# [Semester 2] Week 9: Lights Out Animation and Polish

After adding the lights back in, I decided to animate the lights to go out (Figure 55.1-55.3) when the player goes towards the wrong door, because having the right doors be the colour red in contrast to the grey doors isn't enough indication to the player that they're going the right way because the fog is thick. I took this time to update the lighting in the chase cut scene to match the lighting of the actual chase scene (Figure 55.4) so that the transition from the cut scene to the chase scene is consistent when the player gets teleported. I think lighting is another good way of enhancing the game feel of the game. I wanted to relate it to Swink's (2008) study how game feel doesn't have a definite meaning but can be perceived in different ways.

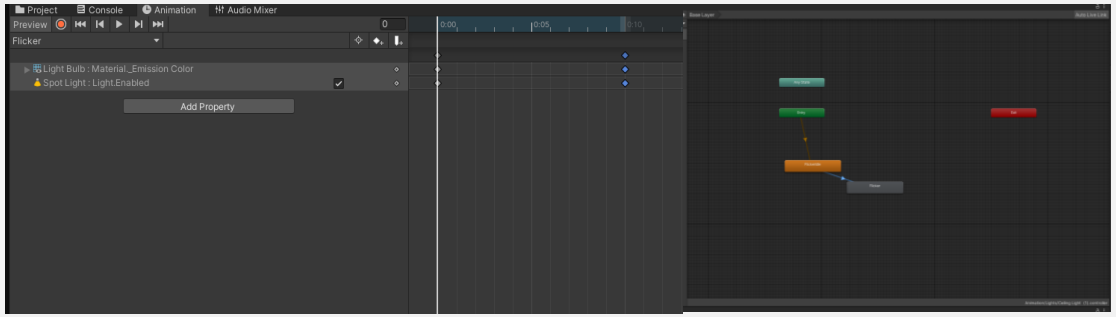



Figure 55.1 - Light bulb animation and animator.

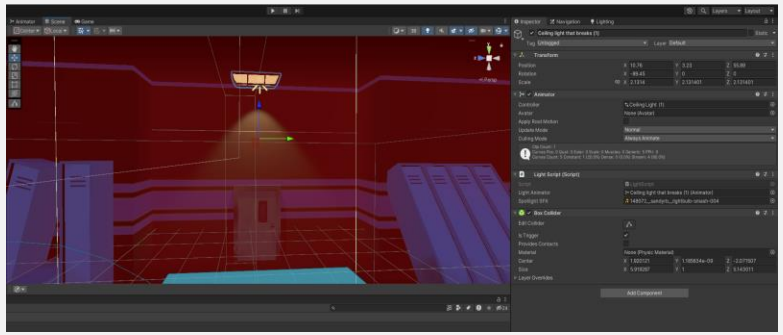


Figure 55.2 - Light box collider set up.

Figure 55.3 - Screenshots of light turning off game play.


Figure 55.4 - Final chase cut scene lighting.

# [Semester 2] Week 9: Voice Acting

As a duo, we voice acted the lines of the NPCs in our game. We did this to give the NPCs more personality and emotion to them which enhances the narrative through the dialogue and helps the player remember who they're talking to because our escape room requires you to socialise with the NPCs to get clues on the code to escape. It also nullifies the horror we portray in the game, making it lean more towards dark comedy because of the goofy voice lines. It's also good for the player to hear the voices of the NPCs because there's so many of them in the prison environment, it'll help the player to navigate through each of the areas. Since we're both not professional voice actors, we used Premiere Pro to cut out any bloopers or mistakes we made when reciting our lines (Figure 56.1) and we needed to change the format to MP3 because Unity does not support M4A files (Figure 56.2).

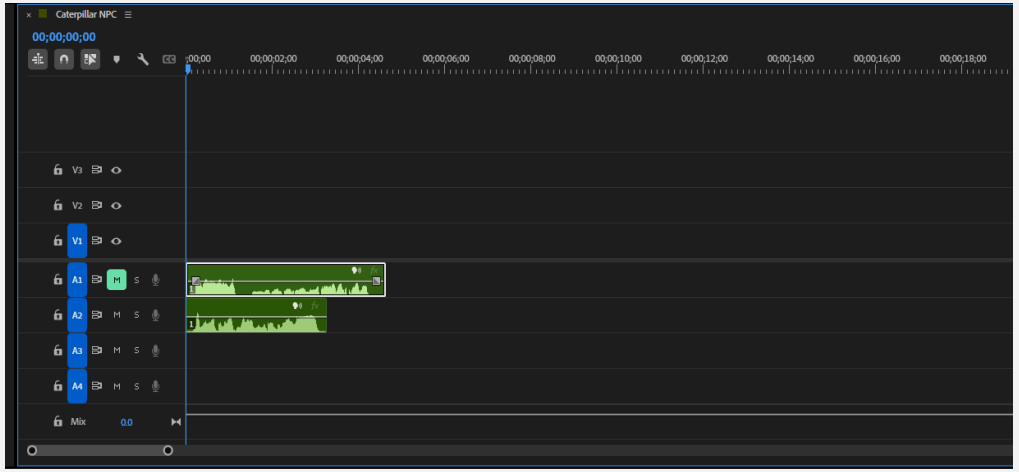



Figure 56.1 - Screenshot of editing the voice lines in Premiere pro.

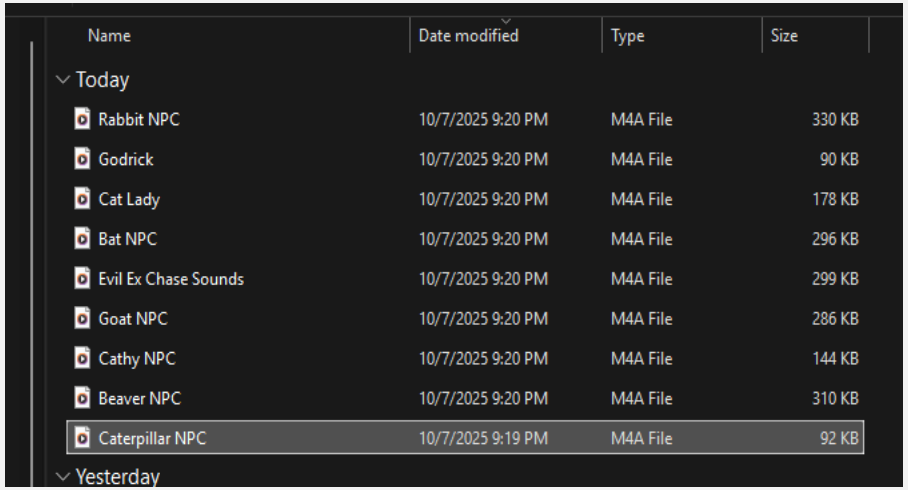



Figure 56.2 - List of voice lines I edited.

# [Semester 2] Week 9: Game Over and Win Screen Final

Here is a showcase of the final game over and win screens. For the win screen, I added extra plants to the environment (Figure 57.1), darkened the shadows and emphasised the lighting to symbolise that the light is flooding away the darkness of living in a prison and the control from their ex-lover. For the game over screen, I darkened the shadows (Figure 57.2) to emphasise the loss and the feeling of impending doom Y/N feels after being caught by their ex-lover, who is seen dragging them back into a cell. This symbolises the ex-lover consolidating their power and control over Y/N. The spotlight is there which mocks sun light, and them being dragged away from it further enhances their loss. Then we changed the text in Unity to match with our dialogue font on our Title screen. I also decided to keep Y/N's faceless so that my drawings of them match with my teammate's drawings to keep a consistent design of Y/N.


Figure 57.1 - Before and after final win screen with text.


Figure 57.2 - Before and after final game over screen with text.

# [Semester 2] Week 10: Updating text on mesh and Billboarding Ex

Some minor polishes I made were changes to the text in the escape room environment was finally adding "Black Ops One" (Figure 58.2) for better readability and because it's stiff-vintage/retro feel matches with our prison environment's chosen aesthetic. To billboard the ex-lover in the chase scene, so that the player can always see the enemy, I created a new script after looking through a Unity discussion page ("how I can create an sprite that always look at the camera?", 2011) on billboarding.

It was just one line of code (Figure 58.3) that needed to be implemented to the enemy game object. I had to adjust the line by removing "-Vector3.up" (Figure 58.4) to prevent the game object from flipping upside down. Billboarding the ex-lover was necessary so that their 2D sprite wouldn't disappear when the player turns a corner, without billboarding, the enemy would have an unfair advantage on the player because there's some areas where the player can't see them cause they're not facing the camera. It adds to the narrative that the ex-lover is always watching Y/N because of this change.

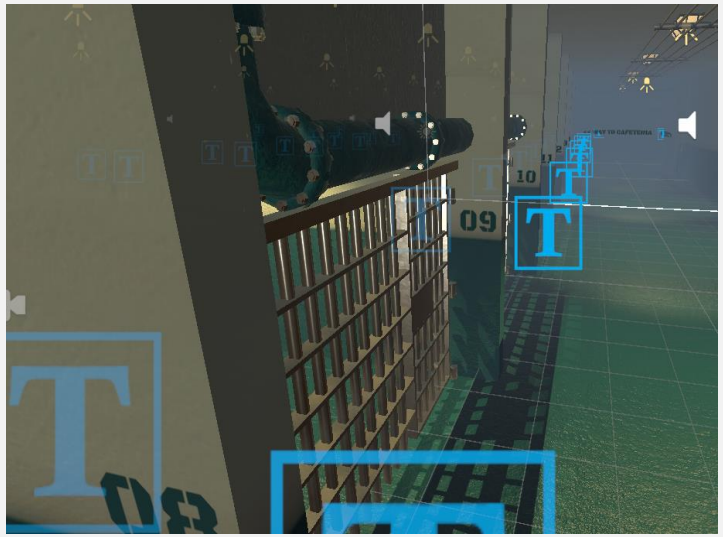


Figure 58.1 - Updated prison cell text on mesh.

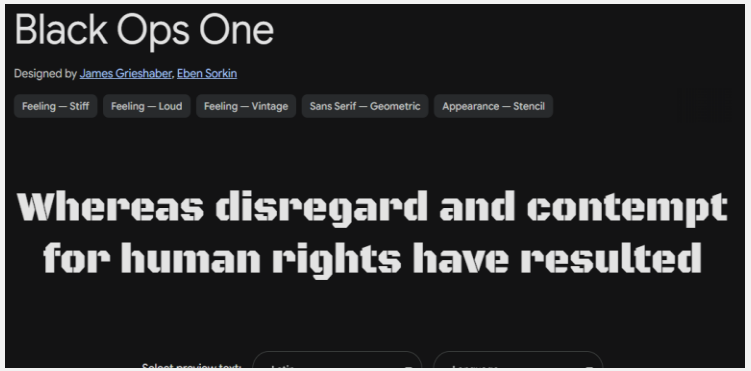



Figure 58.2 - Screenshot of Black Ops One font example.

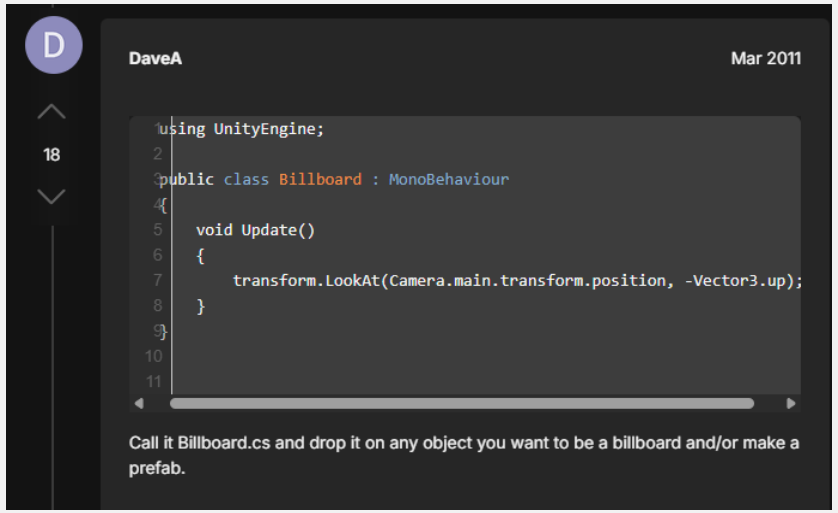



Figure 58.3 - Screenshot of Unity Discussion post on Billboarding.

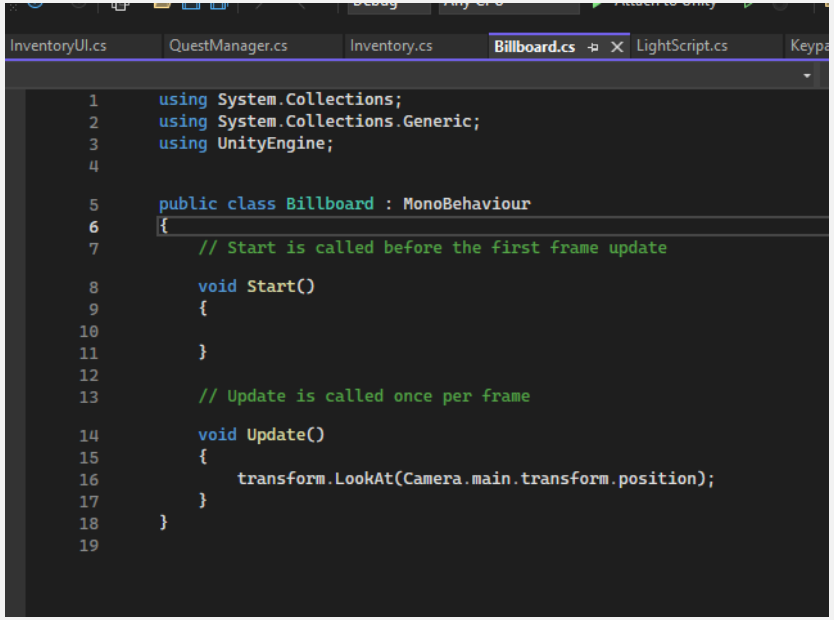



Figure 58.4 - Updated Billboard script.

# [Semester 2] Week 10: Ability to have text pop up of what's written on the letter

This was another minor polish that my teammate wanted to implement into our game so that the lover letters have better readability. This minor change relates to my capstone proposal because it helps make the environmental interaction with the love letter more efficient by translating the poor lipstick handwriting on the love letter sprite. Coding isn't my focus but with my basic knowledge, I was able to help her by adjusting the lovenote.cs code (Figure 59.1), to have a function to turn on and off the read note game object. Then we just adjusted the OnClick on the read note button to help the text panel appear (Figure 59.2-59.3).

```
0 references
public void CloseNote()
{
    playerMove.enabled = true;
    Cursor.lockState = CursorLockMode.Locked;
    Cursor.visible = false;
    loveNote.SetActive(false);
    readNoteButton.SetActive(false);

}

1 reference
private void OpenNote()
{
    loveNote.SetActive(true);
    Cursor.lockState = CursorLockMode.None;
    Cursor.visible = true;
    playerMove.enabled = false;
    interactionPrompt.SetActive(false);
    readNoteButton.SetActive(true);
}
```

Figure 59.1 - Love Note Script updated.

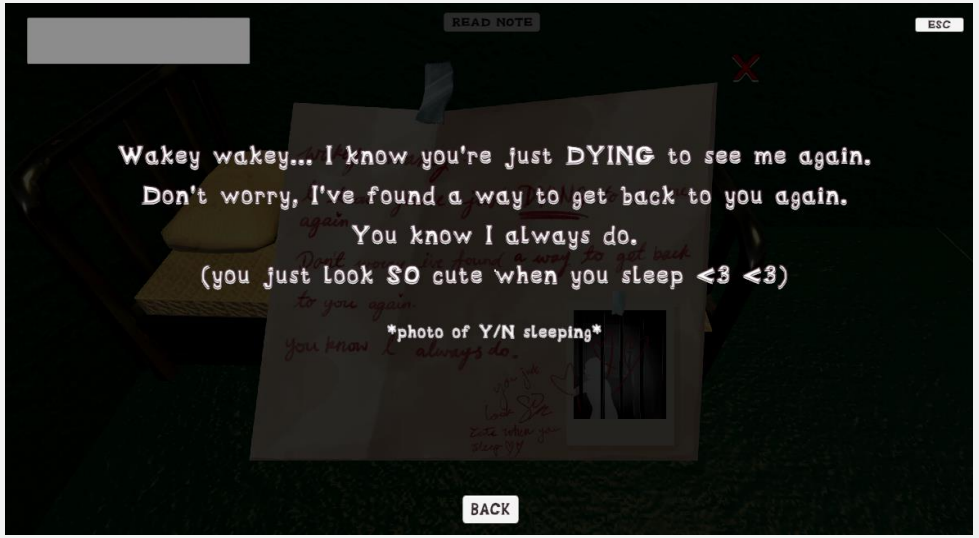



Figure 59.2 - Added translation panel for love notes.

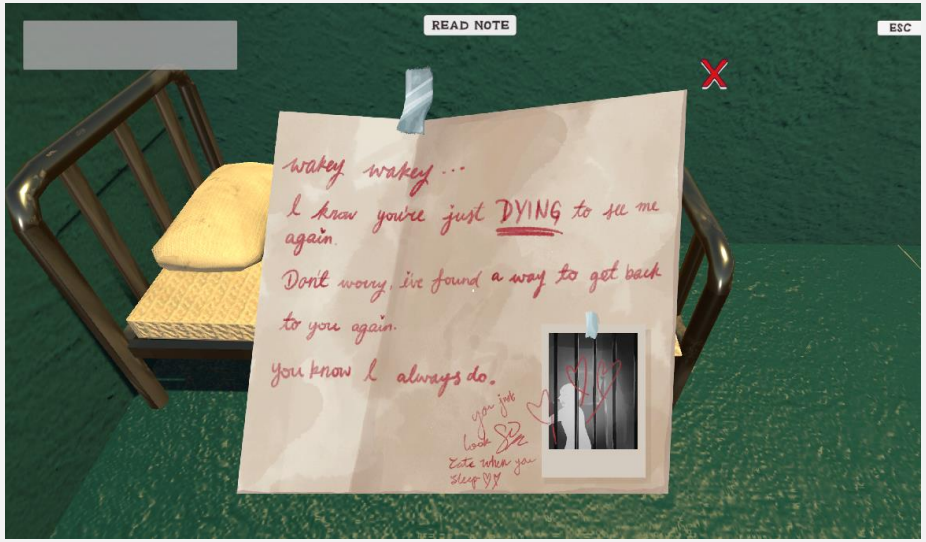



Figure 59.3 - Screenshot example of the read note button.

# [Semester 2] Week 10: Adding Inventory UI Pop up 1

After many complaints from the last playtesting, that the objects were too hard to find, I implemented this basic inventory UI which tells the player which object they're holding (Figure 60.1). On Procreate, I quickly sketched the images representing each item that the player can pick up (Figure 60.2). I used this Minecraft book image as reference for the cultist book design (Figure 60.3). My idea was that when the player picks up or receives one of these objects from the NPCs, the images representing each item will appear on the slots of the inventory UI and will disappear when they don't have it anymore. I did this for every item except the janitor's keycard, because upon picking up the janitor's keycard, it automatically opens the janitor's closet and destroys itself.

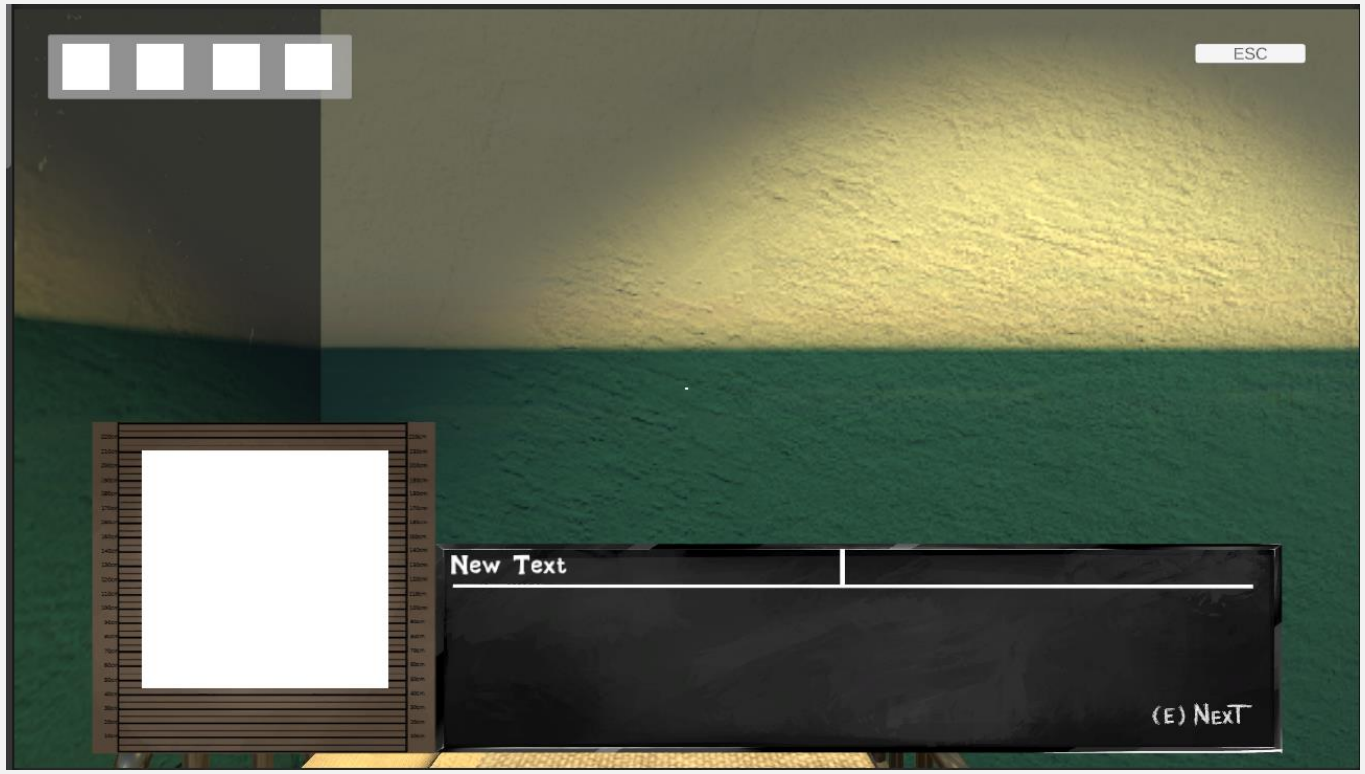



Figure 60.1 - Inventory panel UI set up.

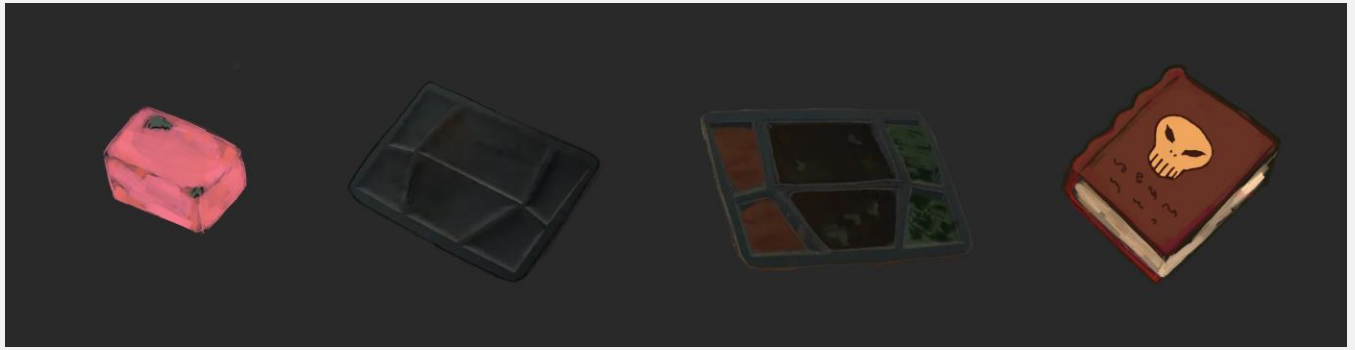


Figure 60.2 - Final item icon sketches made in Procreate.

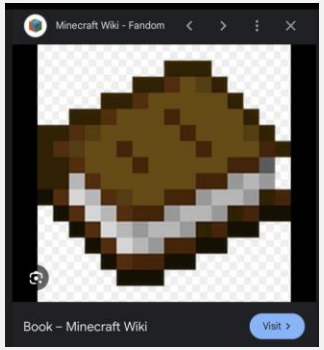
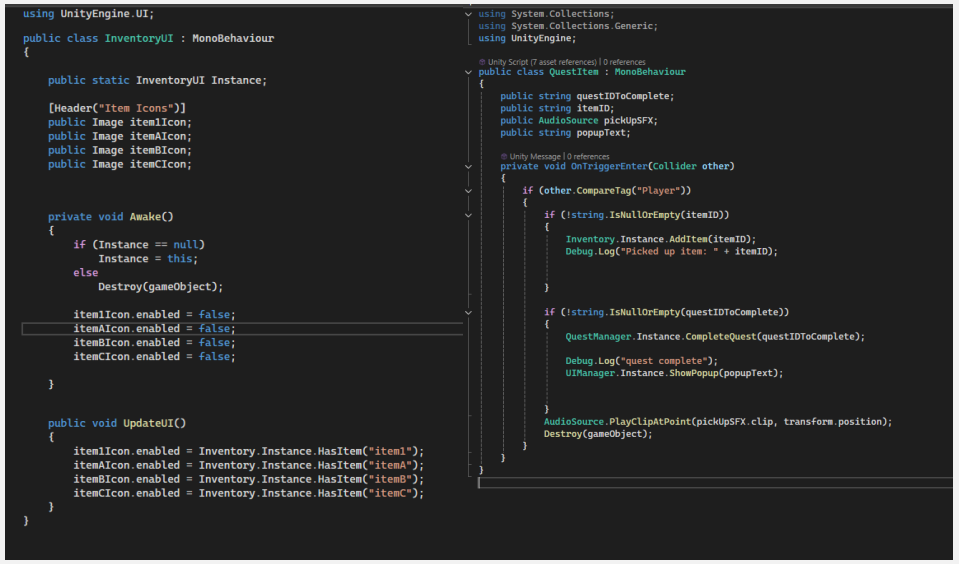



Figure 60.3 - Minecraft book reference.

# [Semester 2] Week 10: Adding Inventory UI Pop up 2

Since the scripts relating to this mechanic belonged to my teammate, I chose to use ChatGPT to help me make this function work as a last resort and to help me understand what she's written in her scripts (Figure 61.1). It instructed me to create a new script called InventoryUI.cs and to update my teammate's inventory script to reference the InventoryUI script. From what I learned, I'm enabling image game objects (Figure 61.2) and the right image is called based off their Item ID name.

Doing this should hint at the player interact with their environment more, knowing that there's a reminder of what they're holding. As a duo, we didn't want to lead the players directly out of the escape room, we want to let them figure it out themselves through NPC interactions and finding the objects in the environment.



```
using UnityEngine.UI;

public class InventoryUI : MonoBehaviour
{

    public static InventoryUI Instance;

    [Header("Item Icons")]
    public Image item1Icon;
    public Image itemAIcon;
    public Image itemBIcon;
    public Image itemCIcon;



    private void Awake()
    {
        if (Instance == null)
            Instance = this;
        else
            Destroy(gameObject);

        item1Icon.enabled = false;
        itemAIcon.enabled = false;
        itemBIcon.enabled = false;
        itemCIcon.enabled = false;

    }


    public void UpdateUI()
    {
        item1Icon.enabled = Inventory.Instance.HasItem("item1");
        itemAIcon.enabled = Inventory.Instance.HasItem("itemA");
        itemBIcon.enabled = Inventory.Instance.HasItem("itemB");
        itemCIcon.enabled = Inventory.Instance.HasItem("itemC");
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

Unity Script (7 asset references) | 0 references
public class QuestItem : MonoBehaviour
{
    public string questIDToComplete;
    public string itemID;
    public AudioSource pickUpSFX;
    public string popupText;

    Unity Message | 0 references
    private void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("Player"))
        {
            if (!string.IsNullOrEmpty(itemID))
            {
                Inventory.Instance.AddItem(itemID);
                Debug.Log("Picked up item: " + itemID);


            }

            if (!string.IsNullOrEmpty(questIDToComplete))
            {
                QuestManager.Instance.CompleteQuest(questIDToComplete);

                Debug.Log("quest complete");
                UIManager.Instance.ShowPopup(popupText);


            }
            AudioSource.PlayClipAtPoint(pickUpSFX.clip, transform.position);
            Destroy(gameObject);
        }
    }
}
```

Figure 61.1 - InventoryUI Script screenshot.


Figure 61.2 - Screenshot of the Inventory UI popping up in game.

# [Semester 2] Week 11: Animate lockers in chase scene and Camera Clutter Level Design Polish

I chose to follow the feedback I got from our second playtest, the comment on animating the lockers to fall in front of the player, so that it'd add more life to the liminal aesthetic of the chase scene. Overall, making the obstacle course gameplay more intense and interesting. I made the starting position of each locker to look like it's about to fall, to indicate to the player that they need to get out of the way (Figure 62.1).

Then I combined a Metal creaking and closing sound effect in Premiere pro together (Figure 62.2), the metal creaking is to warn the player that the locker is about to fall and the metal closing sound effect imitates the sound of metal slamming on the ground. Afterwards, I connected a big box collider with trigger checked, to give time for the animation to play and land just in front of the player (Figure 62.3). On addition to polish the chase scene, I removed the cameras from the walls (Figure 62.4) because they can't be seen anyway and instead scattered them on the floor, to show the destructive nature of the controlling ex-lover in the environment, the cameras were a symbol that the security had the prison under control, but them being tossed on the floor amongst the destruction of the staff hallways shows that the ex-lover is the one truly in control of the prison now. Within the escape room scene, to support my investigation, I made the cameras look broken in the rooms that the ex-lover tampered with like the camera at the entrance of the staff hallways. It implies that the ex-lover is leading Y/N out of the prison so that they can meet.

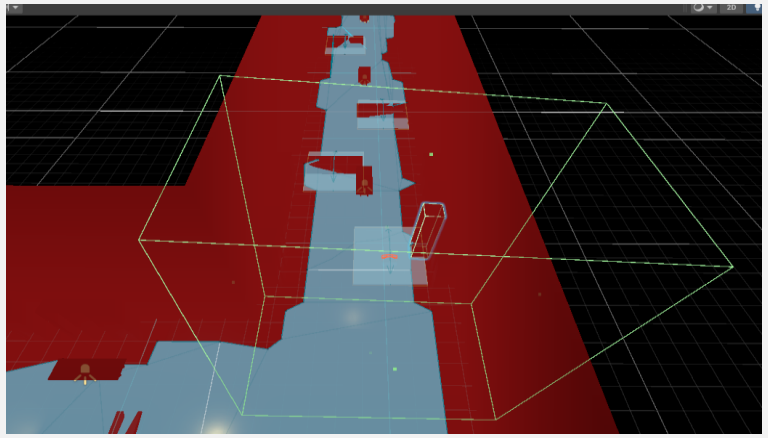


Figure 62.1 - Locker trigger example.

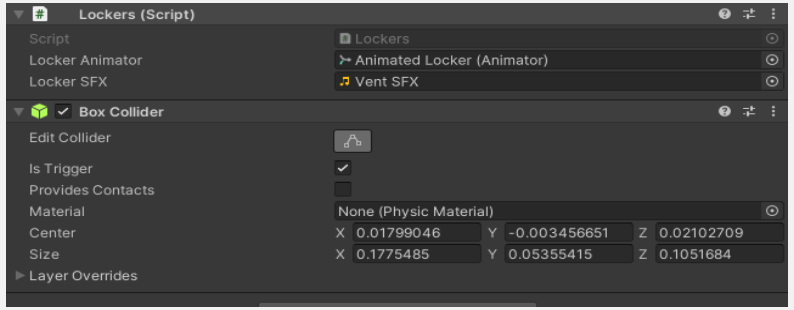



Figure 62.3 - Locker inspector set up.

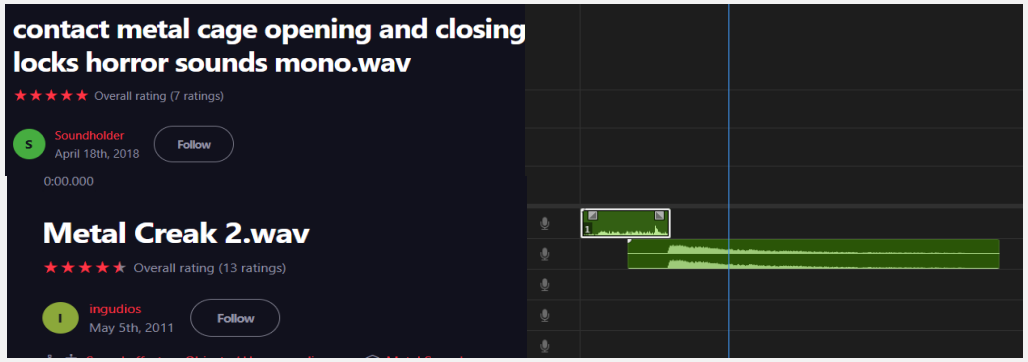



Figure 62.2 - Screenshot of coming the locker audio in Premiere pro.

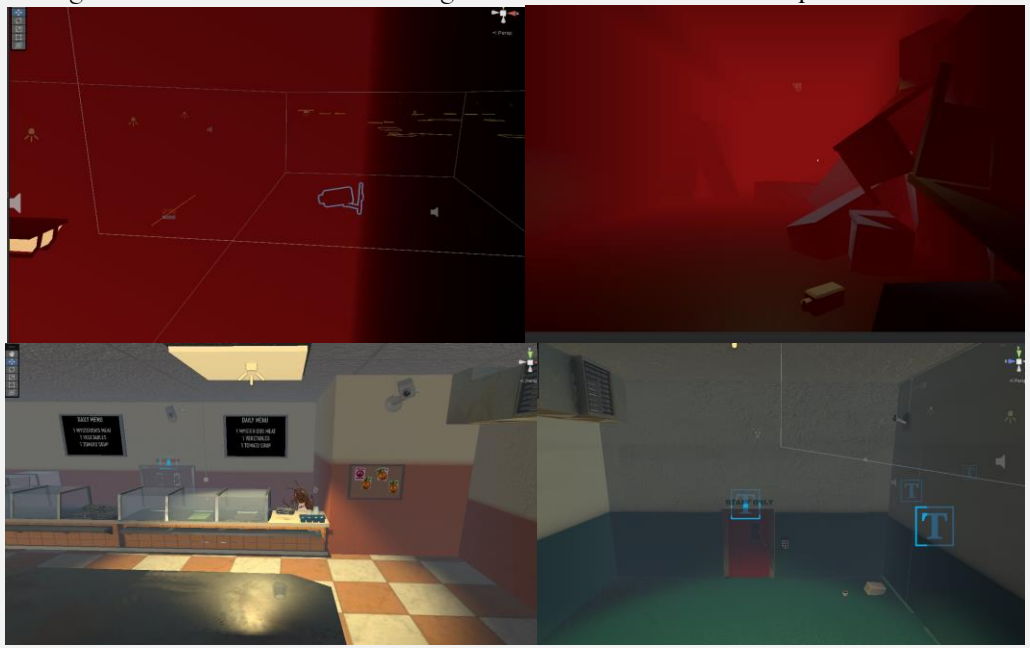


Figure 62.4 - Screenshots of cameras throughout the environment.

# [Semester 2] Week 12: Creating Backgrounds for the animated scenes in our trailer

I was tasked with creating the backgrounds for the animated scenes in our trailer. Our trailer will showcase an animated scene of Y/N waking up, finding a love note from their ex-lover and the ex-lover slithering through the vents.

Using our Miro storyboard as reference, I screenshotted (Figure 63.1) a top angle of a bed in a random prison cell and the inside of the vents. Then I took these screenshots and using the method I did for the game over screen I traced the line art (Figure 63.2) for the bed and coloured with the same brushes to keep the grimy feel cohesive with the other illustrations I've made for the game. The idea was that we wanted the trailer to give hints to the audience that somebody is watching them. I also didn't want to re-render the screenshots completely; I just wanted to outline the 3D environment so that you can still see that our game is 2.5D because there's a contrast with Y/N being 2D in the 3D background.

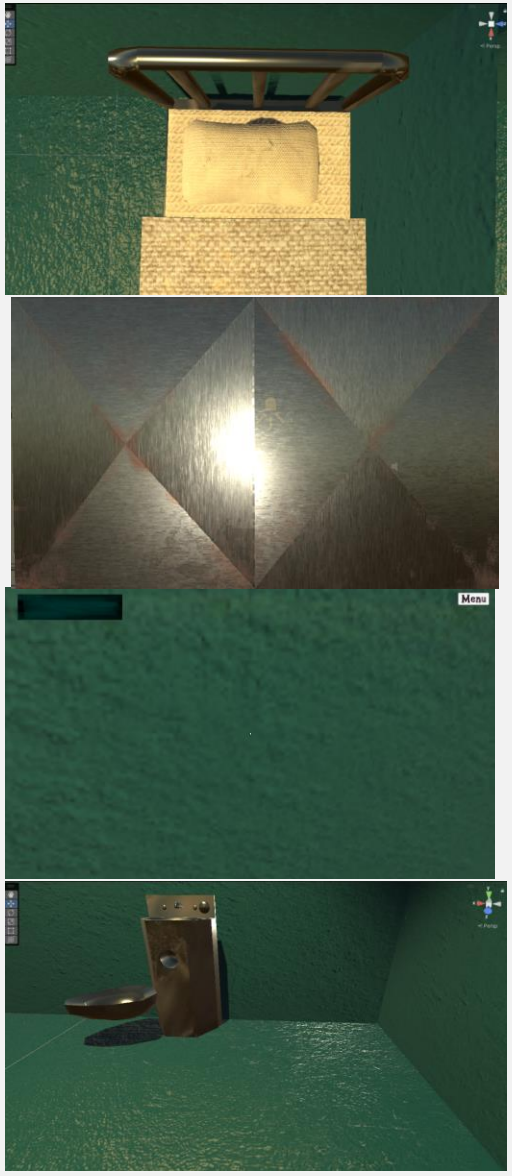



Figure 63.1 - Screenshots from our game.

Figure 63.2 - Edited screenshots from our game.

# [Semester 2] Week 12: Polishing Cafeteria

I realised that I had forgotten to make a drink dispenser in the cafeteria, these small details matter because they're what weave the retro/vintage aesthetic together. With my chosen reference image (Figure 64.1) I quickly 3D modelled a low-poly drink dispenser in Maya (Figure 64.2).

When I imported the 3D model to Substance Painter, I wanted to highlight the force of being healthy in this prison so I designed the drink icons to be fruits and vegetables (Figure 64.3). To reflect more control over the environment, I made a "Temporarily unavailable" sign on the clean water (Figure 64.4) to symbolise that this prison doesn't have access to clean filtered water anymore which adds to the mistreatment of the prisoners.

Within the cafeteria, I chose to add two drink dispensers to cover the area of the counter, I added cups to make the table feel less empty and so that it would be cohesive with the other environmental assets in the cafeteria (Figure 64.5).

Figure 64.1 - Drink dispenser.

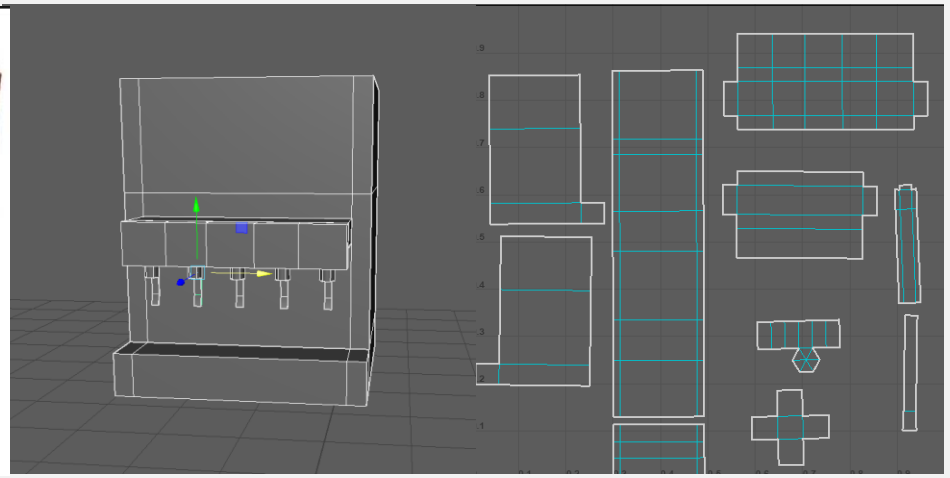


Figure 64.2 - 3D models and UV map of drink dispenser.

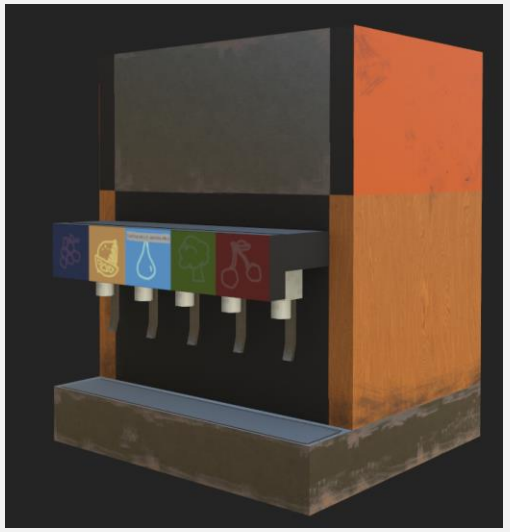


Figure 64.3 - Textured drink dispenser.

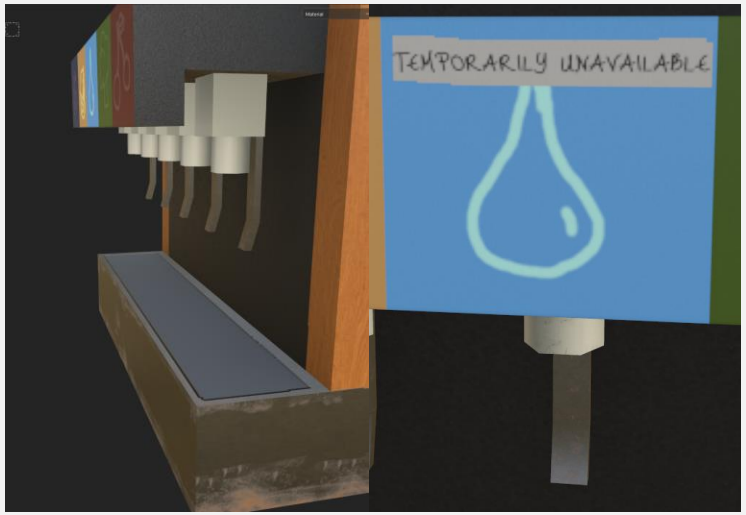



Figure 64.4 - Adding details to the drink dispenser.

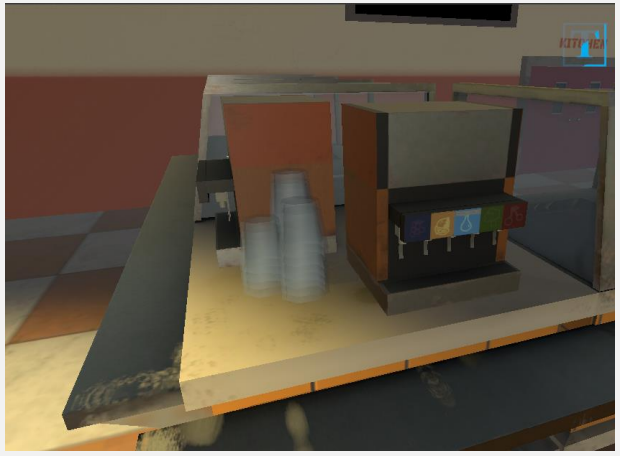


Figure 64.5 - Importing drink dispenser into the cafeteria.

# [Semester 2] Week 12: "Bugs" (More like precautions.)

The only bug that has to do with environment interactions in our game is when the player enters the trigger of the keypad and they press cancel on the keypad. The player gets "stuck", all you have to do is just press enter to center the cursor again (Figure 65.1) and it makes the player move again.

One more general bug is the audio, when the player mutes the game in the escape room and teleports to the next scene, it keeps the music muted the entire chase scene. The voice lines are coded a different way, I didn't code them, so they couldn't be adjustable in the volume sliders (Figure 65.2). Another bug is that the lost tray instantiates through the floor but it still functions properly. The pivot point was set normal for this tray, so it's a mystery on why it's spawning like that.

Although these "bugs" are present in the final, they don't affect the gameplay, it's still playable.


Figure 65.1 - Screenshot of where the player can get stuck in game.

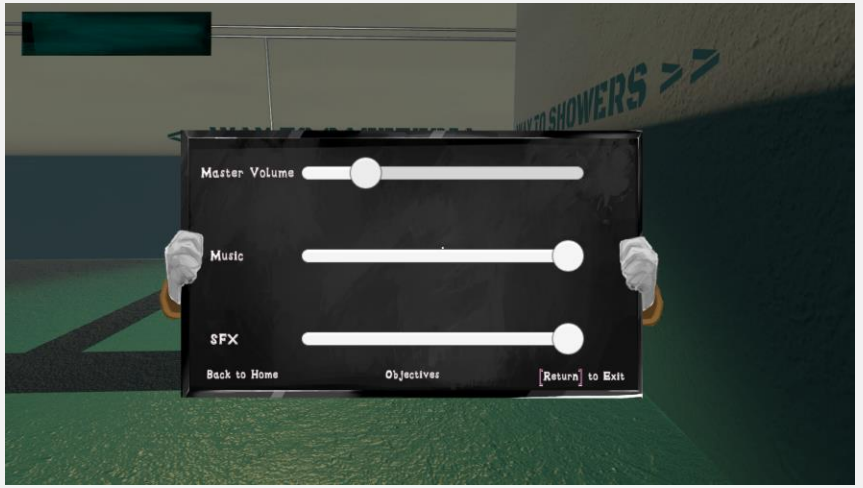



Figure 65.2 - Screenshot of volume slider at low volume.


Figure 65.2 - Tray intersecting with floor.

# Reflection

This assignment has challenged me a lot as an environment artist and game designer this year. From losing a teammate, we had to downscale our project by a lot. Which came with its strengths because we were challenged to create a new narrative within a prison 2.5 escape room game. This year has also taught me how to effectively practice a proper 3D modelling pipeline, utilising Substance Painter, Maya and Unity. I also got to shape up my art skills through adding details in Substance Painter. Although coding wasn't my focus, I was able to understand the terminology through sufficient research, tutorials and the occasional help from ChatGPT. I was able to fulfill my goal of using Maya as much as I could before it becomes inaccessible to me next year. There's so much room for improvement but I'm happy with the progress I've made so far especially with UV unwrapping and code which were my biggest weaknesses last year.

Accomplishments: Despite losing a team mate, as a duo I was able to manage making a 3D environment that reflected the experience of control through the assets, handling tasks while having to code occasionally, and being able to create two environments with similar aesthetics that relate to my capstone proposal and one environment for the cutscene. It was also our first time creating a cutscene so I'm proud of us for being able to make one so dramatic. I was happy with how I was able to implement footsteps changing on the floor of the showers room, it was good sound feedback. I also think the optimisation for the models are okay but they could be better especially the lights.

Improvements: There are still so much polishes I could have done if we didn't lose time due to the loss of a teammate, like more environment transitions and animations for the chase scene to have more level progression. More quests for the other NPCs would have been a nice add for more environmental interactions with the items. I think I could've improved more of the clutter in the showers room, but the clutter now is just enough.

If I were to do Capstone again: I would like to learn more about baking lights in Unity for better performance, it wasn't an issue to use real-time lighting for now because it gave the vibe we wanted but It still would be better if the lights in the escape room were baked because they are static objects unlike the lights in the chase scene.

# Bibliography


AKAR NZ. (2025, September 23). *About - EcoSecurity Utensil*. EcoSecurity Utensil. https://www.ecosecurityutensil.co.nz/

Angelina Cinema. (2025, May 24). *Y/N COMPIL (weeweepiss tt)* [Video]. YouTube. https://www.youtube.com/watch?v=a2PwCCfbbZc

Auckland University of Technology. (n.d.). *Game design Week 2 Overview.* Auckland University of Technology Canvas. https://canvas.aut.ac.nz/courses/16552/pages/game-design-week-2-overview-2?module_item_id=874306

Brain Fail Productions. (2021, October 30). *Easy Snap* [Unity Asset]. Unity Asset Store. https://assetstore.unity.com/packages/tools/utilities/easy-snap-113010

Burns, S. D. (2023, July 5). *Beyond Extent | Trimsheets*. Www.beyondextent.com. https://www.beyondextent.com/deep-dives/trimsheets

Clip Collection Vault. (2022, February 13). *Unity - Open Door, with Safe Code*. YouTube. https://www.youtube.com/watch?v=dVUIg8A71RE

Code Master. (2022, June 27). 🔴 *Vertex Snapping • Snapping • Unity Game Development for Beginners • Unity Fundamentals • (Pt. 114)* . [Video]. YouTube. https://www.youtube.com/watch?v=KMj7tYnV9Hk

# Bibliography


Code Monkey. (2021, June 15). *How to use Unity NavMesh Pathfinding! (Unity Tutorial)* [Video]. YouTube. https://www.youtube.com/watch?v=atCOd4o7tG4&t=553s

Etredal. (2021, August 15). *Volumetric Fog in Unity using Particles (Any Rendering Pipeline)* [Video]. YouTube. https://www.youtube.com/watch?v=UllkvfMR96s

Fingerprints Icon. (2025). *Fingerprints Icon, Transparent Fingerprints.PNG Images & Vector - FreeIconsPNG*. Freeiconspng.com. https://www.freeiconspng.com/images/fingerprints-icons#google_vignette

Forrest, M. (2024, October 14). *Blender vs Maya: What is the best student tool in 2025? - ACC*. Access Creative College. https://www.accesscreative.ac.uk/blog/blender-vs-maya-what-is-best-2025/

Freepik. (2025). *Mugshot Images - Free Download on Freepik*. Freepik. https://www.freepik.com/free-photos-vectors/mugshot

HubSpot. (2025). *Unified Templates Resource Library*. Hubspot.com. https://www.hubspot.com/business-templates/daily-schedule

Kozmobot Games. (2023, January 17). *Making a 3D platformer in Unity - Player follow tutorial* [Video]. YouTube. https://www.youtube.com/watch?v=6Ai0xg6xTUk

# Bibliography


Kozmobot Games. (2023, November 22). *How to OPEN A DOOR With KEY in UNITY - Easy Tutorial*. YouTube. https://www.youtube.com/watch?v=RQ61nY2WOtA

LordEvilM44. (2022, October 26). *Add Rain to your Unity Games! | Unity Tutorial* [Video]. YouTube. https://www.youtube.com/watch?v=MBVGUD5nZeA

LSPLASH. (2021). *DOORS* [Video Game]. Roblox. https://www.roblox.com/games/6516141723/DOORS

Macklin, C., & Sharp, J. (2016). *Games, Design and Play: A Detailed Approach to Iterative Game Design*. Addison-Wesley.

McGarry & Madsen Inspection. (2018, June 18). *What is the difference between green and white sewer drain pipes?* [Blog post]. HowToLookAtAHouse. https://www.howtolookatahouse.com/Blog/Entries/2018/6/what-is-the-difference-between-green-and-white-sewer-drain-pipes.html#:~:text=The%20green%20sewer%20drain%20pipe,three%20times%20the%20green%20PVC.

Mob Entertainment. (2022). *Poppy Playtime - Chapter 2* [Video Game]. (https://store.steampowered.com/app/1817490/Poppy_Playtime__Chapter_2/

Mob Entertainment. (2024). *Poppy Playtime – Chapter 3.* [Video Game]. https://store.steampowered.com/app/2555190/Poppy_Playtime__Chapter_3/

# Bibliography


Mohammad Faizan Khan. (2022, April 2). *Unity3d How to Change Pivot | Set Pivot in Unity with offset using a empty parent object*. YouTube. https://www.youtube.com/watch?v=cnl6r68y-IE

myDev Learning. (2023, January 29). *Unity3D – Text as object in the scene (Text Mesh) | Tutorial* [Video]. YouTube. https://www.youtube.com/watch?v=SoMv-pQH2Mc

Omogonix. (2022a, June 3). *How to Add Footsteps Sounds in Unity* [Video]. YouTube. https://www.youtube.com/watch?v=uNYF1gsvD1A

Omogonix. (2022b, June 6). *How to Change Footsteps Sounds in Unity (Land to Water)* [Video]. YouTube. https://www.youtube.com/watch?v=IXy8bl6ujV4

Paulino, P. H. (n.d.). *6 Tips to Improve Your UV Mapping Workflow — Pauhpaulino*. Paul H. Paulino. https://www.paulhpaulino.com/6-tips-to-improve-your-uv-mapping-workflow

PlatinumFalls. (2022). BARRY'S PRISON RUN! (OBBY) [Video Game]. Roblox. https://www.roblox.com/games/8712817601/BARRYS-PRISON-RUN-OBBY

Rawlinson, K. (2024, May 22). Inmates dug through Winchester prison walls with plastic cutlery, report finds. *The Guardian*. https://www.theguardian.com/society/article/2024/may/22/inmates-dug-through-winchester-prison-walls-with-plastic-cutlery-report-finds

# Bibliography


Saporito, N. (2022, March 9). *How To Apply A Paper Texture In Procreate*. Logos by Nick. https://logosbynick.com/paper-texture-in-procreate/

Steel Wool Studios & ScottGames. (2021). Five Nights at Freddy's: Security Breach [Video Game]. http://store.steampowered.com/app/747660/Five_Nights_at_Freddys_Security_Breach/

Sinclair, J.-L. (2020). *Principles of Game Audio and Sound Design: Sound Design and Audio Implementation for Interactive and Immersive Media.* CRC Press.

Swink S. (2008). *Game Feel: A Game Designer's Guide to Virtual Sensation.* Morgan Kaufmann.

Unity Technologies. (2011, March 17). *how I can create an sprite that always look at the camera?* Unity Discussions. https://discussions.unity.com/t/how-i-can-create-an-sprite-that-always-look-at-the-camera/16891

Unity Technologies. (2015, July 24). *Player Respawn Script*. Unity Discussions. https://discussions.unity.com/t/player-respawn-script/144467/2

Unity Technologies. (2016, January 28). *Floor flickering*. Unity Discussions. https://discussions.unity.com/t/floor-flickering/614465

# Bibliography


Unity Technologies. (2018, January 26). *Respawn after Collision*. Unity Discussions. https://discussions.unity.com/t/respawn-after-collision/202550/4

Unity Technologies. (2022, August 20). *NavMeshLink in Unity - Adding Jumping to NavMeshAgents | AI Series Part 2 | Unity Tutorial*. Unity Discussions. https://discussions.unity.com/t/making-agents-jump-and-climb/891660

Unity Technologies. (2023, June 12). *How to respawn a GameObject to it's Original Position*. Unity Discussions. https://discussions.unity.com/t/how-to-respawn-a-gameobject-to-its-original-position/633158/5

Unity Technologies. (n.d.) *Cursor.lockState – Scripting API* [Documentation]. https://docs.unity3d.com/6000.1/Documentation/ScriptReference/Cursor-lockState.html

UtilityFinder. (2024, May 5). *Understanding Utility Markout Colours in New Zealand | UtilityFinder NZ*. Utilityfinder.co.nz. https://utilityfinder.co.nz/articles/underground-utility-colours-nz

Vegetarian Zombie. (2023, February 21). *Beginning C# with Unity (2023 Edition) – Getting Started* [Video]. YouTube. https://www.youtube.com/watch?v=lgUIx75fJ_E

# Bibliography (Sound)


S: Heartbeats 61.wav by patobottos | License: Creative Commons 0

S: R15-44-Woman Crying Angrily.wav by craigsmith | License: Creative Commons 0

S: motor industrial vent bassy hiss.flac by kyles | License: Creative Commons 0

S: mini waterfall prefx postfx.wav by hank9999 | License: Creative Commons 0

S: Forest Water Stream Loop 2.wav by StormwaveAudio | License: Attribution NonCommercial 4.0

S: Footsteps by vitoriabueno | License: Creative Commons 0

S: Water splash & drops 1 by giddster | License: Creative Commons 0

S: Retro/classic game music.wav by ZHRØ | License: Attribution 4.0

S: 164bpm video game music (preview) by Seth_Makes_Sounds | License: Creative Commons 0

S: The Chase by Victor_Natas | License: Attribution 4.0

S: Skibka Music - Winner.mp3 by SkibkaMusic | License: Creative Commons 0

S: victory.mp3 by Loyalty_Freak_Music | License: Creative Commons 0

S: Creepy Foreboding Horror Music by Audio_Dread | License: Attribution NonCommercial 4.0

S: GameOver_Dark.wav by HarDuKK | License: Creative Commons 0

S: Tension and Chase! - Disturbed Horror Loop by JessKawaiiXxX | License: Attribution 3.0

S: 020_evil_laugh.wav by freesound | License: Creative Commons 0

S: Manic evil laugh.aiff by andymanister | License: Creative Commons 0

S: Any Old Rag (1909) - Richard Goosman - Arranged for Strings by GregorQuendel | License: Attribution NonCommercial 4.0

S: BackCafeteria.wav by ToniTobe | License: Creative Commons 0

# Bibliography (Sound)


S: Some Background I made for my game Ambient Music by DRFX | License: Creative Commons 0

S: No time to fuck around (cops on my back) by Victor_Natas | License: Attribution 4.0

S: Boss's music for game.mp3 by renatoperes | License: Creative Commons 0

S: Denim_And_Cloth_Step_Foley_07 by BlondPanda | License: Creative Commons 0

S: Metal_push door open3.wav by pagancow | License: Creative Commons 0

S: Rattle_Snake.wav by 7h3_lark | License: Creative Commons 0

S: Rumbling Rocky sound by Opsaaaaa | License: Attribution 3.0

S: Footsteps_Mountain_Boots_Tile_Mono.wav by Nox_Sound | License: Creative Commons 0

S: splashing footsteps1.mp3 by soundmary | License: Attribution 4.0

S: Jumpscare 1 by zombyKlr | License: Creative Commons 0

S: LIGHTBULB SMASH 004.wav by sandyrb | License: Attribution 4.0

S: Metal Creak 2.wav by ingudios | License: Creative Commons 0

S: New Planet Explorer by SergeQuadrado | License: Attribution NonCommercial 4.0

S: Sound Design Elements SFX PS 008 by AudioPapkin | License: Creative Commons 0

S: contact metal cage opening and closing locks horror sounds mono.wav by Soundholder | License: Attribution 3.0

S: Liszt - Liebestraum No. 3 (Love Dream) - Arr. for Music Box.wav by GregorQuendel | License: Attribution NonCommercial 4.0

S: Movie Action Trailer Marimba Female Male Choir Atmo Music Cinematic 120Bpm Mastered.wav by szegvari | License: Creative Commons 0

S: Radio changing channels.wav by Bpianoholic | License: Attribution 4.0

S: Modern Tech Logo by SergeQuadrado | License: Attribution NonCommercial 3.0

S: DSGNStngr_basic trailer boom impact by harrisonlace | License: Creative Commons 0