Game Design Document

Cepαleo

EMP Studio

September 2020

Version 1.0000kαppα

1. Introduction

1.1 Scope

This document is intended to be read by programmers, artists and producers involved in the design, implementation and testing of **Cepαleo.**

1.2 Type convention

Cursive will be used to explain what kind of information a chapter needs.
Arial font will be used for items which are agreed upon. Times font will be used for items which have not been officially agreed on. *Comic Sans* font will be used for items which still needs to be discussed

1.3 Current status

The project is set up in Unity and saved in Azure Devops, with everyone set up to connect it with the program Sourcetree (Git GUI). The main synopsis of the story has been written down in big detail, the smaller details will be written when the game is further in production. Also the sprite for the player is finished together with the movement controls, dash ability and shooting controls.

For now the sprite of the first enemy will be made together with player combat and the player shooting animation. For the graphics of the game we start by making the beginning of the first level by creating the tilesheets for it and creating the user interface for the player.

The current state of the project is that we are working on making the first dungeon, adding a basic weapon to the game. Besides that we have Nick working on the main ability of the game, the dash. The artists are working on artwork for the first dungeon.

With the first pass of art for the first dungeon finished and the basic shooting and aiming mechanics fixed we are now working on implementing the first dungeon while the artists work on the town and working on better looking guns than the quick L we made.

The town is progressing and the guns are being implemented now the work has started on making every gun feel different as they are all currently simple pistols.

1.4 Concept Statement

In Cepαleo the player character is transported to an unknown fantasy world, and gets told by the village elder to go to the forest to find out more about the world. Cepαleo is a 2D pixel art bullet hell shooter.

2. Target platform

2.1 Target system

The main platform we are targeting with our game is Microsoft windows, we are testing our builds on Windows 10 so that is our targeted platform. but since we aren't using any windows 10 specific features it should run on older versions of windows such as windows 7/8.

As for the specs of the computer we don't expect our game to be a very graphically demanding game, as such we expect low to mid end computers to be able to easily run **Cepαleo.**

Anyone interested in **Cepαleo** for their computer will not need to buy any special equipment besides a keyboard and mouse as **Cepαleo** will make use of both of them. **Cepαleo** will as of the current agreement in the team not support a controller at launch.

2.2 Competition

our main competition is form the games where we draw inspiration from, games such as enter the gungeon and hyper light drifter.

Enter the gungeon

Enter the gungeon is a 2d rogue-lite shooter that challenges the player with descending with the gungeon to defeat the final boss. While descending down the gungeon the player will

find all kinds of new and exciting guns to kill their enemies with. Enter the gungeon was released on 5 April 2016 and got an big update on 19 July 2018 causing a resurgence in players as it added more free content to the game. Enter the gungeon was developed by Dodge Roll and published by Devolver Digital. Enter the gungeon all time players peak was 15792 on Steam according to steamdb. it has sold between 2 million and 5 million copies on steam according to steamspy, but have to give a steamdb link as its a paid feature on steamspy, but according to playtracker its more around the 2.9 million to 3 million, steam copies.

What makes **Cepαleo** different from Enter the gungeon is that where Enter the gungeon is a rogue-lite game focused on replayability through random number generation. Where **Cepαleo** focuses more on the story it wants to tell and has combat being a medium for the player to interact with the story.

Hyper light drifter

Hyper Light Drifter is an action adventure RPG in the vein of the best 16bit classics, with modernized mechanics and designs on a much grander scale. Hyper light drifter was released on 31st of march 2016 developed and published by heart machine. hyper light drifters player peak was 4954 players five years ago, according to SteamDB. steamspy (but a steamdb link because steamspy wants money for the estimate) thinks it has sold between 500000 to 1000000 copies on steam alone. Whereas playtracker thinks it has sold around 2 million copies on steam.

2.3 Monetization

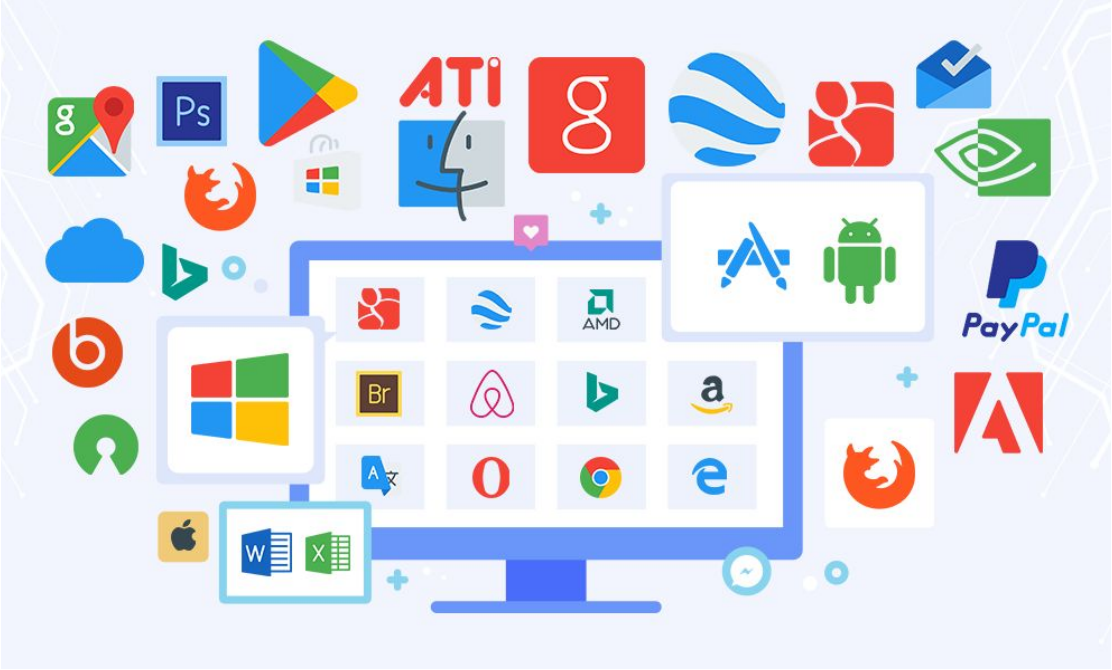
Cepαleo is being developed as a regular retail game where the player pays once for the game. Any possible DLC will be sold separately from the main game. As for our retailers we would like to target all major online retailers such as steam, epic games store and GOG. As such we will not sell any ingame items(skins for the player or guns). Our price for the game

will be around the 15 to 20 euros at launch with maybe a 10% discount on release day as that seems to drive sales up a tiny bit.

3. Development System

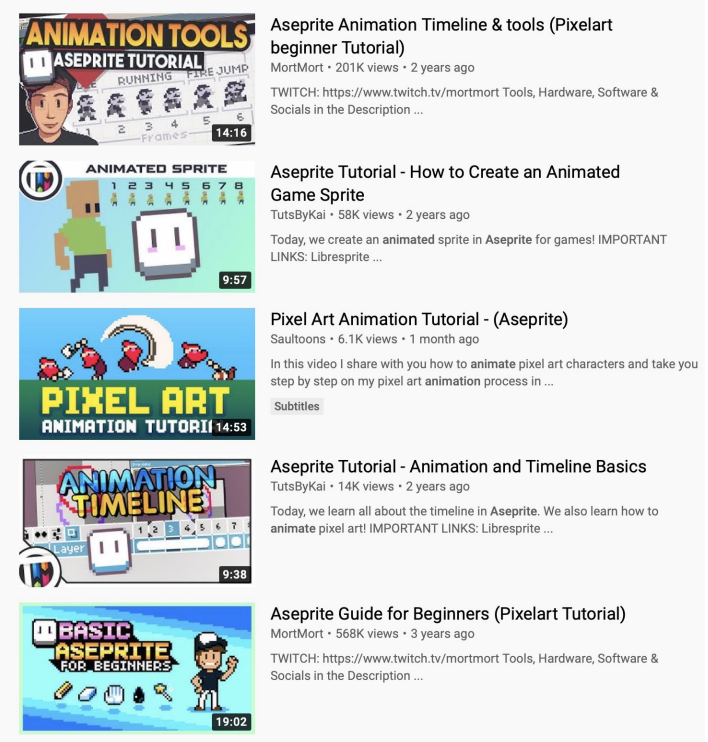
3.1 Software

During the course of the project, we will be using different software to help us through the development of our game. We use various different software for different tasks like; art development, communication, brainstorming, programming and much more. I've split this chapter into paragraphs, every paragraph talks about a different software we are using.



Our artists in this project are using multiple different software products to achieve our desired assets. The first of which is called Aseprite. Aseprite is an animated sprite editor and pixel art tool that costs $19.99. We chose this program as it's recommended online for beginners starting to do pixel art. It has a simple, intuitive interface that allows the artists to focus on what they want to create. Even though the program has a plethora of different features, they aren't obstructing the main view as they are tucked away to the side.

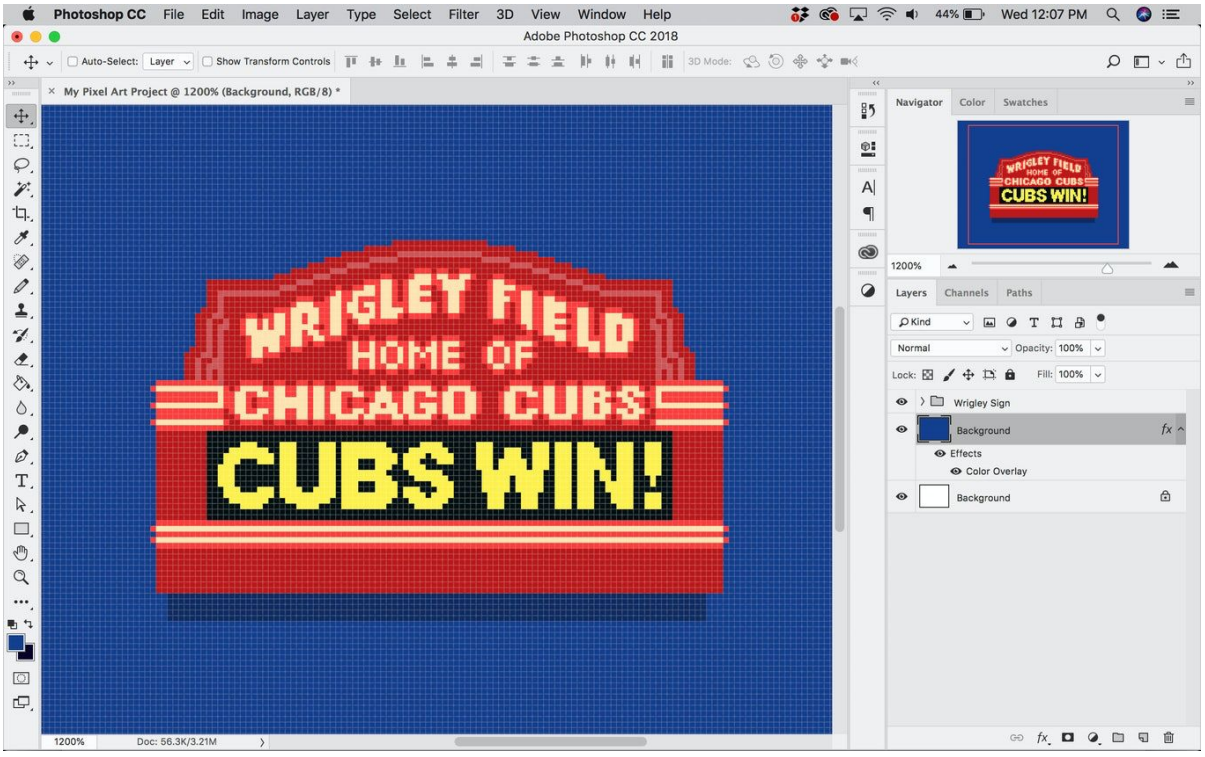
Making animations definitely has a learning curve on Aseprite, it can be still quite complicated for the average user. But since Aseprite is very popular, there are an abundance of different youtube tutorials with step-by-step instructions on how to create animations. Once we are done creating our pixel art animation, we can export the animation as a sprite sheet. This sprite sheet can then be imported into Unity and be used in our game. We have done some test trials and it has worked amazingly.



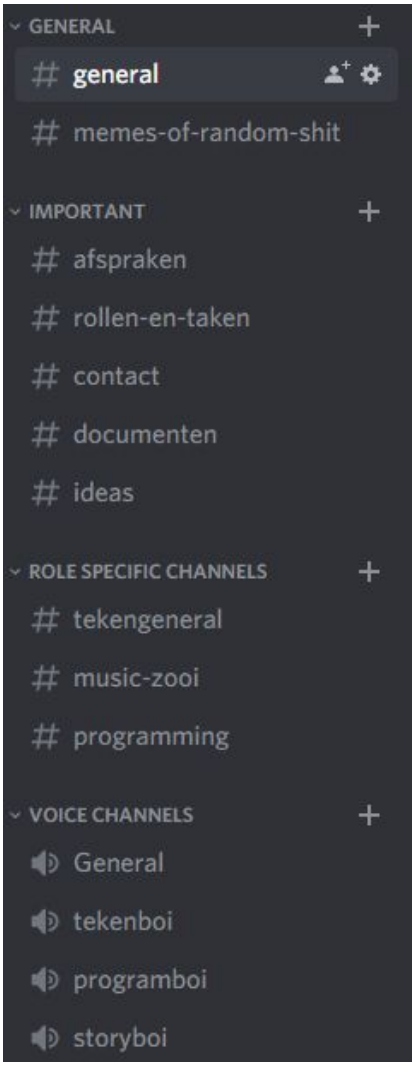
Since none of our team members came from a graphical background, this was the obvious choice to us as it was highly recommended and the mass amounts of youtube tutorials allowed us to feel protected when using it (if we didn't know anything, we could simply google it and find results). *Due to the lack of support online, doing a project with a program that is relatively new with not many tutorials is hard to do (as a beginner).*
Below is some art we have already made in Aseprite,

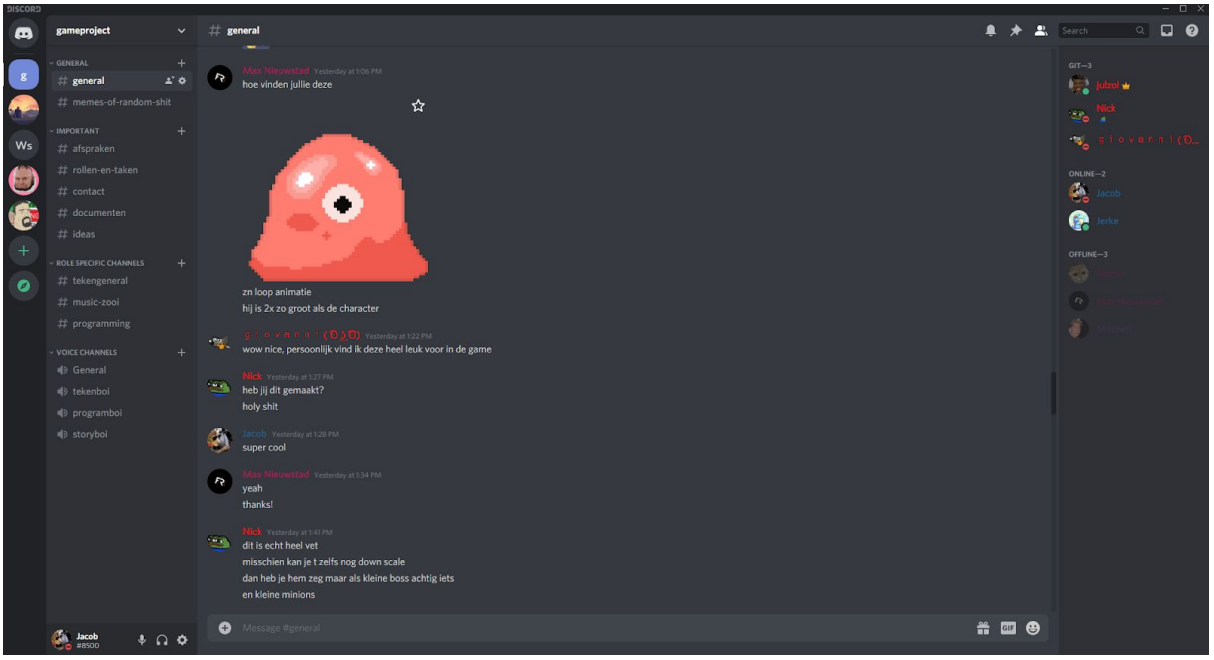
This is a sprite sheet of our main character, walking up, down, left and right. Before we finally got here, we had multiple different drafts of the main player (see below).

Another program that was used for art creation was Adobe Photoshop. Adobe Photoshop is a raster graphics editor, or in other words, software to alter a photographic image. Upon doing some research, we found out that you can use photoshop to make pixel art. Since Photoshop is pixel-based by design, it allows us to make pixel art quite easily. We used this mainly for the concept art as many members in our group are familiar with photoshop but not aseprite. It was more universal for us as we already knew how to use it and we didn't have to learn an entirely new tool. Even though aseprite was much better at creating animations, we could use photoshop to quickly make some concept art and show our designers how we wanted it to look like. This could be done in a matter of minutes compared to learning a new tool that the programmers wouldn't be using any more in the future. An example was that we made a sample "slime" in photoshop. This was completed in less than 10 minutes, screen sharing to the entire group. This was extremely beneficial as we could get opinions from our team in real time and make changes as we completed it.

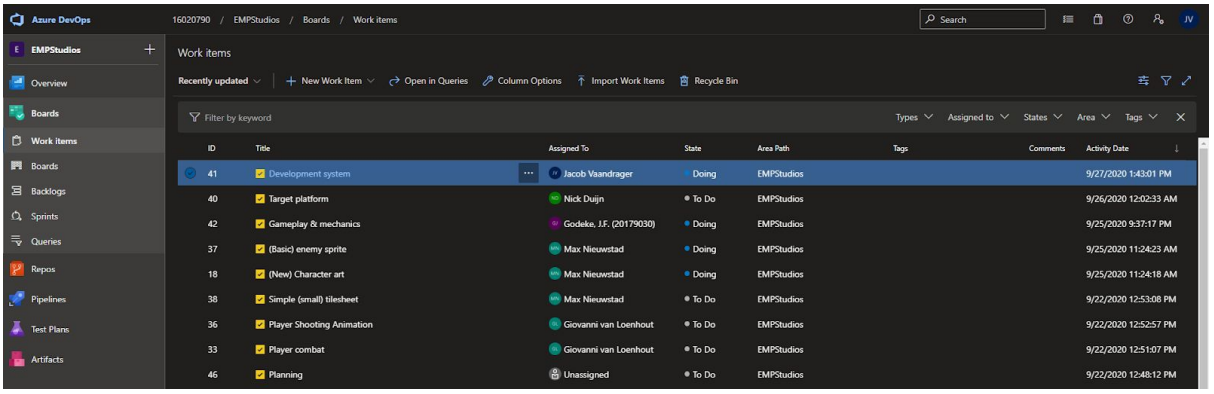


Branching on from Art software, communication between our team is crucial. If there is a question or an important error that needs to be resolved straight away, we need to be able to contact each other immediately. We are using Discord for contact. Discord is a free online messaging platform; this includes text, voice and video. As all the team members come from similar backgrounds (well educated with computers & gaming), we were all familiar with discord. We could set up our own server which all the team members were given access to. On the server, you can make individual channels, we split ours into the different tasks we had to do. This was invaluable as it meant that all the information was strategically categorized so it was easy to reference when needed. We also had different voice channels for the different tasks. This means that the artists can talk together without disturbing the story tellers or programmers. In discord you can assign roles to people, this meant that we could write “@git i need some help with..”, and this would notify all the people that had “git” as their role. This allowed us to not spam the entire group and everyone gets a notification but instead specific people for specific tasks. Below is a screenshot of our discord channel, here you can see the artist asking the group for feedback on the newest enemy he made.



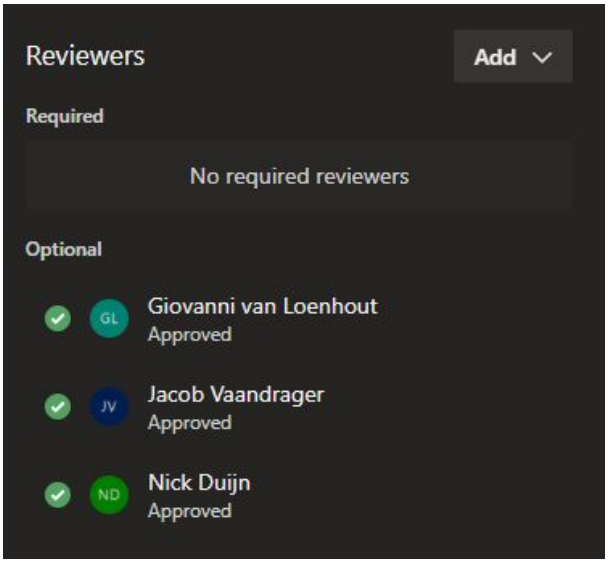


In our project we used Azure DevOps, this is a microsoft product that provides version control, reporting, requirements management, project management, automated builds, testing and release management capabilities. Azure DevOps covers the entire application lifecycle.
Most of the team members knew how to use Azure DevOps as they had used it in previous projects. It was my first time using it but the other team members gave me a tutorial on how to use it. In Azure DevOps, we could plan our sprints and what tasks we wanted to do in them. We could also assign leaders to those tasks, they were the ones in charge of it being completed. This means that someone is also responsible for a task, it ensures that the task is completed.

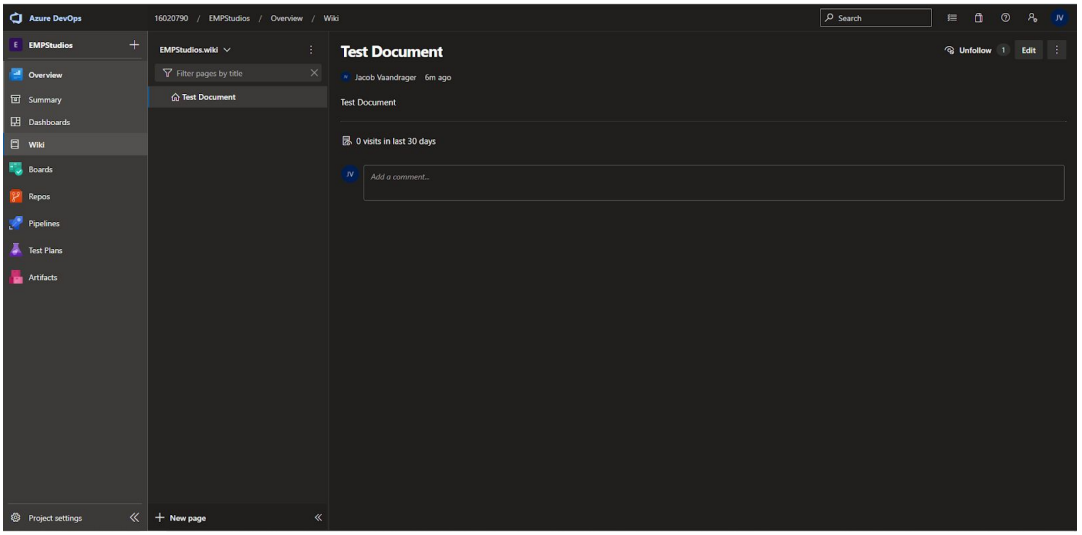


This is very efficient and ensures our tasks are completed on time. Since we are using the “Scrum” process, Azure DevOps is perfect. At the start of the week, we meet all together and plan what tasks we want to do. We can drag them/move them into the sprint for that week. This makes sure that everyone knows what they need to do.
Scrum is an agile framework for developing, delivering and sustaining complex products. It has an emphasis on software development.

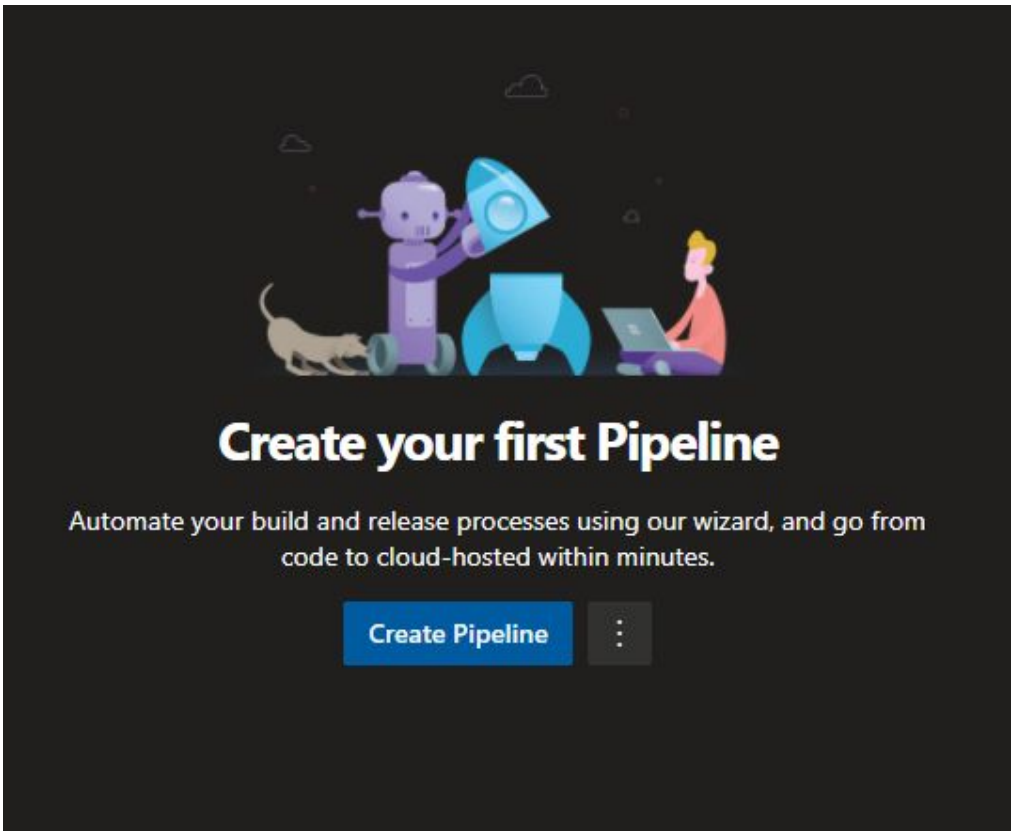
Furthermore, we set up a git repository, this allows us to streamline everything in one program. Once someone adds something to the repository, they can push their commit. Once they do this, the team members have to approve the commit to see if there isn't anything conflicting with it.



Azure Wiki is a documentation solution within Azure DevOps. It addresses our documentation needs at various stages of product development such as meeting reports or general information we want to share. It allows us to document the entire project from a single location. We could use google docs or microsoft word but then we wouldn't have everything in the same environment. It allows us to structure all our text in one single area. This means we aren't afraid of losing any files or forgetting where they are located as it's really simple to find. From Wiki, we can also export the files as .docx which is empirical when we want to export something to show to our teachers or clients. On Azure DevOps Wiki we aim to have all our text that includes, game instructions, game story, how to play, meeting notes, brainstorming sessions, review from our lecturers and peers and much more.

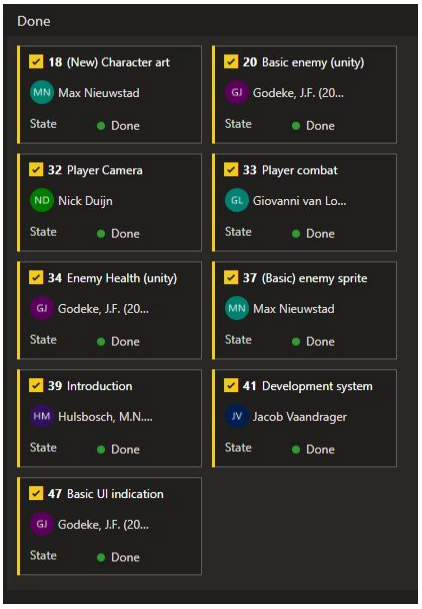


Build pipelines are a continuous integration (CI) pipeline that automates the process of building our application. We will link our bitbucket account to the pipeline (we are going to use YAML syntax pipelines). It allows us to build and compile our application using a host of pre defined build tasks. CI will automatically trigger our build and report the status on every commit, pull request or merge. Since we haven't worked with pipelines before, we watched a couple youtube tutorials and are going to integrate them into our project. Since we aren't very familiar with this topic, we can't talk about it in too much detail and will come back to this section once we know more. As of now, we haven't built our first pipeline yet.

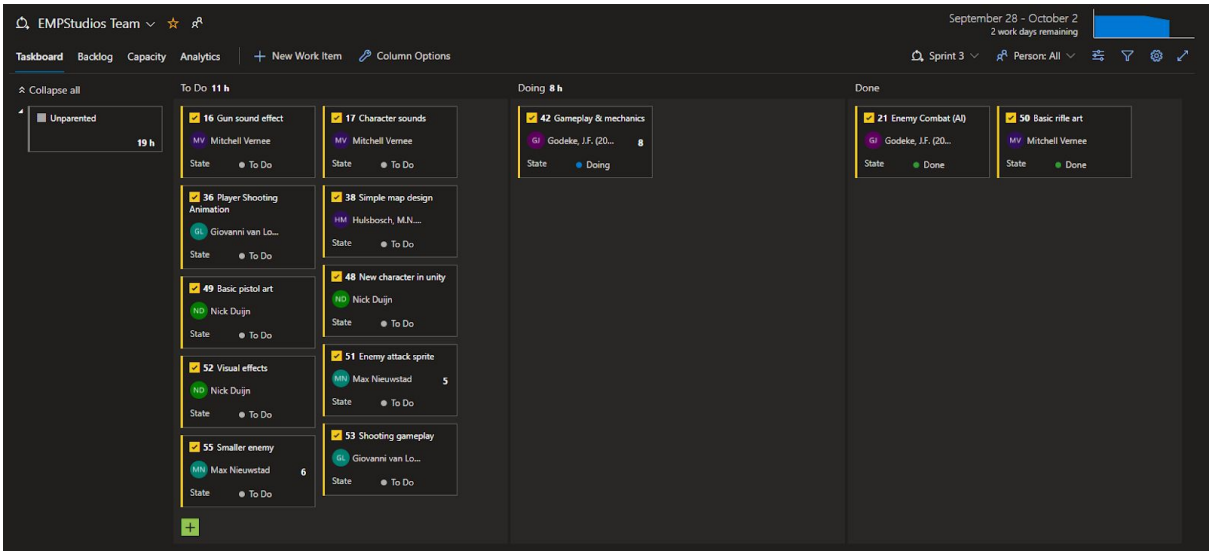


In Azure DevOps we can set up milestones and things that we want to achieve in a certain time. When we started the project, we made a list of every possible task we thought we had to do. We then grouped them under features, those features were then grouped under Epics. In every sprint, we assign tasks that we want to be completed that week. If we have a goal of finishing an epic, then we put more emphasis on the tasks in that epic then other ones. This is so we can spend more time on that epic and get it finished. Our milestones are achieved when we finish epics. Since we make our own epics, we decide when a certain milestone is hit. This is when all the tasks are finished in that epic. We make up our own milestones, we talk about it with the group and decide what are the biggest tasks and jobs that we need to do. We retain our milestones with every epic that is finished.

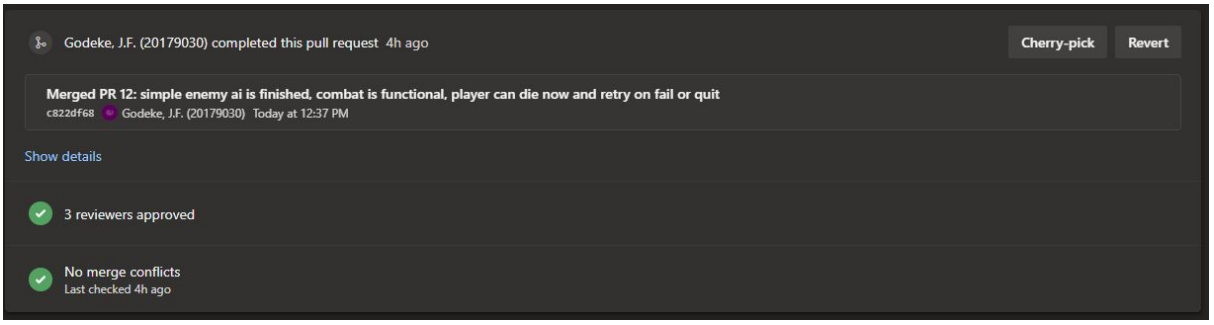
Here you can see when all the tasks are done in a sprint.



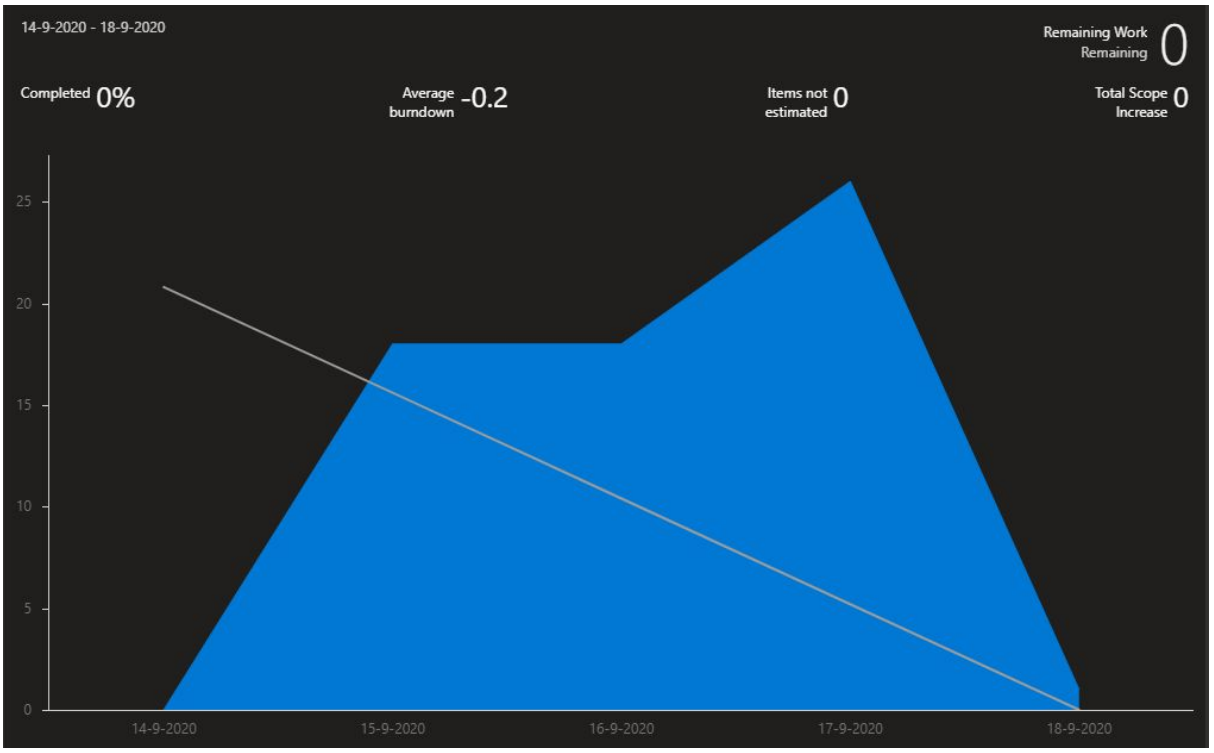
Below shows an ongoing sprint, the tasks that need to be completed, that are being worked on right now and which are done.

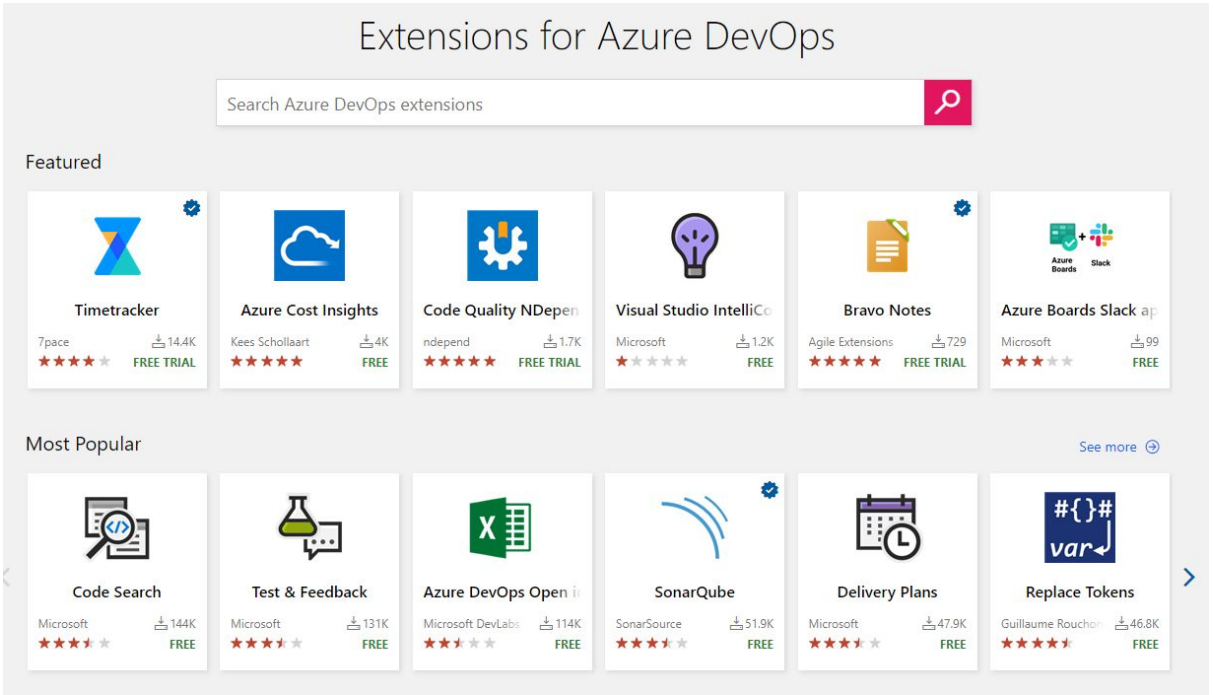


Our pull request policy will be a four eyes approach. This means something is “approved” after at least 2 human beings have given their go-ahead/approval. This means that 2 people always need to approve the pull-request before it's posted, it helps in resolving conflicts and asking the build to break. If one person doesn’t spot the error, there is a high chance that a different person may spot it. This means that we are always protected. Since it goes through 3 people, the original person requesting it and the 2 approvers, we limit the risk of error.



Azure DevOps generates weekly burn down charts. A burndown chart is a graphical representation of work left to do versus time. The work that needs to be completed on the vertical axis, with time on the horizontal axis. They are especially useful for predicting when all the work will be completed. Some advantages of using a burndown chart are that they are simple and easy to follow due to the visualizations. It reflects our achievements and what we still need to complete. It lets us know if we are on target to finish our tasks within the set deadline or not. We can also see if there are any potential bottleneck situations and problems. Some disadvantages of using a burndown chart are that it does not take in account the tasks that are still in progress. It doesn’t truly reflect how close a team is to completing their work. Nevertheless, it can provide insightful information to the team and how we are doing that week. It allows us to self-organise, self-manage and direct ourselves to more success.





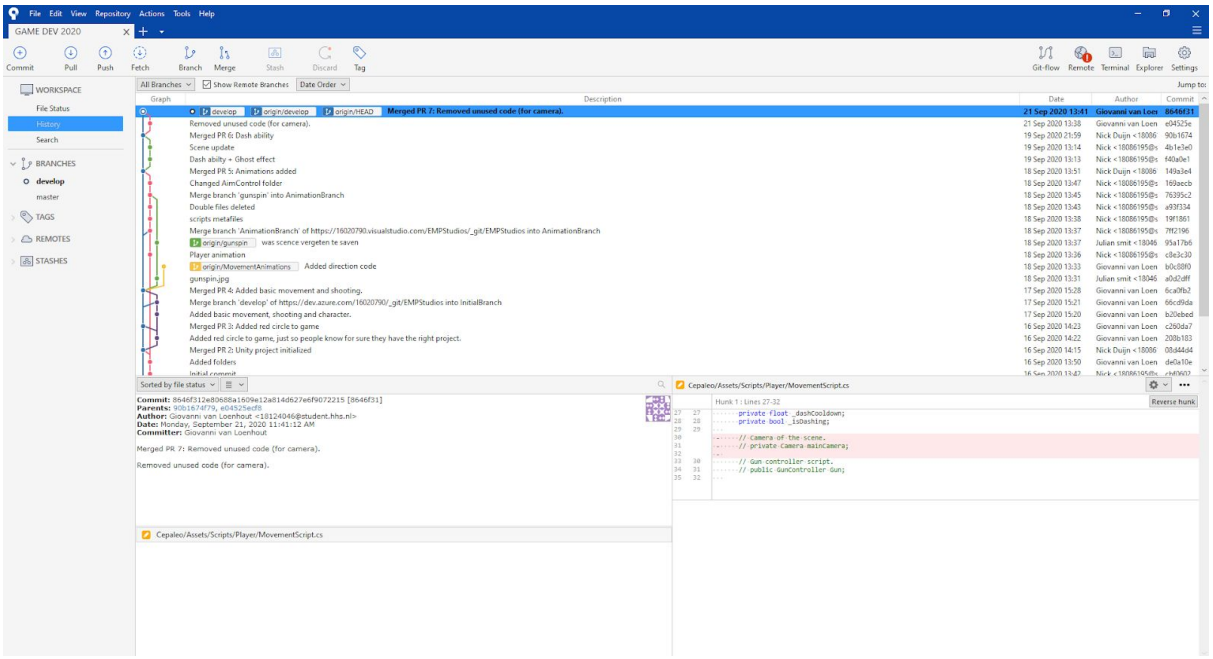
What didn't go well

Add new feedback

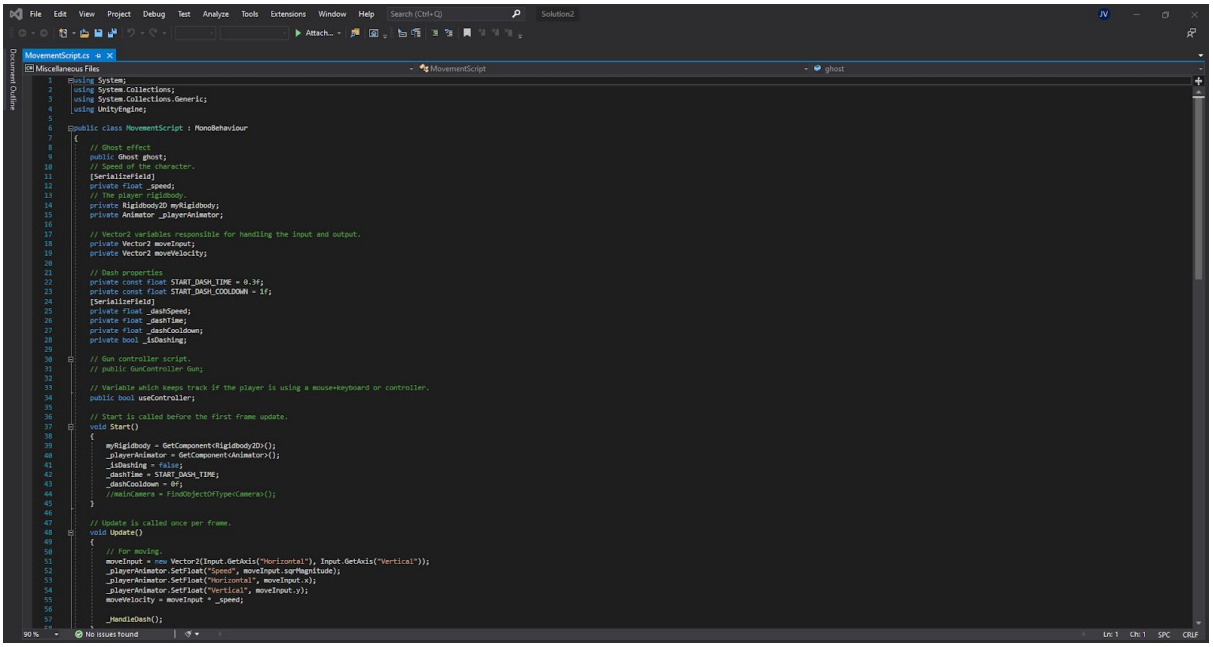
Artwork was harder than expected

Azure DevOps has an online marketplace for extensions. Recently one of our team members found the extension for retrospectives. A retrospective is generally a look back at events that took place that week. We talk about what worked, what didn't and what could've been done better. We meet every monday and do a retrospective together, this is incredibly valuable as sometimes we don't have enough time in the week to talk about what isn't going well etc. By scheduling it every monday, we are forced to sit down together and complete this. We see what worked well so we can continue to do that, what didn't work well so we can avoid that for the upcoming week and what we could've done better. It also allows us to get opinions from the team members and help resolve issues. For example, in week 2 (see picture) we had a list of things that didn't go well. This means that we will fix them for this week and ensure that a burn down chart is generated, assign more help for the artists and update the tasks more frequently. Overall, it's a valuable hour out of the week, we can talk about our issues and improve them.

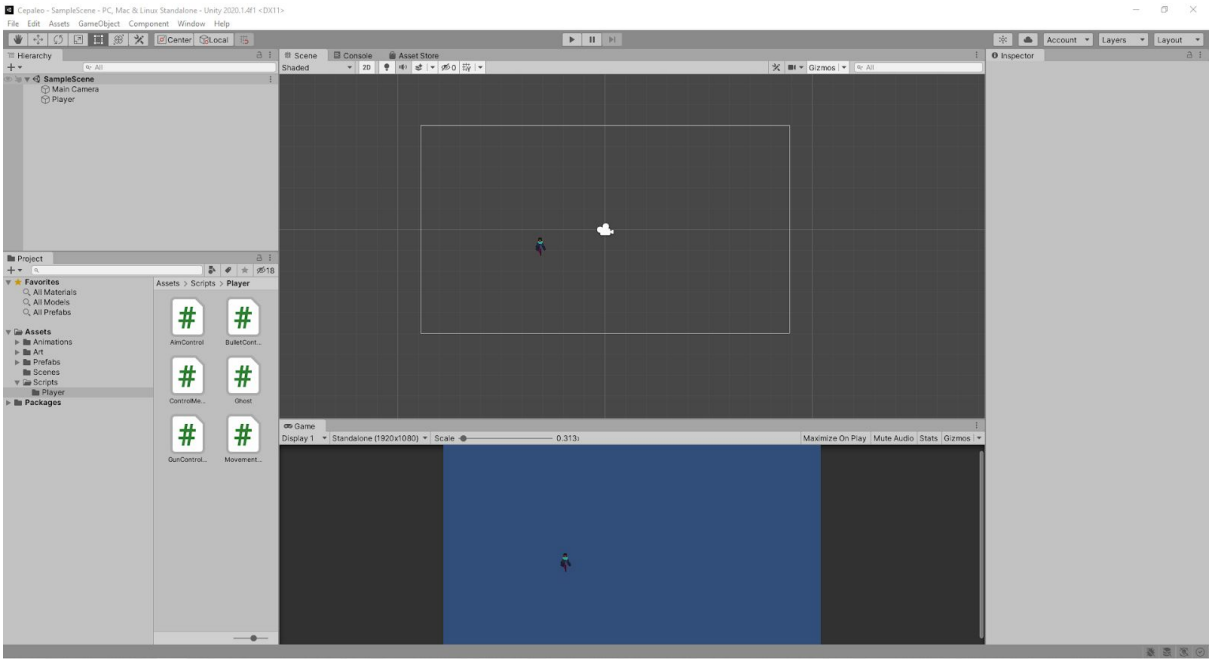
Linking to Azure DevOps, we also used software called Sourcetree. SourceTree is a free Git desktop client for developers on windows. Since not all the team members have worked with Git, sourcetree was useful as it simplified everything graphically for us. It has colour coded branches so you can see quickly what tasks are being worked on, which are done or need to be completed. It also shows the number of commits that are ahead and behind the remote branch, which is really handy for beginners. It also automatically detects conflicts which you can then resolve. It has a simple user interface but still has all the advanced git features any git vetern could want to use. Making it perfect for our team of experienced to non-experienced git users. Also, the simple user interface allows you to commit, push, pull and merge with only one click.



To write our scripts in C#, we opted for visual studio 2019. Visual studio is an integrated development environment. Since visual studio has been around for so long, the online community presence is ginormous. This means that if we ever run into errors, we can easily search for solutions online. The software may seem confusing to beginners but since we have all worked with visual studio before, it was quite the obvious choice as we have all been taught how to use it. Visual studio is receiving constant updates from microsoft so we know there is little chance that it will ever become unsupported. In addition, as a bonus, the visual studio IDE connects flawlessly to Azure DevOps. Below is a screenshot of one of our scripts in visual studio.

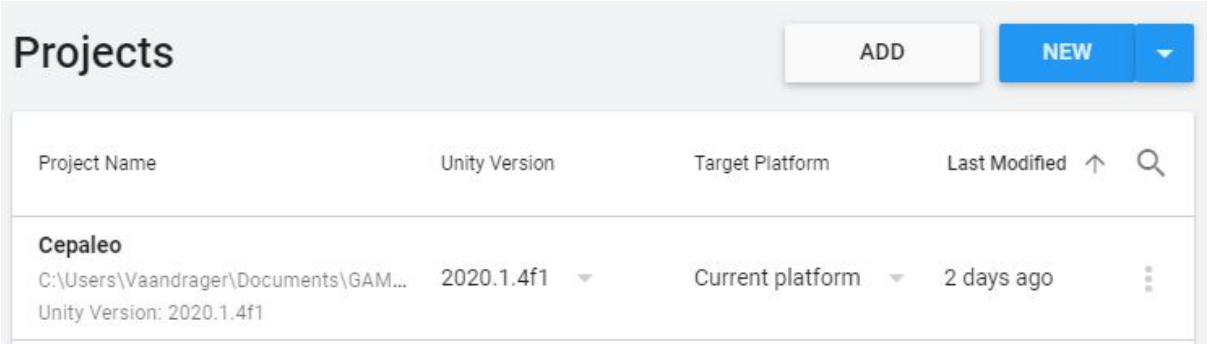


Our chosen game engine was Unity, Unity is a cross-platform game engine developed by Unity Technologies. It supports over 25 different platforms and it's scripting is done in C#. We had the option of picking what game engine we wanted to use, but most of the team members in our group had already developed games using the unity engine, this meant we were much more comfortable with it then other game engines. The CEO of Unity has also stated that half of all games are built on Unity, that is a crazy statistic and furthers our interest into Unity.



Since we had a github educational account, we were able to get access to Unity Pro for free. Having Unity pro means we don't have to use the Unity splash screen but we can use our own one. We can get detailed analytics once we distribute the game; we have the option of using dark mode with Unity and we can use the Unity beta builds.

Once we decided to use Unity to develop our game, we had to choose which version of unity we wanted to develop our game on. We didn't want to develop it on any outdated version or any beta ones. We ended up choosing Unity version 2020.1.4f1. We chose this after we spoke to our end boss and he advised that we should use that one.

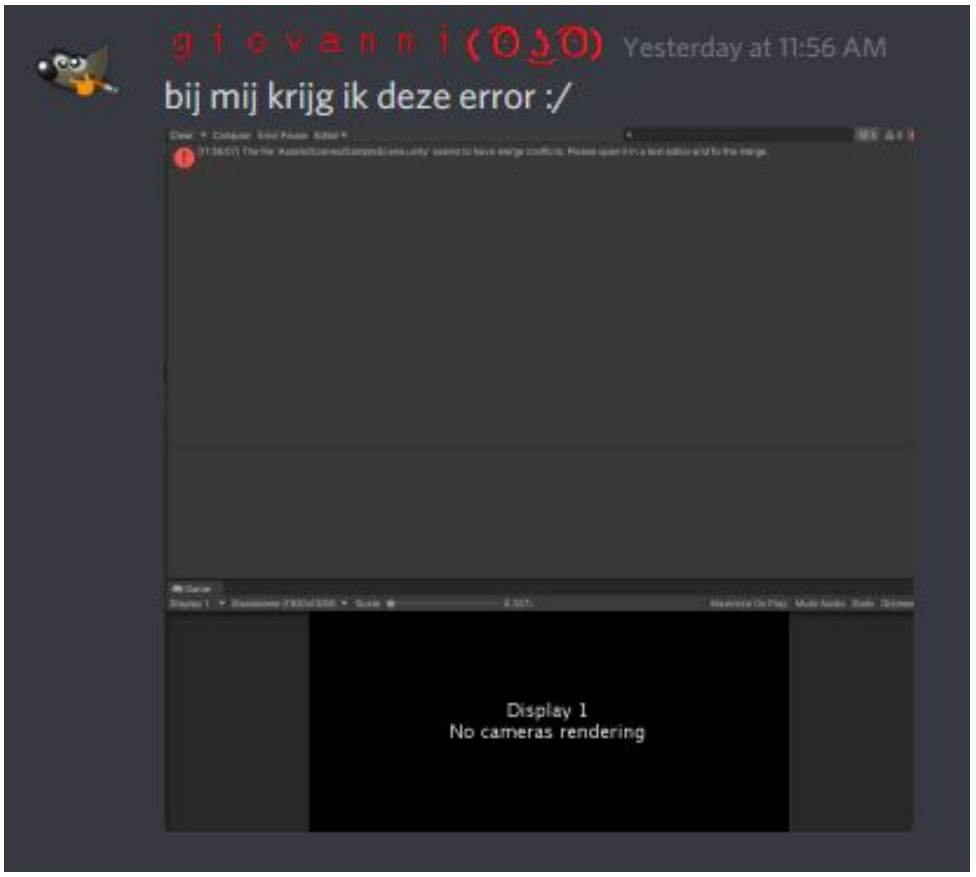


for the music in the game we will be using ableton live as our digital audio station. as for the instruments we will be using micro korg, behringer crave and elektron analog rhythm as our instruments.

3.2 Testing

We are going to be testing our game throughout the development process. There are many different ways of testing and when to do it.

First of all, we are going to do team testing, this means testing with the people in our team. This is useful as we can get quick opinions and fix issues that are urgent. Our team already knows the product and how it is meant to be used so if it's different we know straight away. We are going to be doing this throughout the project, and whenever, someone has an issue they want to fix. Such as a camera movement issue, we can all get onto discord and try solve the issue together. We can test it out and find a solution.



Testing can also be done on friends and family. We will do this later in the project at around the week 10 mark. This is when we will already have a clear test build for the game that we can show. Since our friends and family haven't played the game before they will play it in a different way the team was playing it. It will show us different options and if there are any errors we had not thought of/seen. It's very valuable to get an outside opinion as we spend a long time on the game and we play it in a certain way. Different users will play the game in their own way; for example, your mum may take route a then route c then route b but your brother may follow the order you thought of and take route a, b and then c. Both being correct but you only thought of one way to do it.

When this is done, we would like to send builds to our fellow peers. Since they are also game developers, they know what to watch out for and what errors may arise. Our peers are going to be critical whereas friends and family may not. They may be scared of hurting our feelings as they know we have spent a long time developing the game. Peers are more direct, and will have more constructive feedback.

We are still going to use different testing methods as we go through the development process. Right now, we are at the beginning, so these are the tests that we have already put forward.

4. Gameplay and mechanics

4.1 Concept

4.1.1 Genre

Cepaleo is a bullet-hell platformer with puzzle elements.

4.2 Story

When you come to, you wake up in an unknown room. Actually almost everything seems unknown to you, not even your name. You decide to leave the room and find yourself in a town. It's a small town with a few friendly looking inhabitants. An old man sees you and approaches you, he tells you that they found you unconscious and helped you out. He also explains that weird things have happened, for example multiple structures have appeared and the wildlife has become more aggressive. He also tells you about a rumor about a ghost appearing in the “Forest of Denial” that appeared at the same time as you did and that the ghost might be linked to your memory.

Before you leave, the old man gives you a weapon, a simple pistol, to defend yourself and an option to train a bit before entering the forest. To get to the forest you have to cross a ravine/river by crossing an old rope bridge. while doing this you see the thick dark forest ahead of you. While approaching you see a figure entering the “Forest of Denial”.

The figure that entered before you leads you to the middle of the forest where you are greeted by multiple monsters. You follow it to an open area inhabited by a large creature that attacks you. Defeating this monster will lead to a new area that eventually leads to an even bigger, more vicious creature. Defeating this grants you some of your memories back and the figure you followed transforms into the ghost of a girl that seems somewhat familiar to you.

He follows the ghost through the other areas, trying to talk to the ghost, recovering parts of his memory along the way.

The last of these places seem different, as if you are being watched while at the same time being the one watching. This feeling turns out to be true, at some point a man steps out from the shadow/ out of a mirror. This man looks exactly like you. confused you start to talk to him, and after a while you get the feeling that he is the one responsible for all of this. With no other option you decide to attack the man. Once you have destroyed the building/area you're in, collapses from the destruction and everything fades to white.

You wake up in your apartment, still littered with the booze from, well ever since the accident. You feel different from all those other days, you feel fresh. Looking at the picture of your daughter you feel a new strength, the strength to start living again.

Second Draft:

Travelling on your way back to your family from your latest adventure, you pass a small village. It's a town with a few friendly looking inhabitants. You approach and individually talk to them. They say that they have recently been plagued by dangerous monsters and are scared to leave the village. The villagers main source of income is mining in the dungeon. They use the ore obtained to trade with other villages for necessities such as food and clothing. The monsters have taken the mine hostage and any villagers that have attempted to enter the dungeon have been killed. The village elder tells you that the mine is on the NW side of the village. If the villagers cannot enter the mine soon, they will run out of food. Your adventurer instinct kicks in and you make your way to the mine. Hoping to save the village and restore the peace.

4.3 Game structure

There will be one city that acts as the main hub. The player saves his game and can heal himself at the hub. From the main hub the player will be able to go to the dungeon where he can defeat the slime, once he does defeat the slime, he will have beaten the game.

4.4 Players

The game has been designed as a single player game as such there is only one main character, the Cepαleo-man. the Cepαleo-man will find himself in an unknown world not recognizing where he is. So he talks to the village elder on his quest to uncover the story.

4.5 Actions

As with most games, Cepαleo has different actions which can be used throughout the game.

The most important action the player can use in the game is the movement. The player can walk the 4 cardinal directions (left, right, up and down) and the 4 directions in between. The player moves the Cepαleo-man around using the WASD-keys on their keyboard. By pressing down two of the movement keys.

Another way the player can move around is using the Dash ability. By pressing the Shift key, the Cepαleo-man dashes whatever direction you're walking in, to give the player a sense of speed in the world.

The second most important part of Cepαleo is the gunplay. The player has a crosshair visible on the screen - so they know what they're aiming at - and using the mouse the player can aim on the screen. By pressing the left button, the player is able to shoot their weapon. The game features automatic and semi-automatic weapons. The former allows you to hold down the mouse button to fire, while with the latter you have to press the mouse button multiple times to get fire multiple times.

With the Q key on the player's keyboard, they can switch to the different weapons. A very handy feature when in a gun fight and pressing R - to reload - is not an option.

Certain places in the dungeon require the player to press buttons to solve puzzles. These buttons can be pressed using the letter E on the keyboard, which is indicated on the screen.

4.6 Objective

The objective for Cepαleo is to find and defeat the boss in the spooky forest. As this has been overrun with slimes hinting Cepαleo-man that there has to be a bigger slime around and to destroy the colony of slimes he has to kill its leader.

4.7 Controls

The game will be played with a keyboard and mouse. For movement the “WASD” keys are used, W for forward, A for left, S for backward and D to go right. The player is able to use a dash either by using the shift button or with the scroll wheel if the player has access.

To switch weapons in game, the player needs to use the ‘Q’ button. To shoot the weapons the player aims a crosshair with the mouse and uses the left mouse button to shoot. To reload the weapon the player uses the ‘R’ key.

4.8 Object types

4.8.1 Weapons

The player will mostly find new weapons in his adventure(this feature is scrapped because of time restraints, the player spawns with 3 guns, shotgun,pistol and uzi). These weapons are multiple types of guns. There are pistols, rifles and shotguns. The player can carry up to 3 weapons at once. Each weapon will have an amount of bullets in it that can be shot before it has to be reloaded. This is to have more control over the balance of the weapons and it adds to the strategies the player can use throughout the game.

Though there is a reload mechanic, your ammo will be limited but there is no limit to the amount you can carry around with you. We decided on this because it would add a challenge to the game as some guns are way more powerful than others. To get more ammo the player will find ammo boxes in the dungeon giving him more ammo for all the guns he is carrying at that moment.

Your first weapon will be a pistol. It’s a very basic weapon that can deal a low amount of damage to enemies minimum damage. It has a fast rate of fire but it's limited by the player's ability to click fast and still be accurate with his aim. The pistol will have the most ammo at the start of the game because it's the weakest one in the game. We had planned more versions of the pistol but once again because of time restraints, the only version in the game is the basic version. The pistol has a magazine of 15 bullets that can be fired in rapid succession.

The other variants of this gun were planned to have less ammo in the magazine as they were more powerful than the starting one. The speed variant would have shot two bullets per click that dealt more damage than the starter one. but in exchange had a lower amount of bullets in the magazine(12) and because of its burst fire and lower magazine it would force the player to reload more often than the base variant.

The second pistol variant would be more of a powerful variant kind of like a deagle in counter strike where the ammo capacity would be way lower(6) but in exchange for this it would deal a lot more damage and have a lower fire rate than the base pistol. This design would reward the player that aims his shots with a higher per shot damage than the base pistol.

The next class of weapons is rifles, the rifles will be faster firing and fully automatic unlike the pistols. Rifles will also have more ammo per magazine then pistols(20 base rifle uzi). Because the weapon is automatic the player will not have to click per shot. the base rifle will do more damage than the base pistol, but the player will start off with less bullets for the uzi than the pistol forcing the player to make a choice when to use it.

Just like with pistols there were plans for multiple versions of the rifle. The first one would have been a more powerful assault rifle that would have fired slower, and with prolonged firing would have started to have bullet deviation. Besides that it would have had a bigger magazine. The rifles are designed as a jack of all traits kind of weapon. The weapons would never be bad to use, but because of the low amount of ammo the player will start with it will be hard to use as your primary weapon while playing the game.

The last weapon category is the shotguns. The starting shotgun has 6 rounds in the tube. And just like the rifles will start with a low amount of ammo at the start of the game. Shotguns will be shooting multiple projectiles per shot making it a great weapon for multiple targets. The shotgun will reload with a “magazine” style system as such it wont load a single shell at a time but instead will just refill the tube fully during the reload. Because we chose this style of reload you can't interrupt it during the reload.

Another variant of the shotgun would have been the double barrel shotgun which has only two rounds but in exchange for that would have dealt a lot more damage per pellet and would have shot a lot more pallets per shot. Like pistols the shotguns are semi automatic meaning the player would have to click for every shot leaving the gun.

4.10 Graphics

4.10.1 User interface

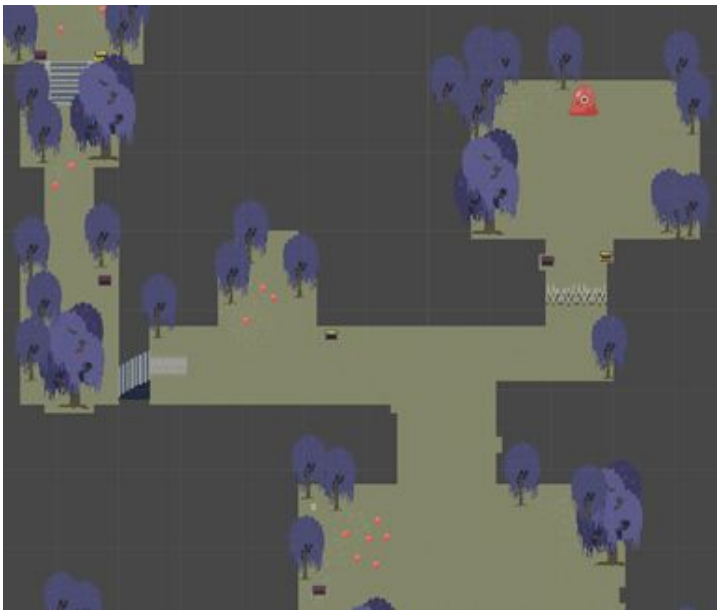
The game has a minimalistic hud only showing the player what they really need to play the game suchs as their health and the ammo in their gun and the ammo they have spare for that gun. As there are no complex inventory systems the player doesn't have to deal with inventory screens or radial menus.

As seen in the picture above the health of the player is shown in the top left corner of the screen. When the player takes damage this bar will decrease according to the health he has left. in the circle in the middle of the screen the healthbox is shown this can be used to get the health back that the player has lost.

In the bottom left of the screen the player can see what weapon he has equipped and how much ammo there is inside the gun and how much ammo he has spare in his backpack. And

the yellow box contains ammo for every gun the player carries, when the player collides with the box.

4.10.2 Landscape



The first dungeon wants to create that corrupted forest feeling. Because of this we used a lot of muted tones and such to really drive home that the forest is feeling off. As the player explores the first dungeon/the creepy forest it will become clear that there was once an ancient civilization living there as their moving platforms and spike walls are still there. It seems like some ancient force has reset these machines and will prevent the player from progressing without turning them off, or getting the platforms to move to new locations. The dungeon has this darkened theme to it, giving an off feeling to it.

As seen the village is an older style city with a lot of wooden buildings and with the town square covered with cobblestone on which the player will meet with the elder who will give him his quest.

4.10.3 Objects

Spikes can be found in the dungeon. These spikes have been in place for a long time, but have always been lowered, to allow people to move over them. The player can lower these spikes, with the help of different switches/buttons in the world, to allow them to traverse to different parts of the dungeon.

The above image shows the spikes in its different stages, with the left side of the image being the fully extended spikes, the right showing the spikes lowered down and the images between representing the animation of the spikes.

4.10.4 NPC’s

The world the player will find himself in is not only inhabited by hostiles but also by friendly townsfolk. Most townsfolk won't talk to the weird stranger in their town. As he looks very different and also has these weird objects with him. The Townsfolk wear their blue robes.

The leader of the town wears brown robes with a red headdress and has a magical staff on himself but because of the corruption of the forest the staff is losing power.

(older slime drawing than in the game)

Inside the first dungeon the player will encounter a lot of slimes. These mindless beings attack anyone that comes close to it. But most of the time if there are small slime there is a bigger one controlling the colony.

4.11 Sound and music

This music accompanies the story, so the level and the music make sense together.

Dreamy fountain is our main hub tune. This song is created with the following terms in mind: dreamy, in search of answers, mysterious.

The character walks around the village with the fountain in search of answers to his questions. *Who is he? Where is he?*

Fat jiggly bastard is our boss tune. This song is created with the following concepts in mind: bombastic, explosive and ‘Fat and Bouncy’.

You enter a large area and see something on the forest floor. As you walk towards the artefact you feel the earth shaking. The sound gets louder and the shaking heavier. A huge slime appears on your screen ready for battle.

Funky forest is the track you’ll hear in the dungeon we’ll be demo-ing

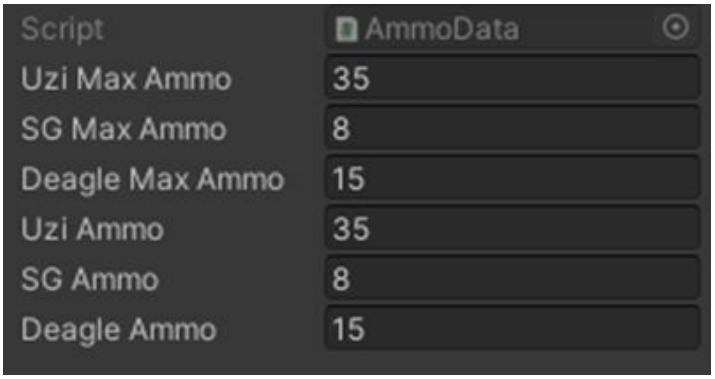
This song is created with the following terms in mind

Upbeat - Battle music - Dreamy forest - wind through the trees - falling leaves - mystic

When you walk into the first dungeon the screen goes black. Slowly the beat starts playing and the dungeon name appears . The blackness slowly fades and shows you the bridge towards the forest(beat starts pumping). You encounter the first enemy (level flows on the beat just as the monsters)

4.12 Data storage

Cepaleo has different parts of the game that need to be persistent between dungeons and playthroughs, so ensure parts of their playthrough remain consistent. As a game with an emphasis on its guns and gunplay, it’s important to retain information based on how much the player has used their ammunition. The player’s leftover ammunition from the three different guns is saved in a scriptable object.



The player’s health is also saved in a similar scriptable object, so the health is saved between the different levels.

5. Marketing

We will market our game using trailers to build up hype for our game. We will use platforms as YouTube and our own website. Every week we will release a update for our game and the patch notes will be posted in the form of a blog post

5.1 Target audience

our target audience is made up out of pixel art fans and bullet hell fanatics. Both of which are looking for a challenging new experience in the form of Cepαleo. Besides our core audience we would love if the indie game lovers would give Cepαleo a try.

5.2 Concept paragraph and unique selling points

In Cepαleo the player character is transported to an unknown fantasy world, and gets told by the village elder to go to the forest to find out more about the world. Cepαleo is a 2d pixel art bullet hell shooter.

Unique selling points:

- One of Cepαleo unique selling points is it's one of a kind movement system.
- Another one is Cepαleo's original pixel art style.
- Cepαleo's banging soundtrack made by Mitchell

6. Ideas and expansions

Non-MVP feature ideas for future expansion are kept here.

More dungeons besides the first dungeon(first we planned to make multiple but after making the first one we rolled the first and second dungeon into one dungeon to get to product out in time)
More guns than the initial 3 guns the game releases with(the art work for different guns is there but we never implemented them into the game).
More powers besides the dodge power the player starts with.
More soundtracks and sounds for objects.

7. Team

Nick van Duijn: programmer and storyteller
Mitchell Vernee: music/audio and artist
Jerke Godeke: programmer and storyteller
Julian Smit: programmer
Martin Hulsbosch: artist and programmer
Giovanni van Loenhout: programmer
Jacob Vaandrager: programmer and artist
Max Nieuwstad: artist and programmer