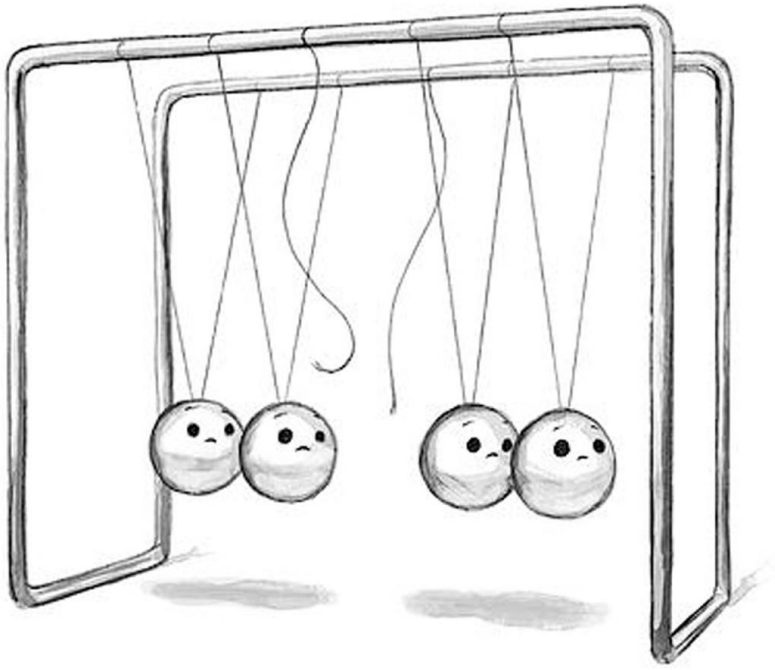
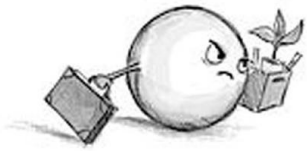
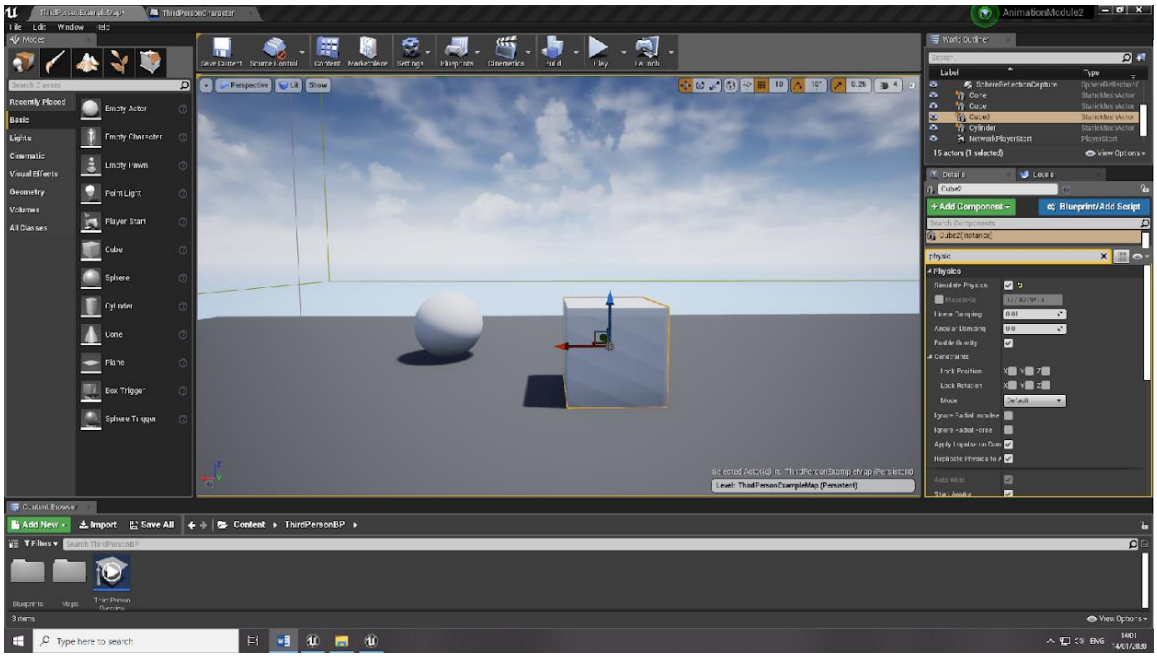
Module 6: Game Animation
Tutor: Chris Butler
January - May 2020

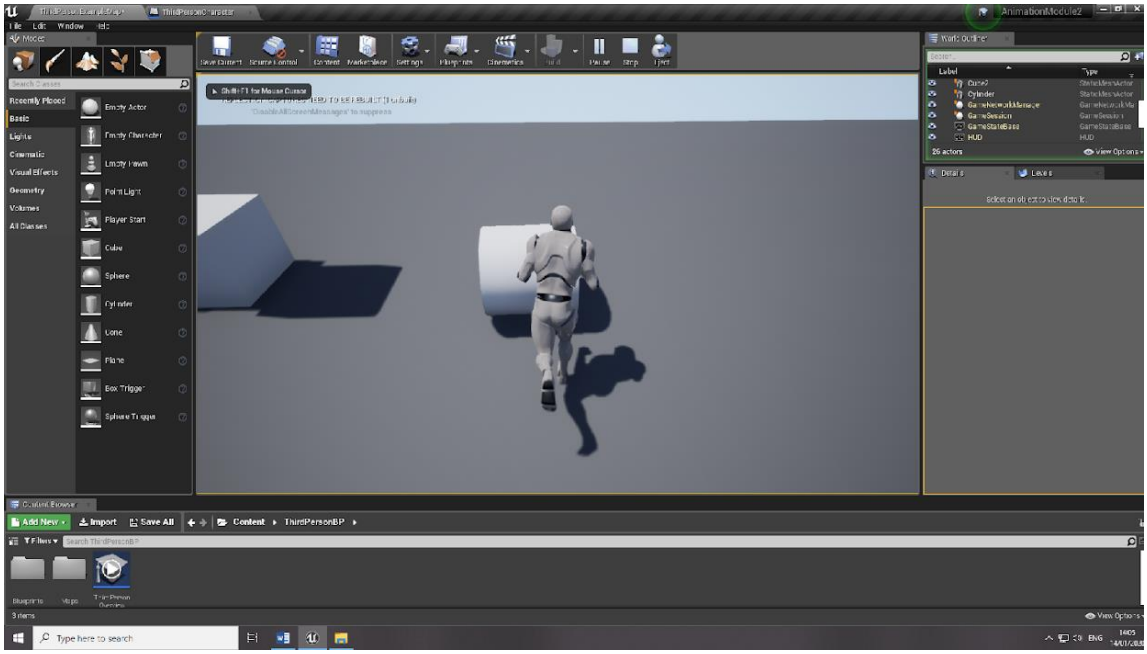




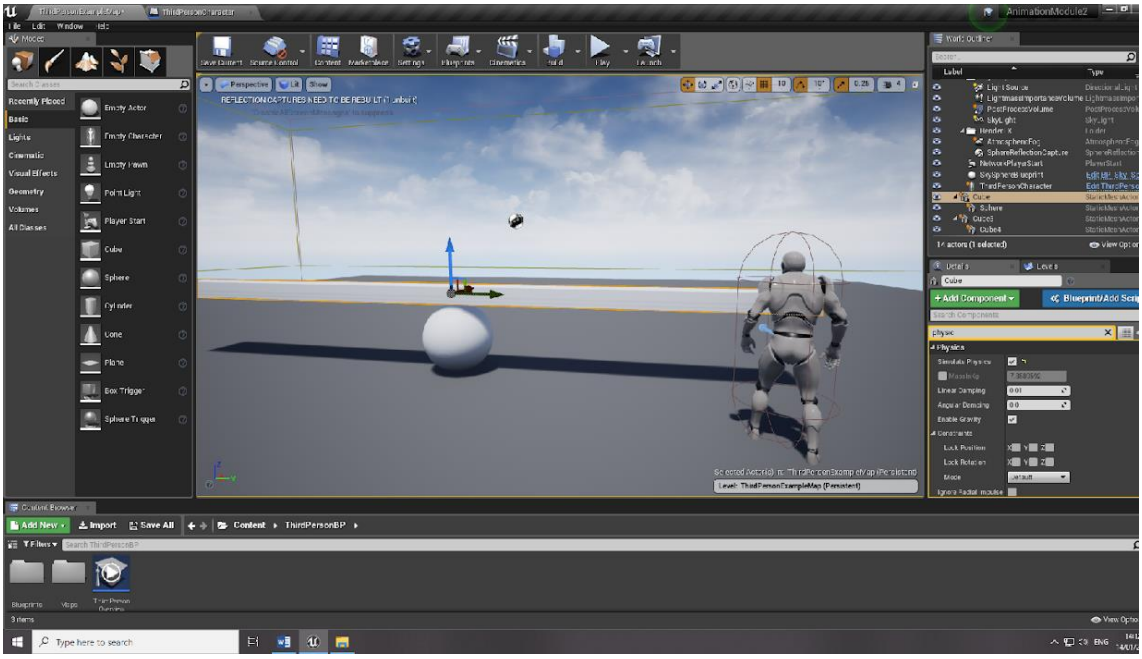
Eve Victoria Ryan

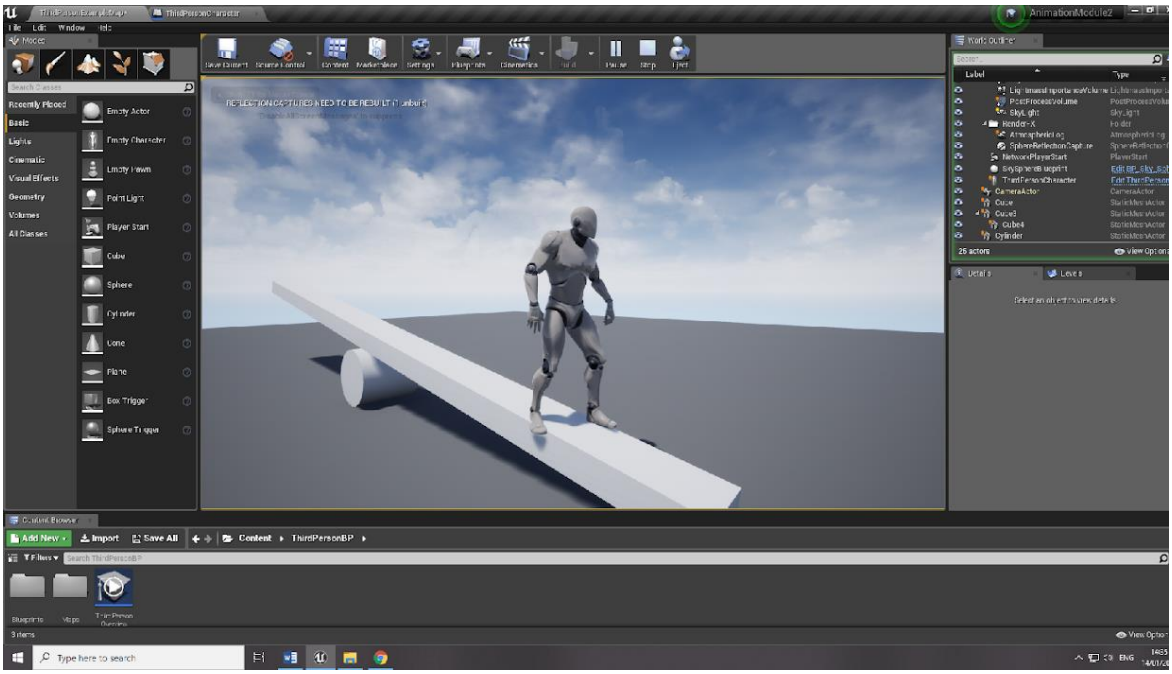
In our first lesson, we learnt a little about how physics works in unreal engine. We started off my inserting random shapes at different sizes, and ticking the ‘simulate physics’ button. We also played around with changing the weights of the items to see how they worked, and tried to get used to the motions and physics within the engine. Unreal has done really well in simulating realistic physics, however sometimes a couple things wouldn’t go as I wanted at first.



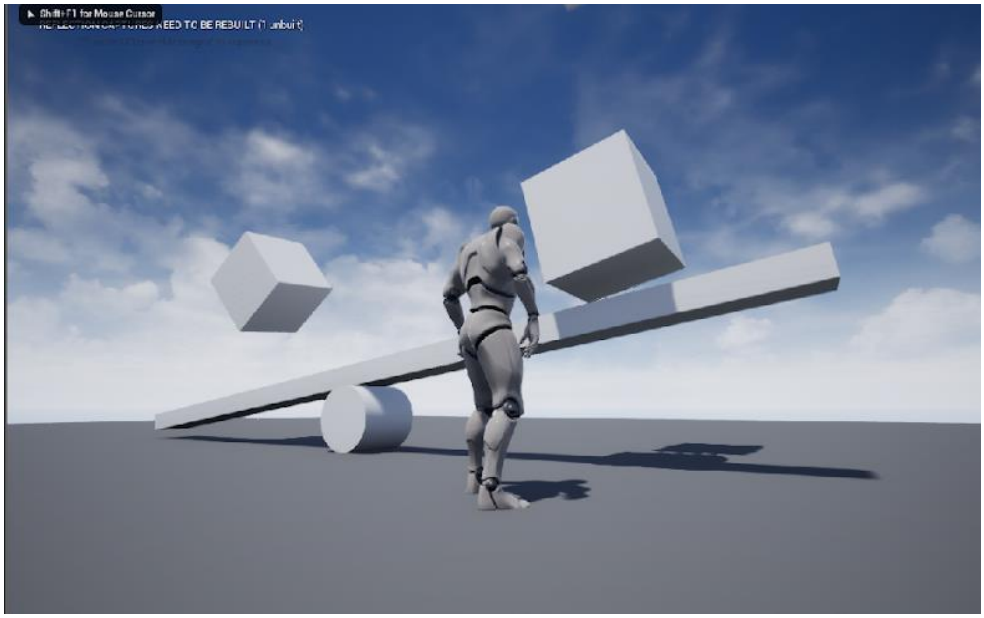


I tried rolling and moving different shapes around, as shown here. We were tasked to try and make a functioning seesaw, which was difficult at first as the base of the object wouldn’t stay attached to the ball, so I decided to change the ball to a cylinder so that it wouldn’t topple sideways. I then grouped the two items together, so they were attached to each other as one item. From this, I also locked the axis (after playing around with the different ones) so that the cylinder wouldn’t move, however the extended cuboid on top could only move along one axis, to function as a seesaw!



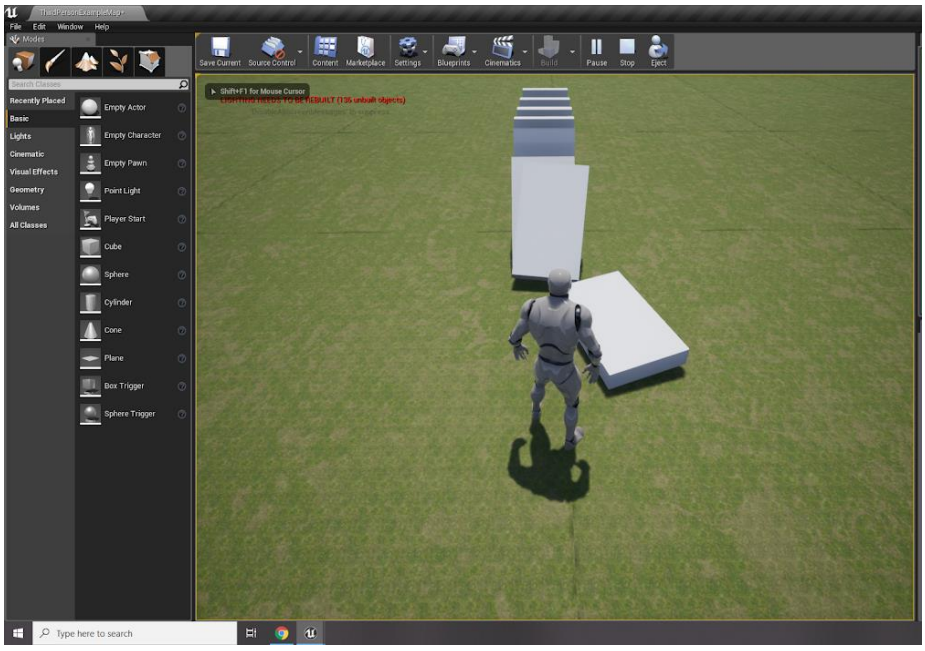


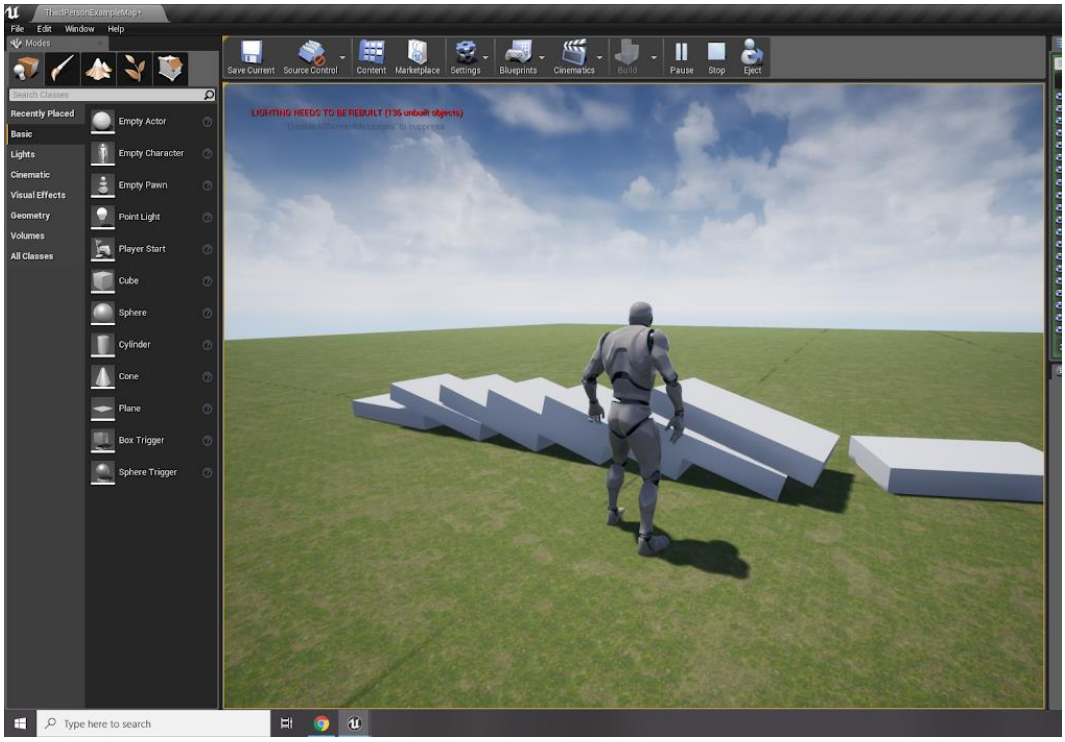
After this I tried to make some kind of a catapult, by dropping the cube on one side whilst another cube rested upon it. This was so I could get a grasp for different weights and see how things travelled in different directions from applied forces.

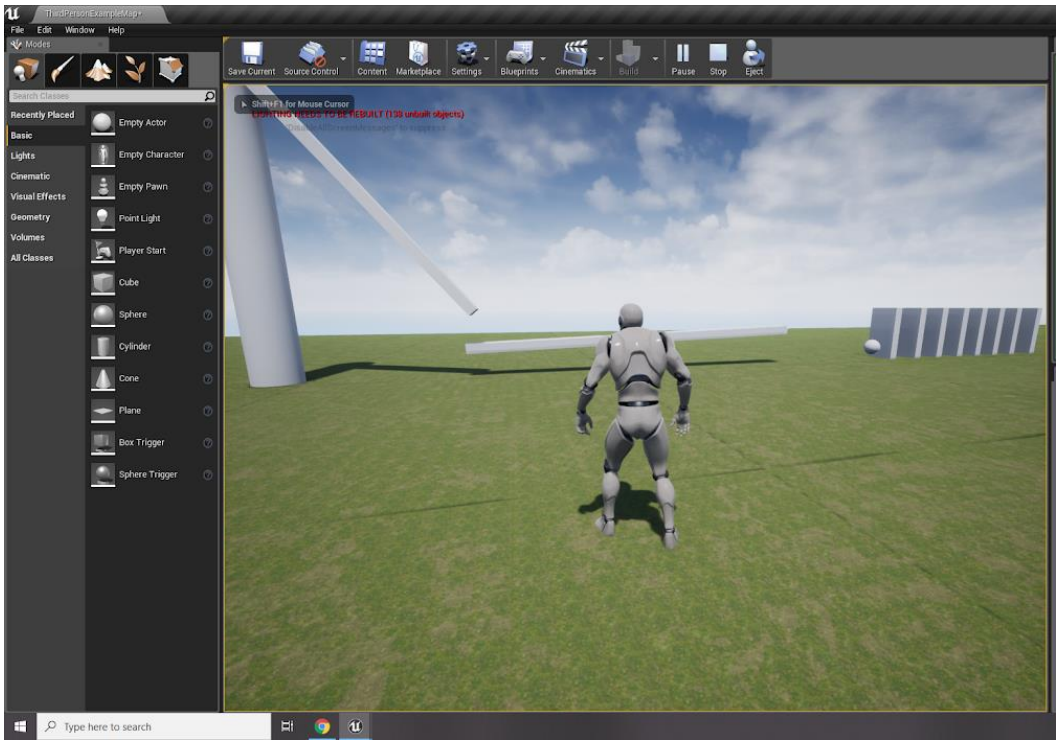


LESSON 2

Today, I decided to create some form of dominoes, and succeeded. I wanted a ball to roll down a hill, and start the dominoes falling. After playing around with different heights and weights, I achieved my goal and it worked!







Physics Based Animation

Physics based animations are computer graphics which simulate real life behaviours of matter at interactive rates. These are used in movies, TV, games and much more. They are used for more visual appeal than numerical accuracy. For example, Star Trek II: The Wrath of Khan used a new technique of particle systems in an explosion scene to create the visual of a shockwave taking over a planet. They used system shock, and the physics in the engine to create this look. Half life 2 was one of the first games to use an advanced physics engine, incorporating pulleys, leavers and many different physics based puzzles. Physics engines are now used in the majority of games available today.

There are 6 sub categories for physics based animation: rigid body simulation, soft body simulation, fluid simulation, particle systems, flocking, and physically based character animation.

Rigid body simulations are cheap, and easy to work with and implement. This is why RBS are the most common feature in games, and are in older games.

Soft body simulations are implemented by using a feature called spring mesh systems. They are made up of individually placed particles that are drawn to one another by using a spring force simulation. They are not easy to control, nor accurate. This is why many game artists used rigid body simulations for people in games. However, soft body sims are very simple, and fast, which is why they are used in games today.

Particle systems are one of the most popular techniques for big visual effects in film and games. This is due to how easy they are to manipulate and control, as well as their efficiency. Particle systems have three phases, which are generation, simulation and extinction. They work by creating themselves, and destroying themselves once they reach the end of their life span. These are choreographed through ‘Sims Paper’. An early example of the first particle effects

were when game designers used really big particles for smoke, however when interacting with other objects, they would look unrealistic.

Fluid simulation is very complex and can cost quite a lot to implement within media and games. Fluids interacting with different objects or opposite gravitational forces are extremely complex to evaluate. Most video games simulate the height of the water and then implement the waves and ripples themselves. Not many games simulate big bodies of water or other liquids in games, however surface elements are simulated to create the liquid. Some game developers use particle effects for waves and ripples as it's a lot easier to create and control. Flocking, which is a very interesting physics animation, is what is used to model the intricate behaviour of swarms of bugs, schools of fish and birds.

The flocking effects simulate the velocities, no collisions, and keep the crowding under control. Flocking can also work on crowd control for humanoids too. Unreal Engine and Half life were the first games to implement the flocking into their media.

Physics based character animation uses keyframes that create animation through many minute motions, compiled together to look complex. They usually don't portray complex interactions with the environment however. It has not been a fast process in the game industry, as it is a very expensive process. A game which has used PBCA would be Skate, and StarForge.

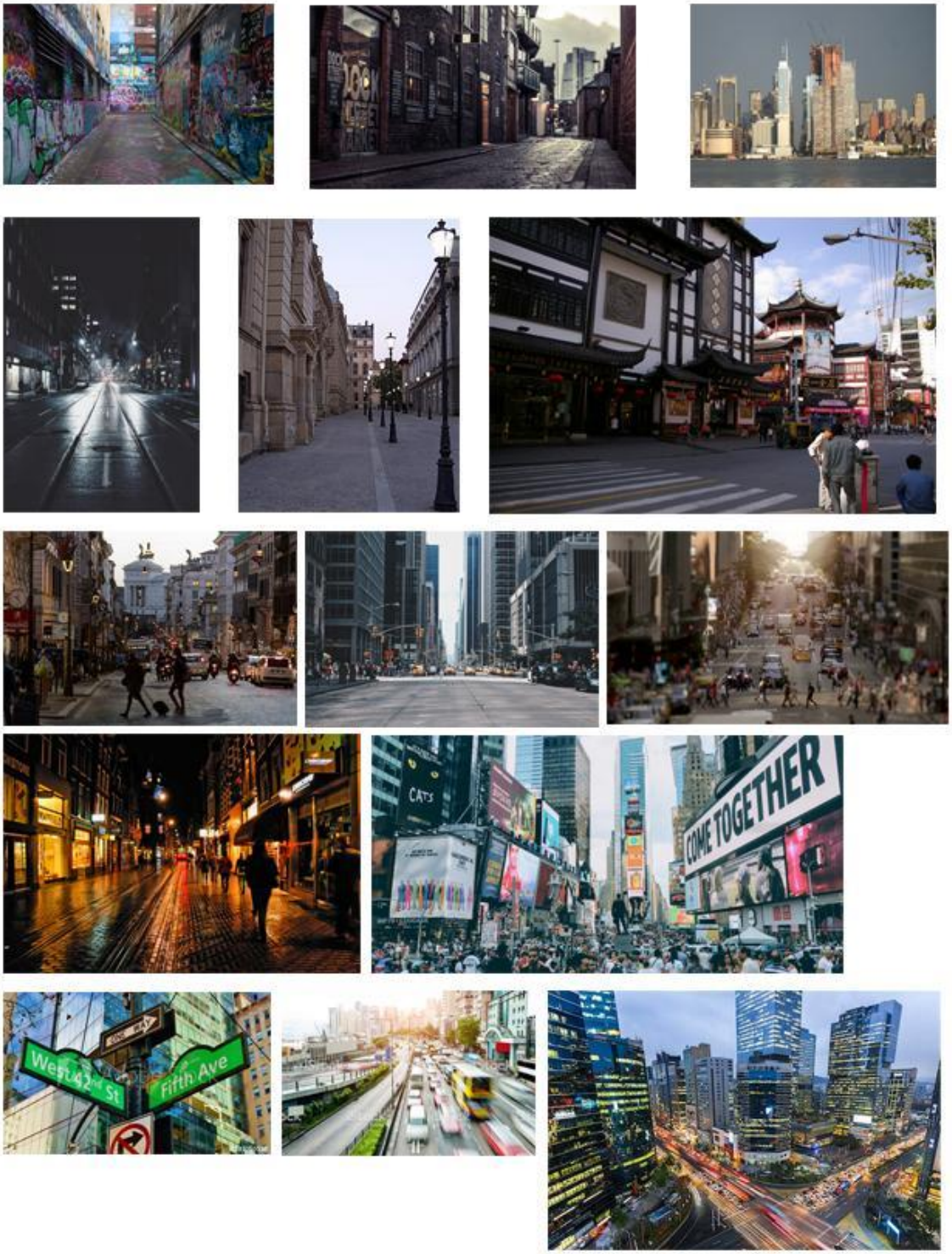
(Physically based animation, 2020)

Pre production:

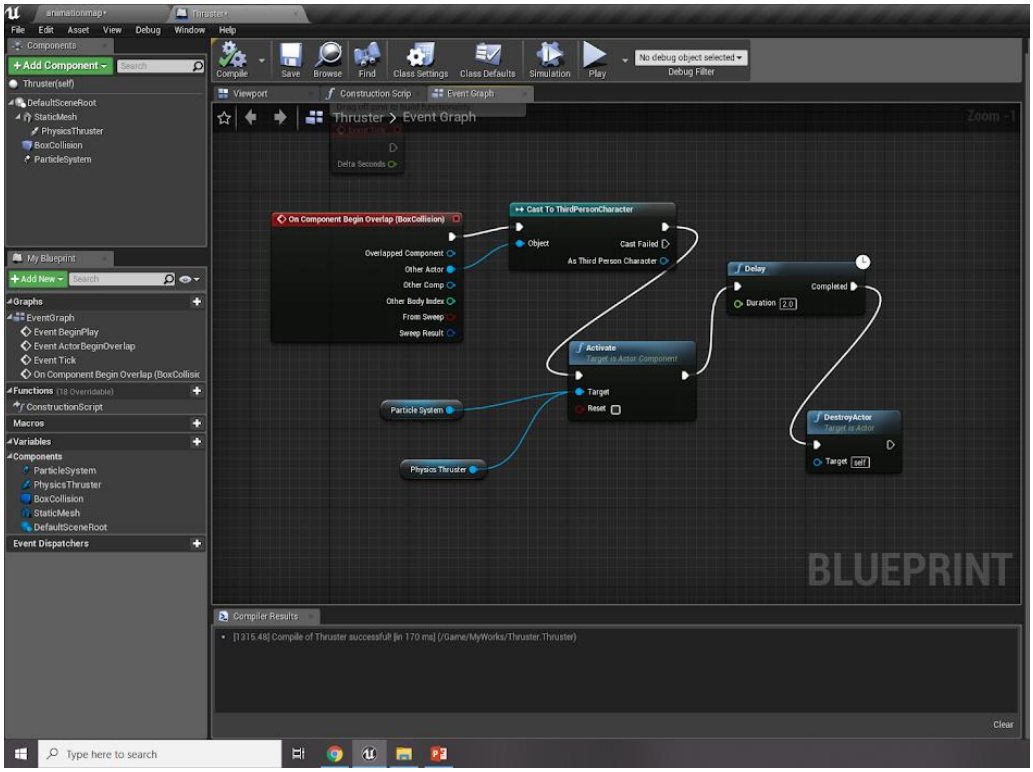
I decided with my friend Becca, that as we are tasked to create one consecutive physics level that all flows together with the class, mine and Becca's will follow each other's directly. I have decided on creating a street in Japan, as I have the Soul City texture pack already with a lot of

beautiful meshes which fit the idea. My goal is to have some sort of chaos happening in the streets, filled with lots of physics animations to create a visually stunning effect for my audience. When my last animation happens, it will end down an alleyway, and then carry on into Becca's level.

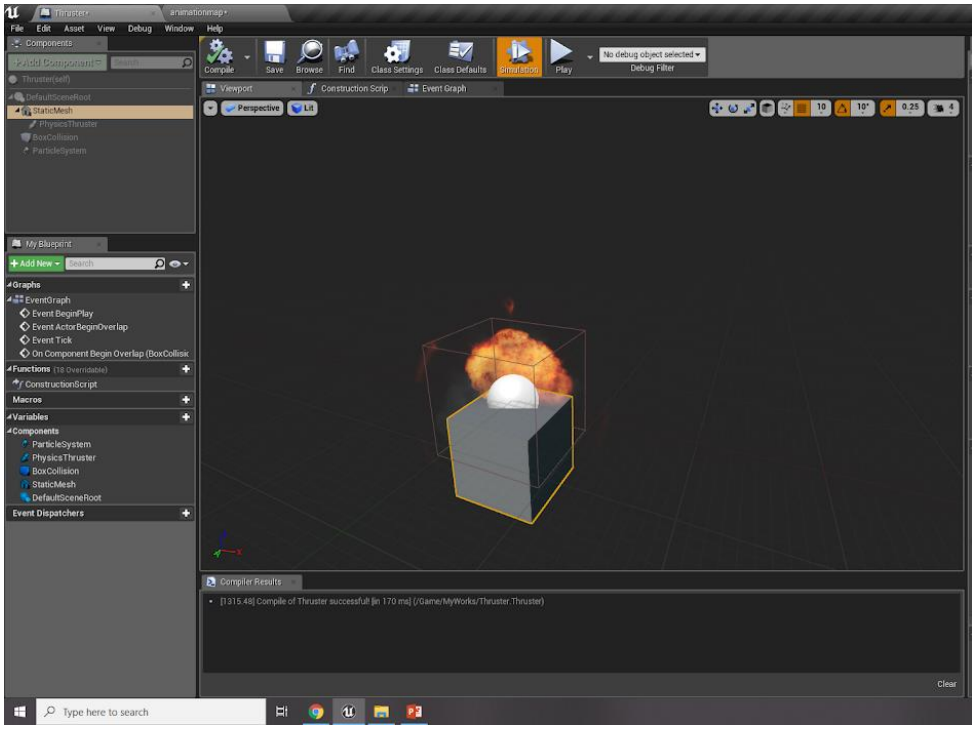
I drew a lot of inspiration from animes I used to watch as a child. I thought about all the different city settings, rural areas and possible colour schemes I could have within the level. I also wanted to think about whether this could be a side scroller, or a first person level, or even a third person level like last time. I decided that the 3D physics animations would look the best in a side scroller, as I could demonstrate them in order as the camera follows along. Almost like a domino effect of animation. Here is a moodboard of the kind of theme I want to go for.



Today we were tasked to make a ‘Thruster’. This would mimic a small explosion within a particular object, when coming into contact with my player character. Physics thrusters are a special physics actor that can be put in the editor and then parented to an object that is physically simulated. The thrust pushes the Physics Thruster in its Negative X direction, or if you look at it another way, the thruster's Positive X axis is the "rocket engine exhaust."

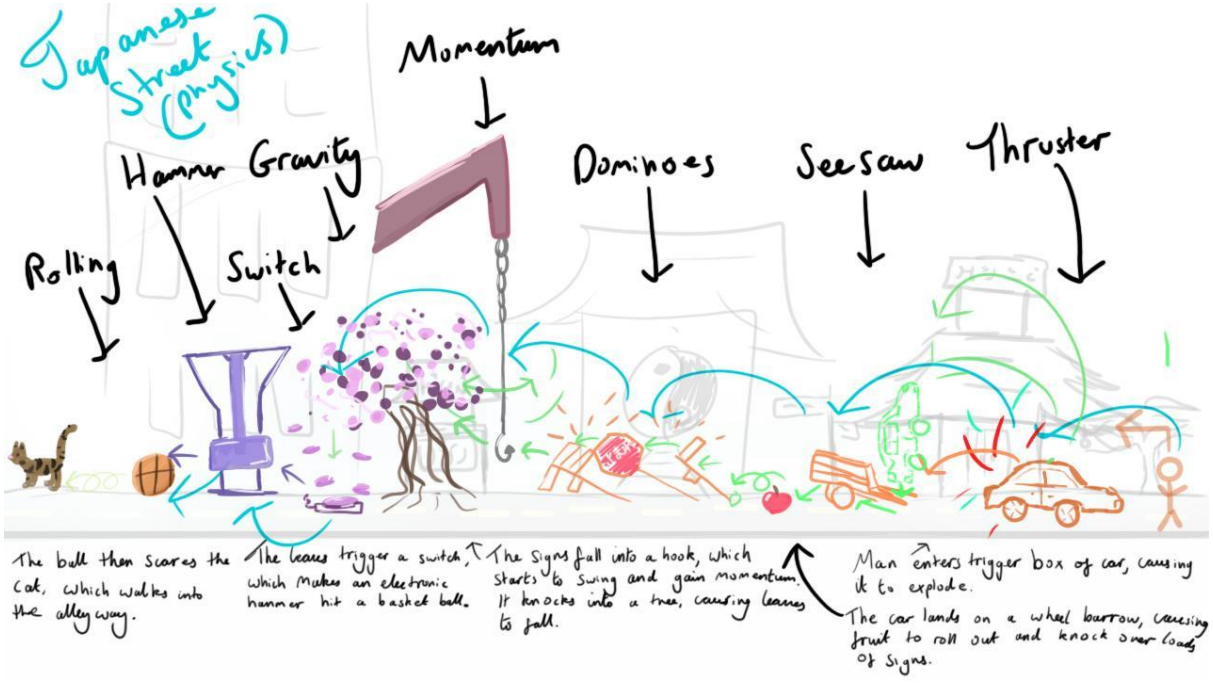


Here is my blueprint for my thruster. When my character overlaps with the trigger box, it activates the targeted thruster, and has a delay of 2.0 seconds before it destroys itself.

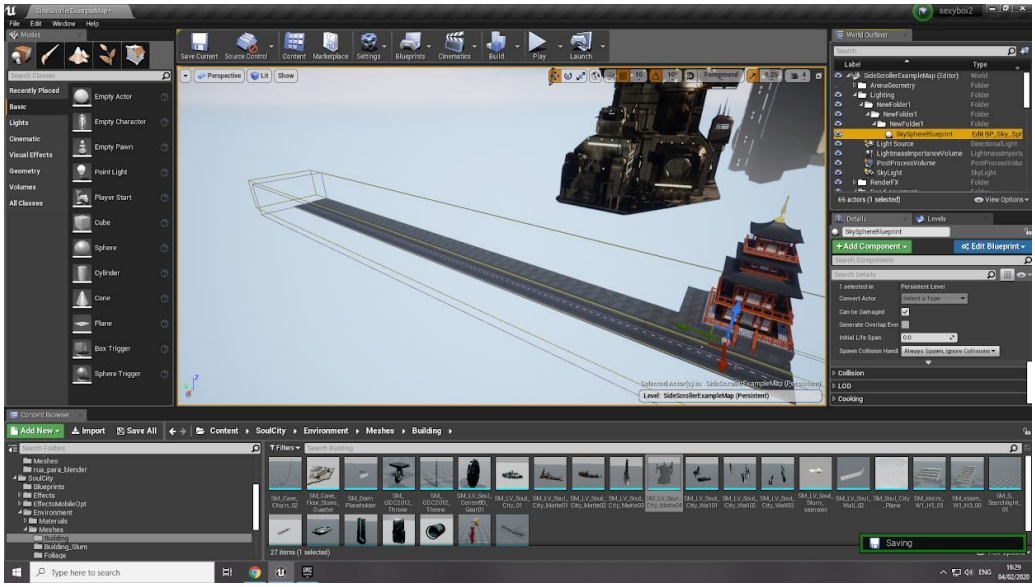


My box kept falling when I came into the collision box, however I forgot to tick the ‘Simulate Physics’ option. The box now launches when my character walks within the box. I may do this with a car in my street, or some other object which would look insanely cool to animate.

Here’s a brief floor plan of what I intend to create:



After this, I decided it was time to start creating my street. I downloaded a road texture, and a Japanese temple from a royalty free 3D mesh website (Turbo Squid). I placed multiple stretched out cuboids next to each other, and added the road texture to every one of them. I also made some slightly smaller cuboids, more elevated this time to create a pavement. I also decided to play around with the backdrop which came within the soul city pack from the unreal library. I played around with distance to make it look as realistic as possible. I want the city to look real, but with a hint of futurism. I also created a street light but have yet to add any physics to my road.

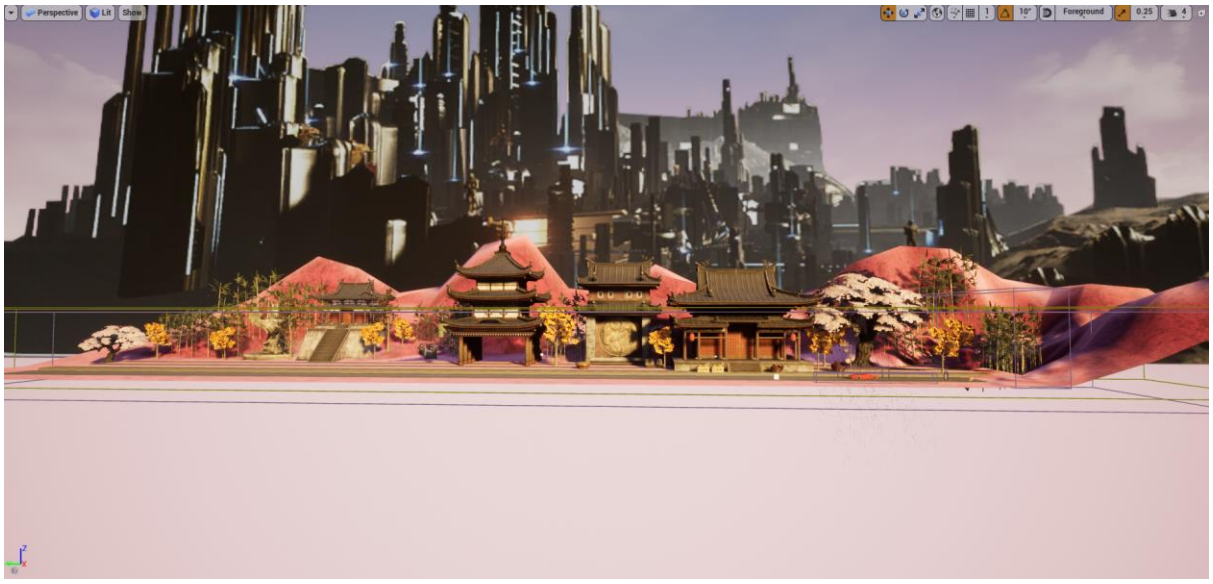


Instead of a 3D first person map, I used a side scroller, as shown in the first image, it follows the player character sideways. With this I can use forced perspective to make things look more real, and add special effects. It also gives me room to be less tidy, as the character has a set pathway. I feel like I want to add some terrain around the road, and create a realistic street, possibly pink or green to look out of the ordinary.

Level Design Completion:


Perspective
Lit
Show
Foreground


Perspective
Lit
Show
Foreground


Perspective
Lit
Show
Foreground

I really reached the aesthetic I wanted for this level design, I wanted a sunset with an orange glow, and pink hills. This looks great and I'm really happy with the way it looks. I strayed from my original idea as I thought it didn't look striking enough for my work, and I wanted to make something which looked quirky and memorable. The game I drew inspiration from, following the colour scheme was Tigger's Honey Hunt, released in November 2000 on the Nintendo 64.

(Tigger's Honey Hunt, 2020)

As much as it isn't a Japanese themed game, I remember absolutely loving the colour scheme of this level, with the intense sunset and pink/orange peach aesthetics. I wanted to draw inspiration from one of the first games I remember playing when making my own games. Even if it's a piece of animation.

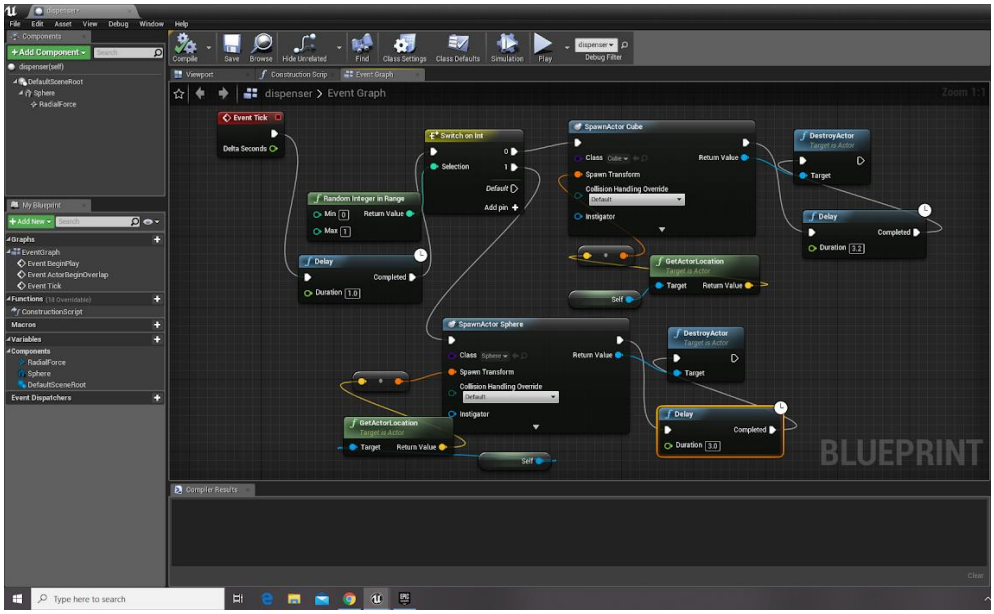
Creating animations:

The next physics animation I want to include is a dispenser, this is because I want something in my level, possibly a fruit card or a stand to dispense fruit! I think it would fit well on the street and really add to the aesthetic of a japanese high street.

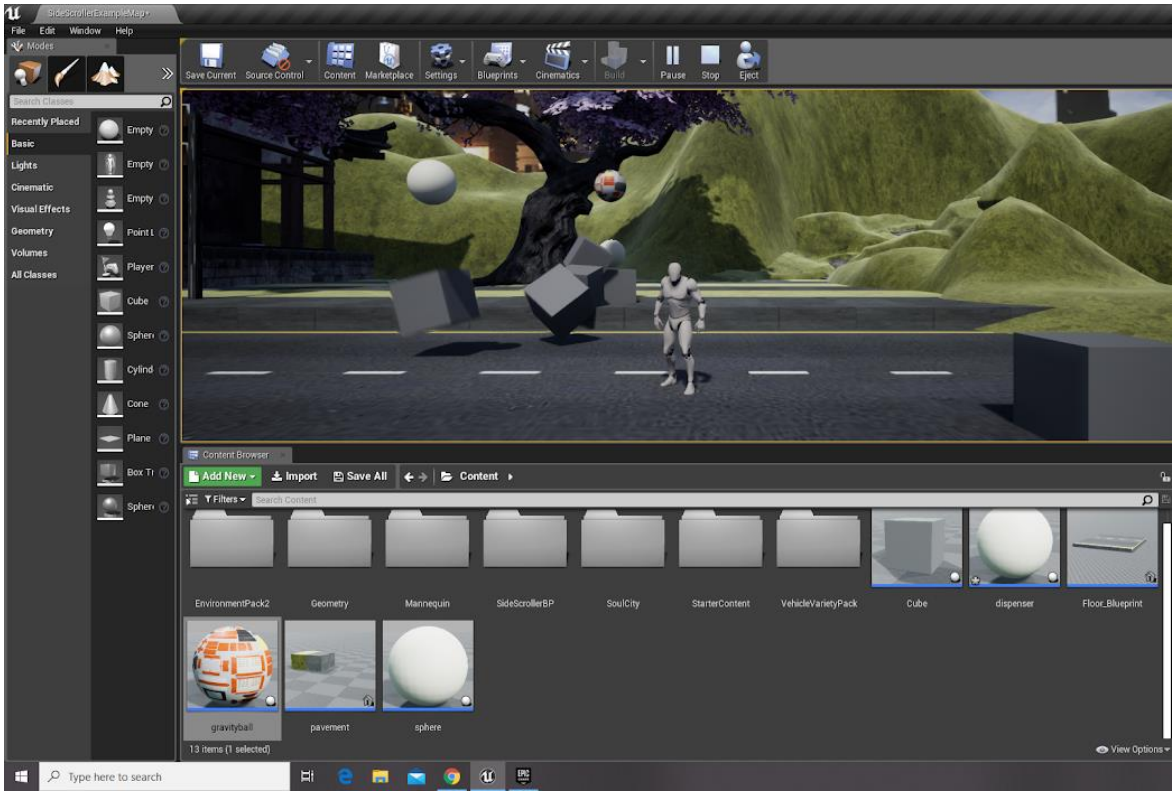

File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Build
Pause
Stop
Eject
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Content Browser
Add New
Import
Save All
Content
Filters
Search Content
EnvironmentPack2
Geometry
Mannequin
SideScrollerBP
SoulCity
StarterContent
VehicleVarietyPack
Cube
dispenser
Floor_Blueprint
gravityball
pavement
sphere
13 items (1 selected)
View Options
Type here to search

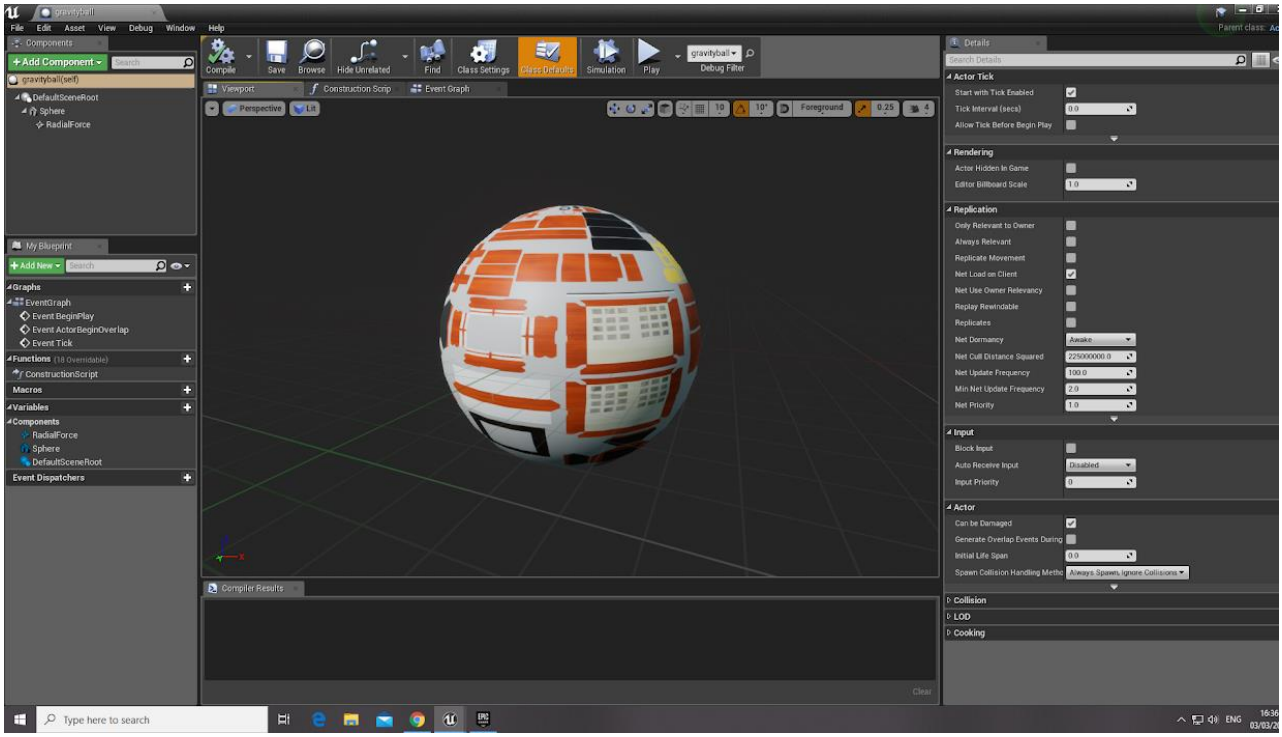

File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Build
Pause
Stop
Eject
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Content Browser
Add New
Import
Save All
Content
Filters
Search Content
EnvironmentPack2
Geometry
Mannequin
SideScrollerBP
SoulCity
StarterContent
VehicleVarietyPack
Cube
dispenser
Floor_Blueprint
gravityball
pavement
sphere
13 items
View Options
Type here to search

This is how my dispenser looks in its most basic form. I used spheres and cubes so I could differentiate between the different spawns when coding. On the Event tick, it randomly spawns either a cube or a sphere, I did this by using a switch and a random integer node. I set them to spawn in the location of the dispenser and to self destruct after a few seconds.



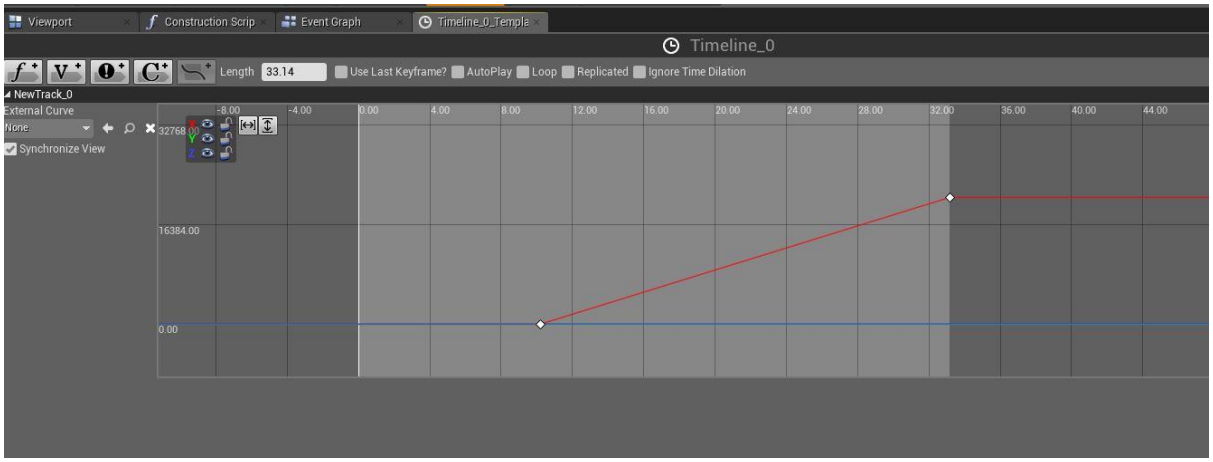
I was thinking of doing a gravity ball to pull objects in my level into itself. I was thinking of making it a UFO to fit into my quirky setting, and it could possibly be abducting something in the background of my level. I did this by adding a radial force to the actor. I thought this would look really visually striking. As shown below, the gravity ball is pulling in all of the actors spawned from the dispenser I made earlier.



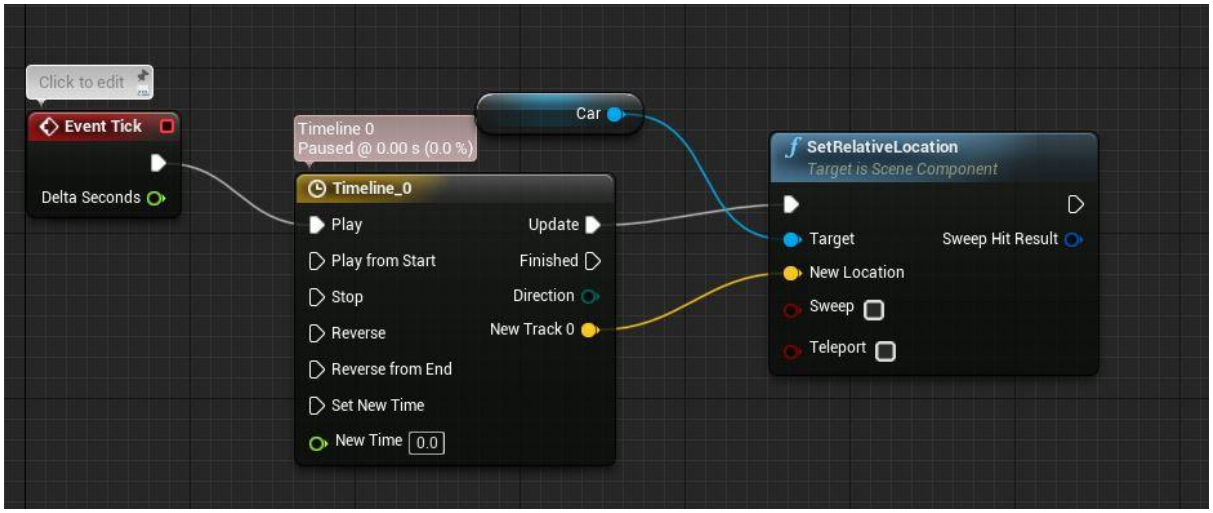


Timeline Car:

I want a car to drive past my character during my level, I am planning to achieve this using a timeline node, however I will have to ensure my car is going the right speed and the correct direction. I am used to timeline nodes as I used them a lot last term when creating my moving actors within my level.



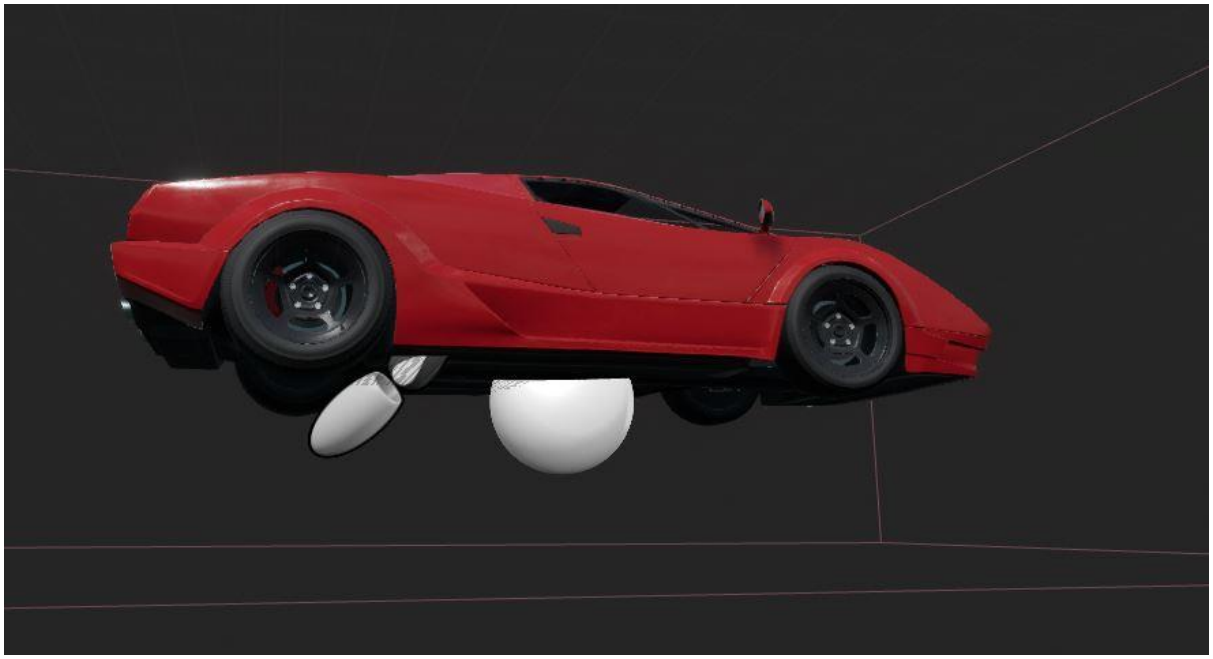
This is the timeline I made, I didn't want the car to move right as the level began so I left a small gap. I also don't want the car to return as it is going to drive into a tunnel which leads to Becca's level. It was hard to figure out which way the car would drive, and the level of speed.



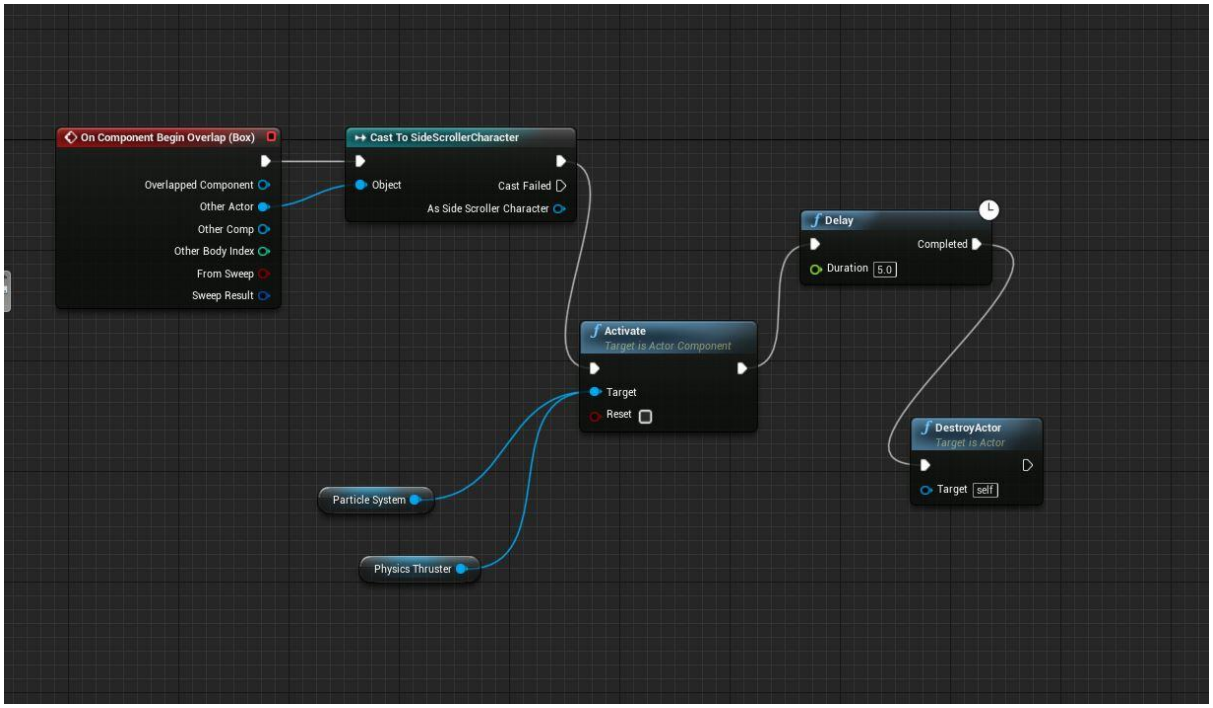
Here is the code for my timeline. I am used to making timelines as I achieved this with my moving platform section of my game last term.

Updates on my animations:

I decided to turn my thruster into a car that explodes when my character comes into contact with my trigger box. I chose a red sports car as I feel like it matches my environment niceley and matches the environment.



I placed the thruster towards the back of the car to create a dramatic flip, instead of it launching directly upwards. However, the thruster is quite unpredictable so it does change every time, but not dramatically.

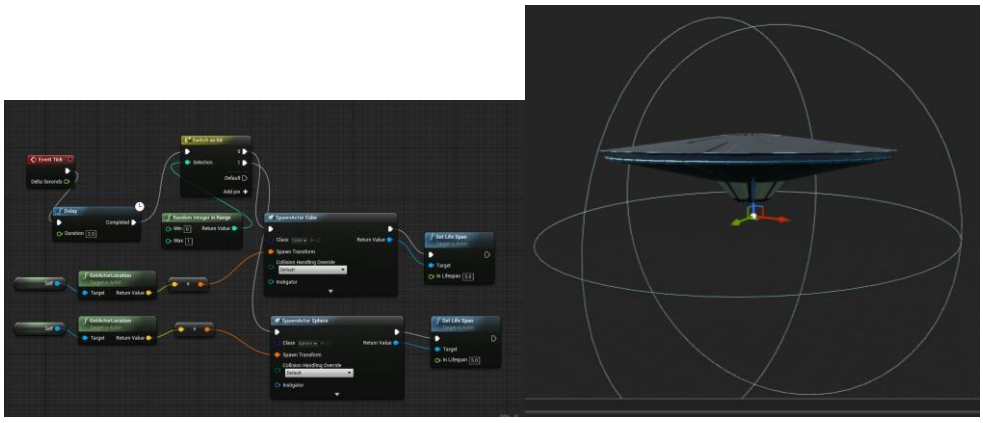


This is the code I created for my Thruster, this basically means when my character overlaps with the trigger box, it activates my thruster and particle effect (explosion), and then self destructs after 5 seconds so my car isn't going crazy for a long period of time.

I adjusted the intensity of the particle effect so the explosion is nice and big. This might take more effort to render however the visual effect is worth it!

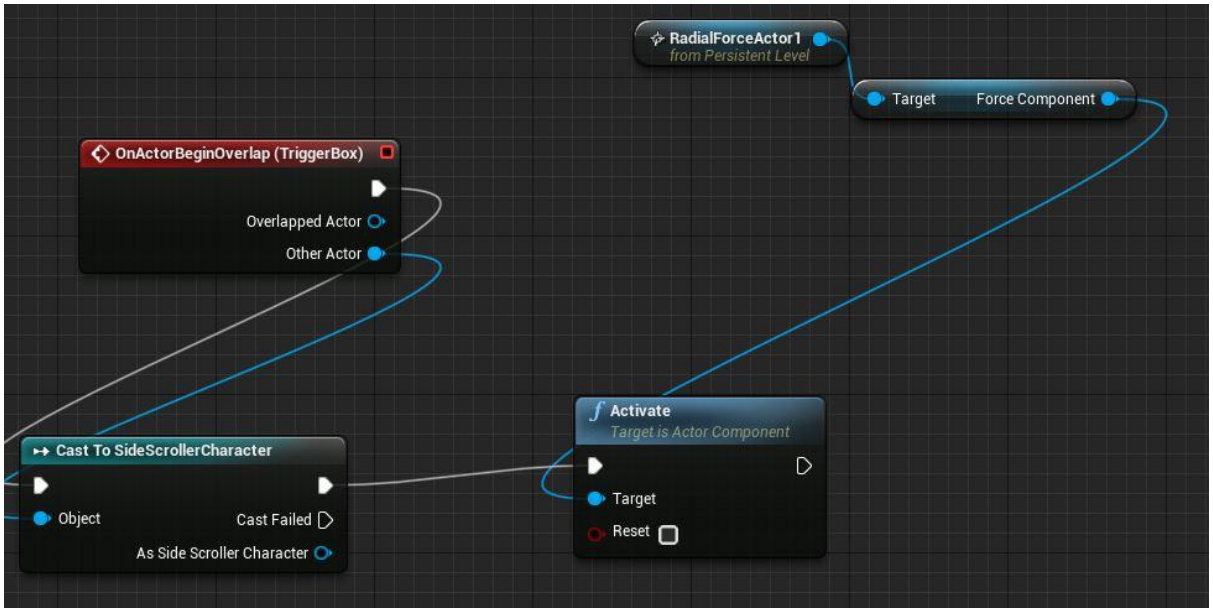
This is my updated gravity ball (spaceship), and dispenser. I thought it would be a cute idea for the spaceship to abduct all of the fruit being dispensed from my fruit dispenser. I downloaded the mesh from a website and changed the gravity balls mesh to the UFO. I had some issues with the gravity ball rotating, as it spun on a strange axis, like a penny spins when it's about to fall flat. I had to play with a lot of settings to have it just spin on one axis at a slow enough speed to be visible.

I changed the mesh of the sphere and cube being spawned from the dispenser to instead be bananas and oranges. I then placed two fruit basket static meshes onto the floor with the invisible-in-game dispenser above it.



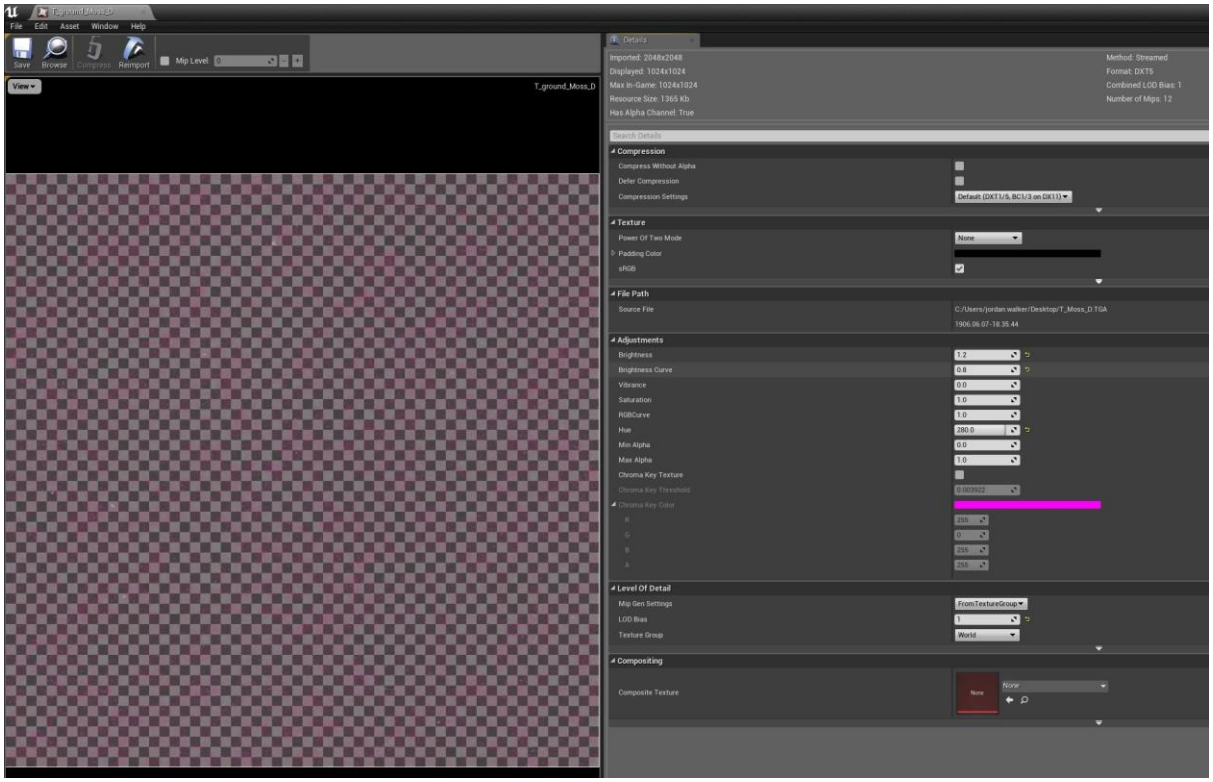
I also decided to tidy up the code of my fruit dispenser so that it was more clear for me to edit. I also had to increase the radius of my gravity ball, but not too much that it picks up my player character.

I wanted to use a radial force to push back some items away, much like a gust of wind. I placed down some stools and activated the simulate physics and changed the weights of them. This is so when my radial force is activated it blows them away. I also unticked the auto activate, as instead I want it to activate when my character comes into contact with the trigger box you can see placed.



Here is how I coded that within my blueprint editor, when my character over laps, it activates the radial force actor.

I also wanted to show how I changed my background from the default green grass to the pink grass.



I selected the material used and simply just changed the colour within the material editor, and kept editing it until I got the desired hue I wanted.

Now it was time to make my level and me and my peers wanted our levels to all flow together. It would start at Megs, and follow on a road leading into my level, this would follow the same yellow car from Meg's level and that is what is the timeline node in my level. This car in my level will then travel into a bridge, then appear in Becca's level following her cat. Then the car would enter Beth's level and park in a garage in the apartment building.

I decided to take it upon myself to make the entire menu system as I was most well versed in making them and I wanted to get it out of the way too.

I wanted the background to be a Japanese street, just like our levels, vibrant with a lot of personality.

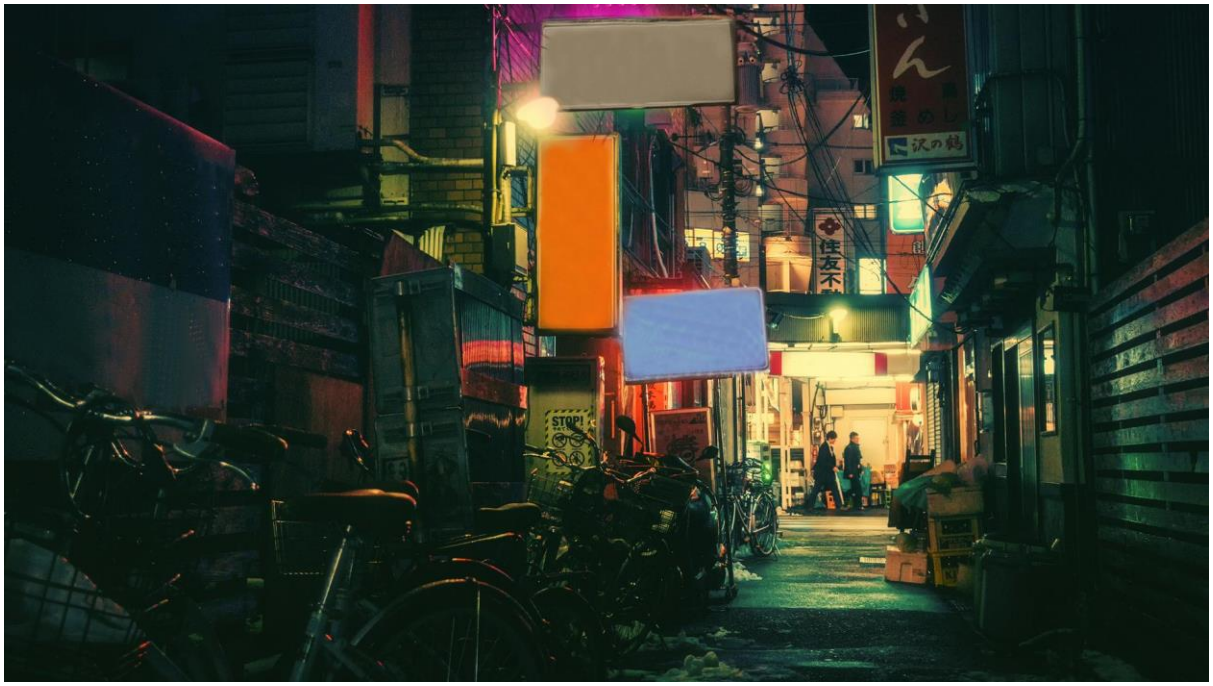
I created a moodboard to gather ideas of what I wanted my menu background to look like, from here I also decided to try and create a sign that would fit into the theme.

This is what I came up with. I wanted a neon sign to match the street signs in japan, as well as a bright and striking colour. I did this with everyone's names in photoshop and asked them what colour scheme they wanted.

I then asked everyone which background they preferred out of a choice of 10. We all decided upon this background.

(Masashi Wakui, 2020)

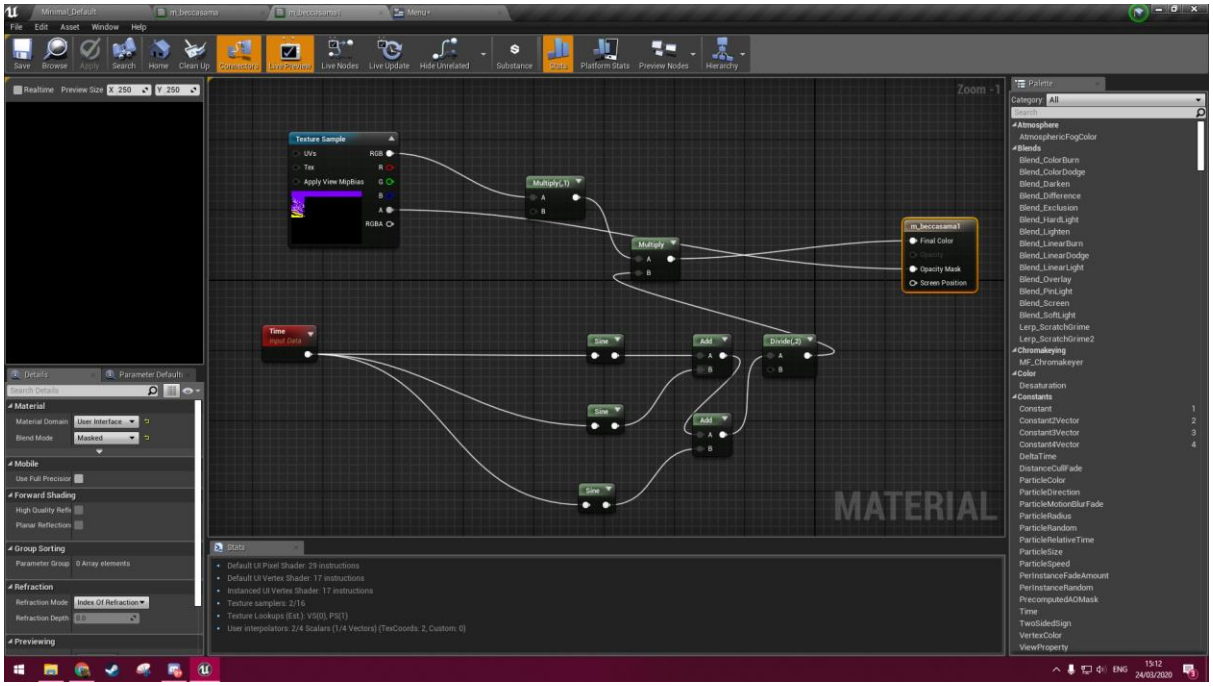
This is the image we decided to use. I started warping the signs I made to fit the shape, as well as using the content aware tool to block out all the existing signs.



It was coming together extremely well so I carried on until I was finished.

I wanted the signs to light up as the user hovered over them with their mouse, so I achieved this by brightening the buttons and placing them into the on hover slot in the menu widget.

I also wanted the signs to flicker in the background, but I couldn't texture this very well. I coded the following, however it made the whole image dark over the button I wanted to flicker.

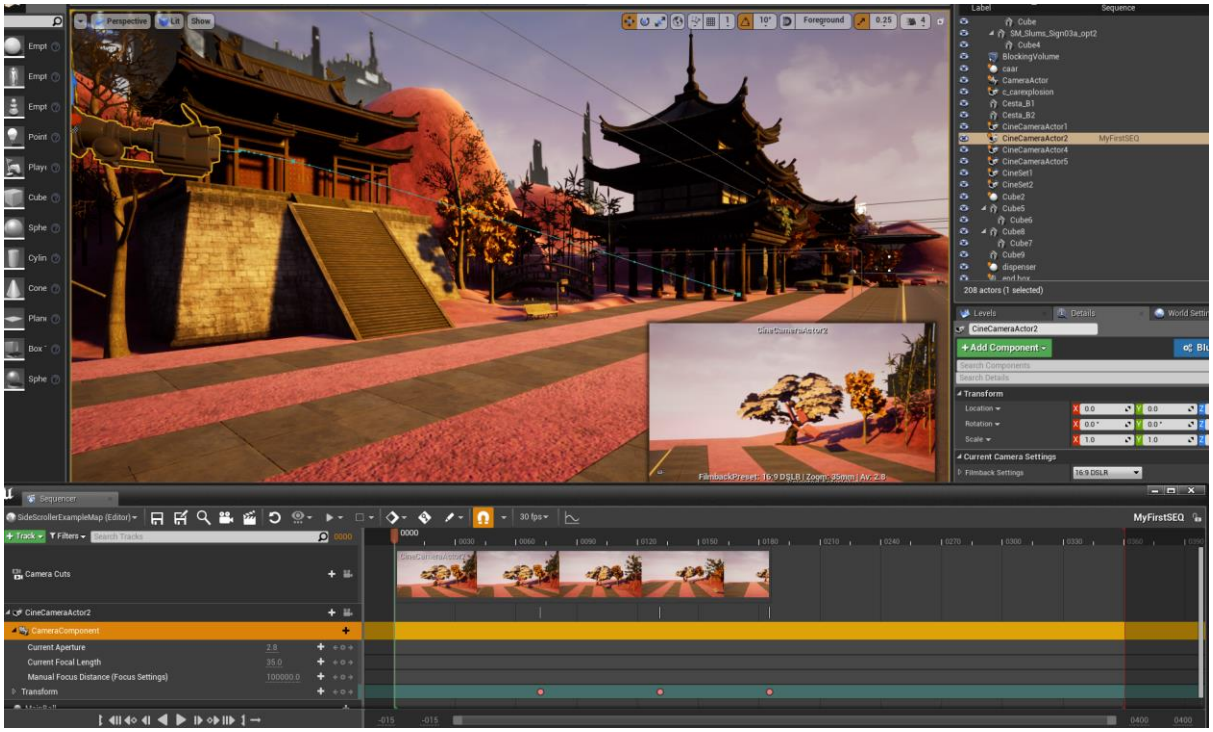


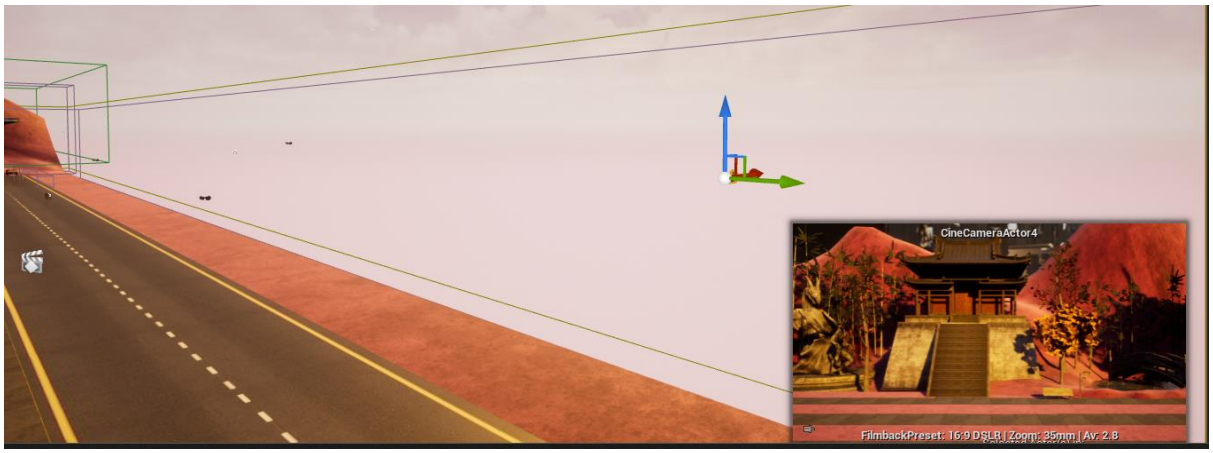
I tweaked around with the settings, and ended up creating a png image of the sign's lettering so only that would fade to black and not the whole area around the button, this ended up looking this way.

This looked better however I ran out of time and couldn't fix it. We also can no longer merge our levels due to the corona virus so the multiple buttons is pointless. I will keep them on but only have my own button working to go into my level. I don't know how to code this to make it transparent instead of completely black. I did learn a lot through this experience and feel more confident in creating materials and hope to fix this at some point in the future.

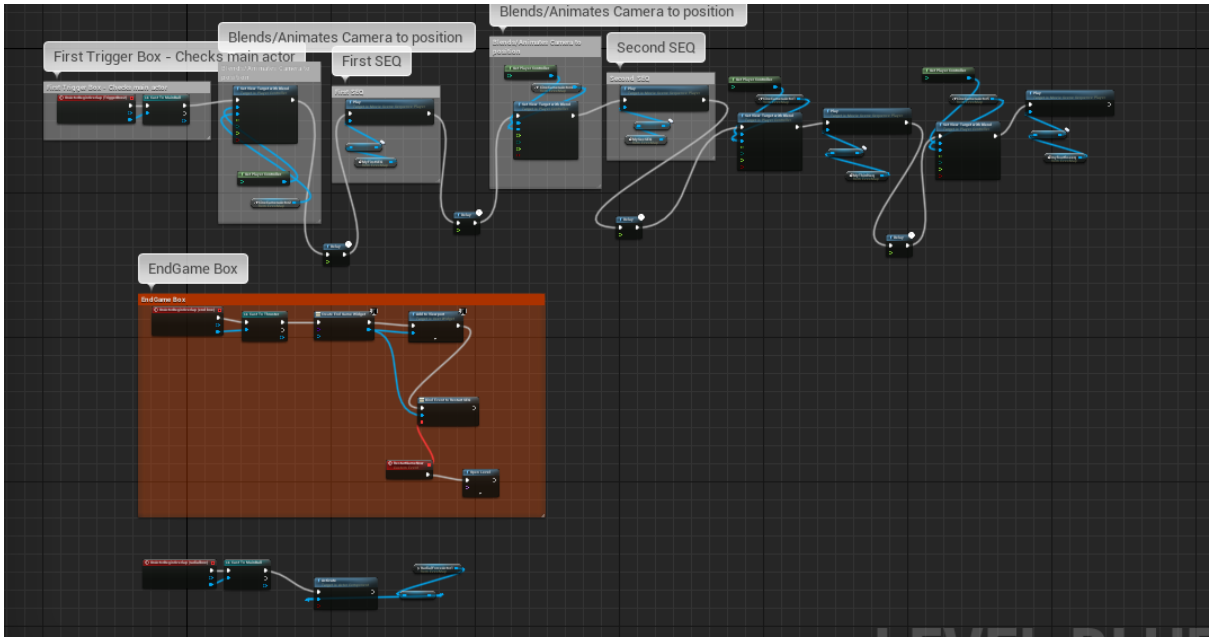
Camera Sequences

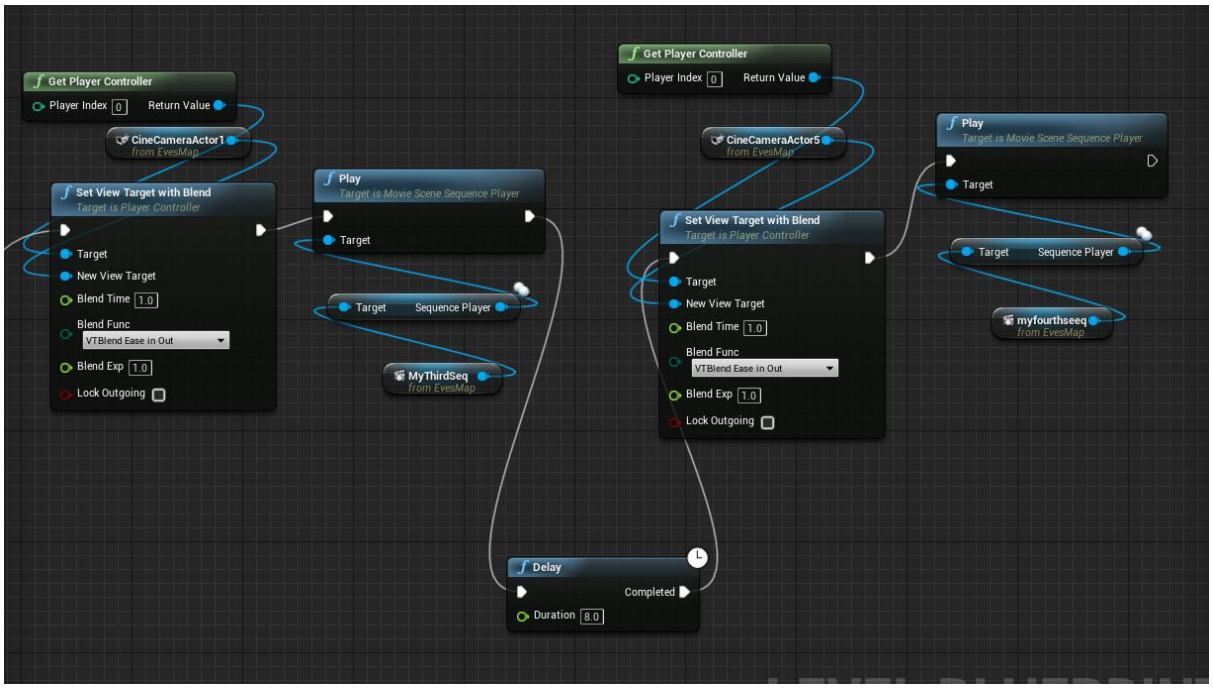
Finally it was time to add the camera sequences. First I created a new scene, and added a camera actor into the level. I connected the camera to the scene, and set my frames to 400x400 so I could see clearly. I didn't add tracking at first so I had a preview of where the camera was. I added tracks where I wanted my camera for each frame and saved after adding tracking. When it cut to my radial force, I no longer used tracking, I edited the angles of the camera as I went, whilst adding the tracks.


Perspective
Lit
Show
Foreground
CineCameraActor2
FilmbackPreset: 16:9 DSLR | Zoom: 35mm | Av: 2.8
208 actors (1 selected)
Add Component
Transform
Current Camera Settings
Sequencer
MyFirstSEQ
Camera Cuts
CineCameraActor2
CameraComponent
Current Aperture
Current Focal Length
Manual Focus Distance (Focus Settings)
Transform


CineCameraActor4
FilmbackPreset: 16:9 DSLR | Zoom: 35mm | Av: 2.8

I set the view target to the camera, set the blend mode to blend in and out, then played my camera sequence. I would then guess a delay time, and repeat the process to lead onto the next sequence.





Video demonstration of level:

https://youtu.be/wPT2qgySxNc

I had to edit the code slightly last minute as I removed my player character, which in the past activated my radial force and car thruster explosion. I instead removed the overlap trigger and

instead used the Main Ball to activate the radial force, and put the car on a timer to explode in 55 seconds, as that is when the camera pans to the car's location.

Evaluation & Peer Review

This module introduced and identified core elements for viewing polygon animations in 3D. Originally, we concentrated on the basic animation techniques and ensured that we grasped the main concepts of motion such as time, mass and action lines. Such technological elements of animation support character production and were important for ensuring successful production of physics based animation.

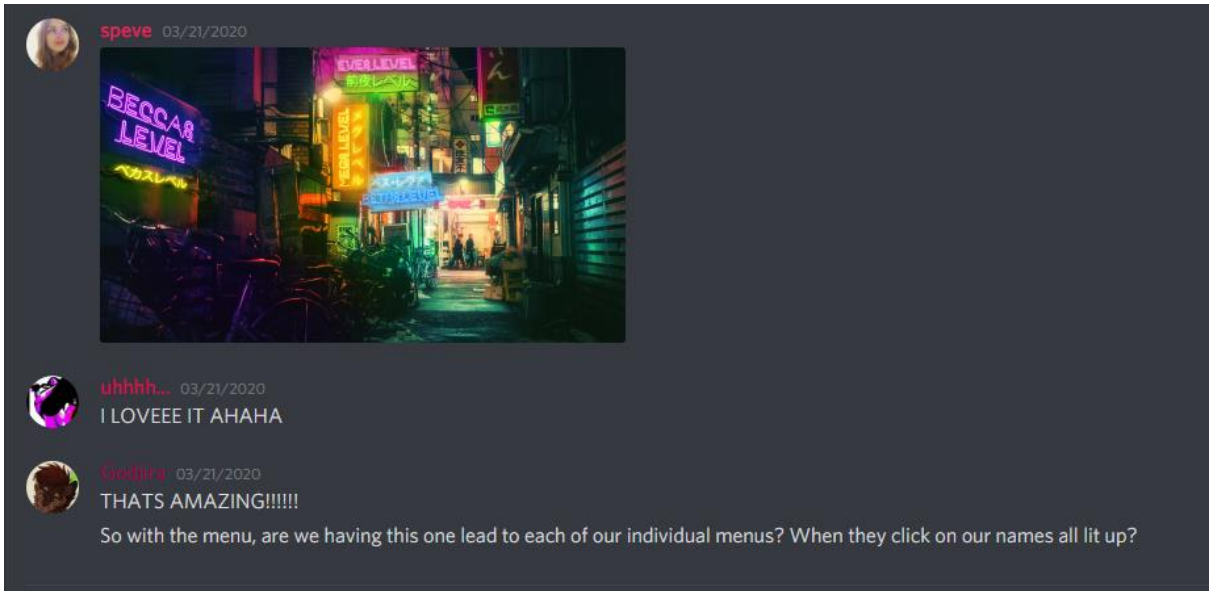
In Unreal Engine 4, we had to use our skills and tutorials provided to create a sequence, demonstrating our knowledge of physics based animation, using particle effects, timing, gravity and weight. We also had to make the level look visually appealing, and add a camera sequence, finally adding a menu system and an end game screen. I completed what the brief asked me to do, I think to a good standard of work. I produced a series of different physics based animation within the level, such as a thruster (using a particle effect). I used radial force to create wind and a pull of gravity, as well as a dispenser using a random number generator. I also successfully implemented the cameras and created a successful series of sequences, five in total. I then created my menu and I think this is the best part of my level. It would have been a lot better if it would have gone to the other girl's levels, however I think it is visually very stunning and I am very happy with the way it looks. I didn't have time to perfect an end screen, I placed a trigger box above the car to activate the end screen when the car touches it. I could have used a timer but I thought this was easier to do.

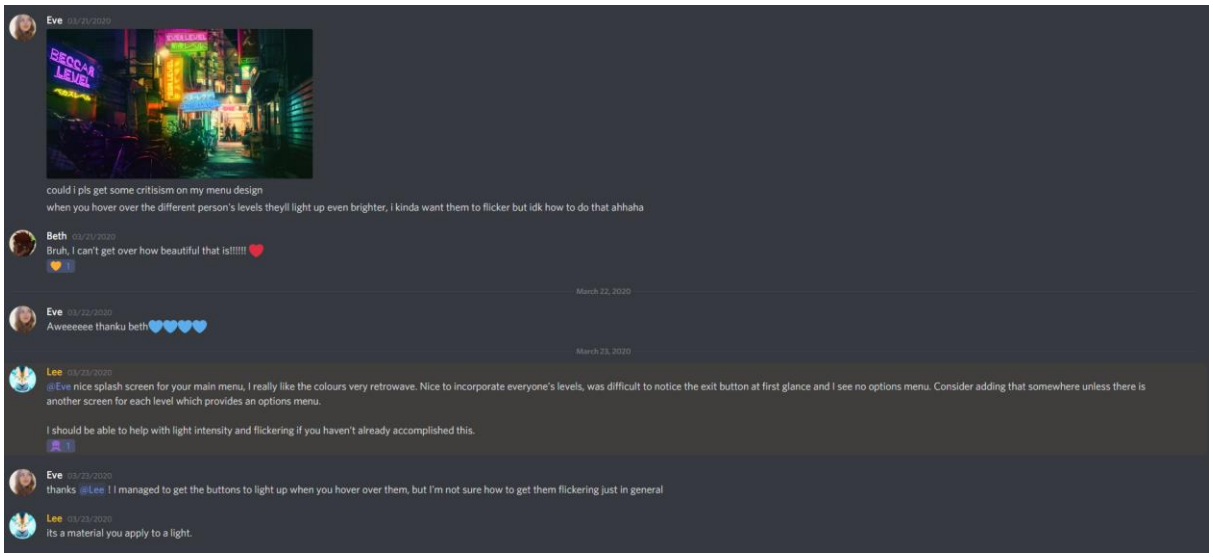
Most of my research came from using online websites. I looked into different types of physics based animation on youtube, in class and through tutorials linked by my tutor either via powerpoint or on Twitch. I looked into games which used physics based animation, as well as movies, to see what type of physics is used for what visual effects on a large scale. This was very interesting to research as you can really see how much time it takes to create these explosions and digital effects, and how it can not be done on one computer, as the rendering usually takes a whole server. I used mainly secondary sources as we have been home, and had to use only the internet, however at the start of the year our tutor talked about particle effects and water/liquid simulation and how difficult it is from first hand experience. I interpreted this research to develop ideas, as it gave me more of a scope of what I could actually do, and what would take too long to perfect. My first thought was to have a tiny microscopic world, following a bug crawling on grass, causing things to fall and create a physics based domino effect. This idea couldn't be done as I didn't have the correct meshes, or environment, however this is something I may like to look into creating in the future. This influenced me to look in online stores for free royalty free meshes, to try and piece ideas together. When researching different

environments and piecing things together, I came up with the idea of a japanese street. As did my friend, so this was great as we could piece our levels together well.

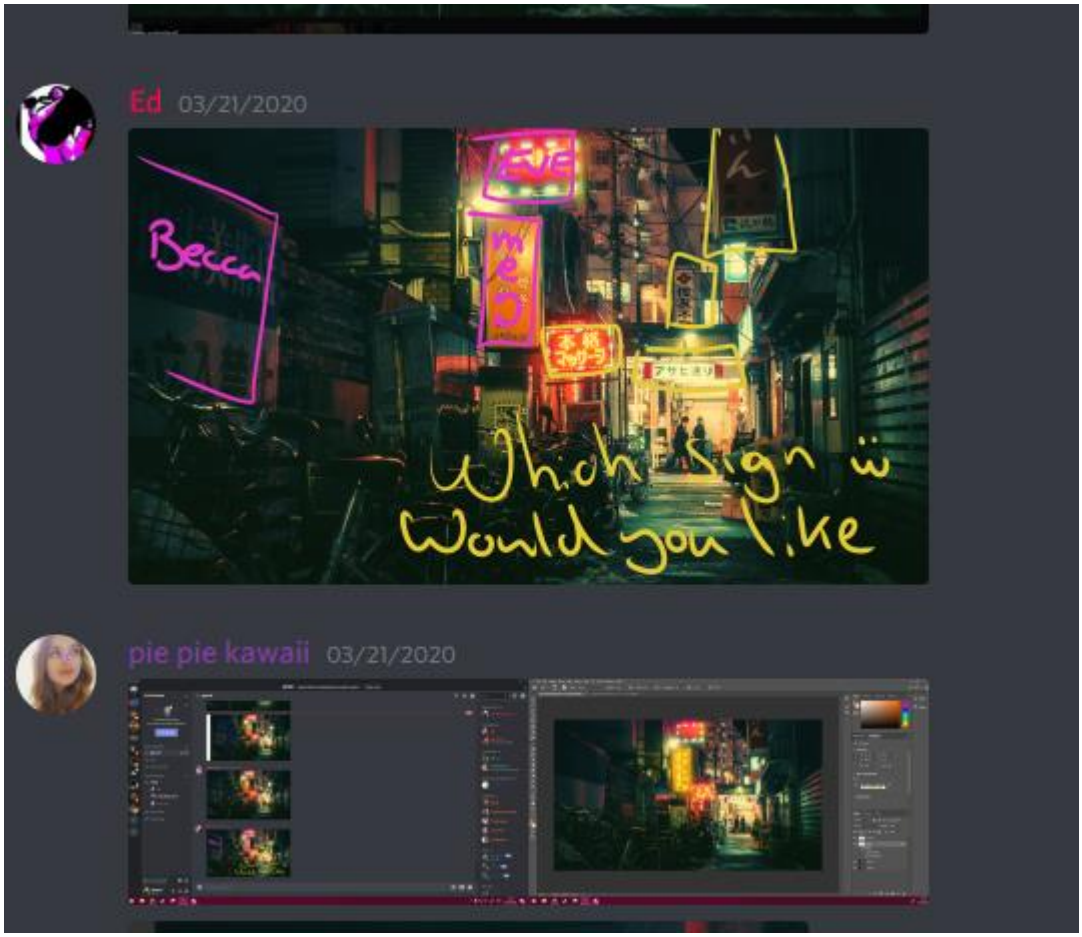
I used Unreal Engine 4 for all of the game design, as it is the only software I know for completing that task, however in the future I might choose to make my own textures either in unreal or use substance painter, as I was learning this alongside the brief. For any planning and design, I used photoshop as I use that for design in general, and have used it for years as I am comfortable with the software. The software is great to work with, and it helped me achieve my ideas to a great extent. The ease and freedom to play around with the environment, textures, lighting intensity and textures, as well as the abundance of free meshes really give me a lot of breathing room to piece together the ideas in my head and lay out an aesthetically pleasing level. The code and nodes are also becoming more and more easy to learn and they are starting to become second nature to me.

I feel as though I managed my time very effectively within the brief, I ensured I did the level design as quick as possible, as I know it's the easiest part when I have an idea, I need to finish it as quick as possible so I don't forget details. It would have been nice to sketch out in more detail what I wanted, however I thought in good time I did everything within the level. I struggled to the end as the actual mechanics of the game weren't up to scratch, particularly the camera work, so I needed some extra time to really perfect this. I planned my project in order, and made sure to research every animation thoroughly before attempting this within Unreal Engine 4. I feel like I could have improved in this a little more, however due to the circumstances it was very hard to do so. I didn't use a written schedule, however we were given online checklists by our tutor so this helped me to visually see exactly what I needed to do for when.I did mean the deadlines set by myself which was good, as I really got my head down with this brief and learnt a lot, pushing myself to try and achieve the best I could in these circumstances. I organised my workload well, however it was a little stressful towards the end of the project.





Here is the digital peer review I asked for. I was also speaking to peers whilst creating the menu asking for critiques as I designed, as well as all of us choosing a background together.



As Lee said, when maybe applying a material to the signs to create them flickering, I tried this for a while, and I was on the home stretch. However I just couldn't get the material to not fade to black. This was really helpful for my learning however unfortunately I couldn't apply it to this project due to me having to move on within the project.

Beth also recommended I make the signs flicker as I hover over them. I had this idea in my head anyway, but it was good to hear that someone else also thought it would look really good.

Within the actual level, when I chose to have a pink background, I asked my peers what lighting they preferred, a head on sunset so everything was illuminated, or a duskier look. Most people such as Patryk and Dylan suggested I go with this sunset idea as it really complemented the other meshes I had within my level.

Bibliography:

Disney Wiki. 2020. *Tigger's Honey Hunt*. [online] Available at: <https://disney.fandom.com/wiki/Tigger%27s_Honey_Hunt> [Accessed 17 May 2020].

En.wikipedia.org. 2020. *Physically Based Animation*. [online] Available at: <https://en.wikipedia.org/wiki/Physically_based_animation> [Accessed 17 May 2020].

Flickr. 2020. *Masashi Wakui*. [online] Available at: <https://www.flickr.com/photos/megane_wakui/> [Accessed 17 May 2020].