| **Evidence template:**  AS91897 - Use advanced processes to develop a digital technologies outcome |
| --- |

| **1. Getting started** |
| --- |

**What am I making? What is its purpose? Who are my target users?**

I am making a level based ‘platformer’ game on Unity with in-air movement. If the character touches a tile, the level will restart. There will be different levels and the goal of the game is to reach the next level by navigating the level you are on to get to the next level. I want the game to be harder than average so my target users are people that have played games before and can be precise in their timings. It should be available for most ages (5+), but not designed for any one age group.

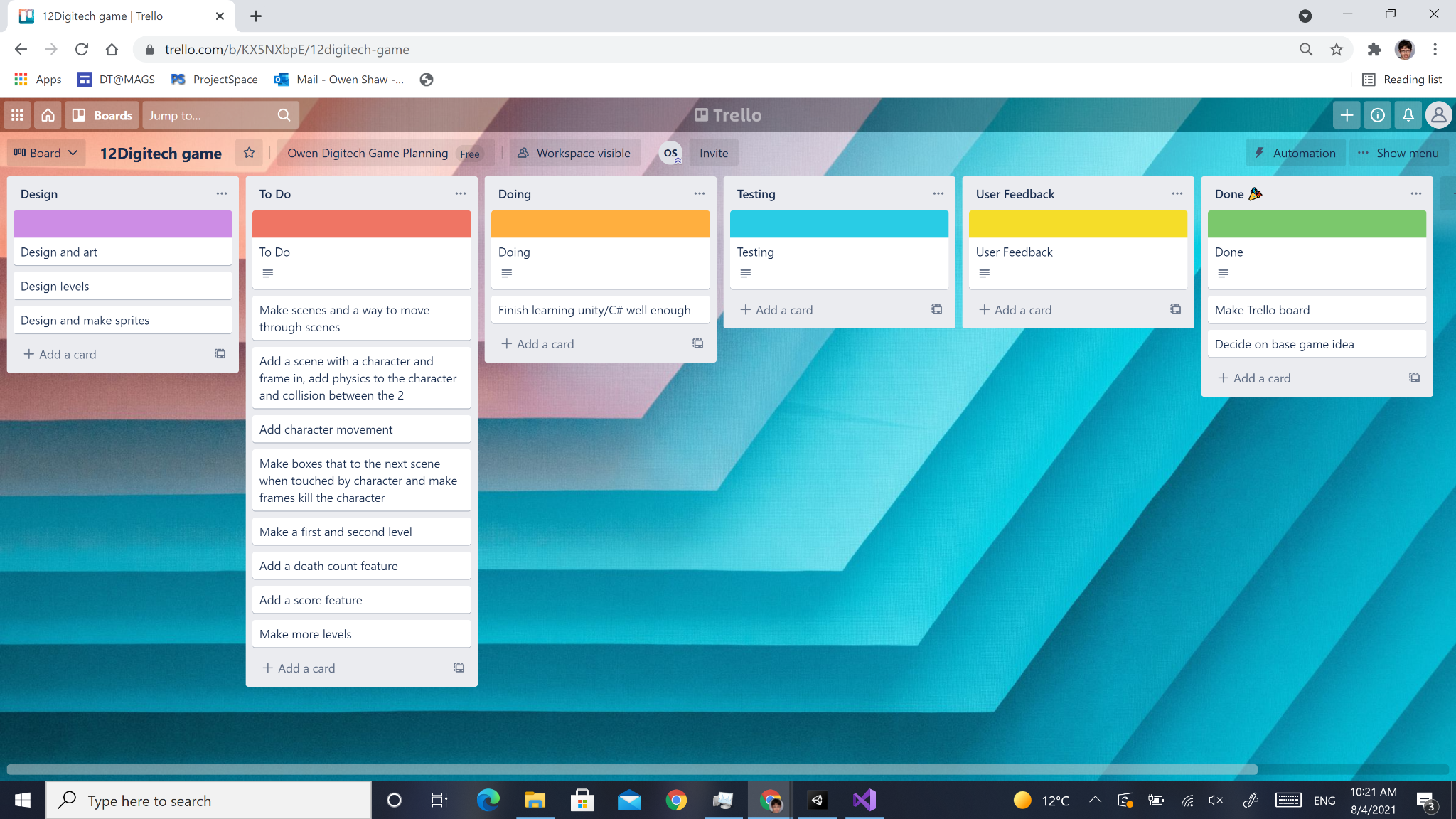
**What are the requirements of my outcome?**

* Have multiple levels
* Have in-air movement
* Have a platformer feel but in the air
* Have a way for character to die
* Have a way to move between levels

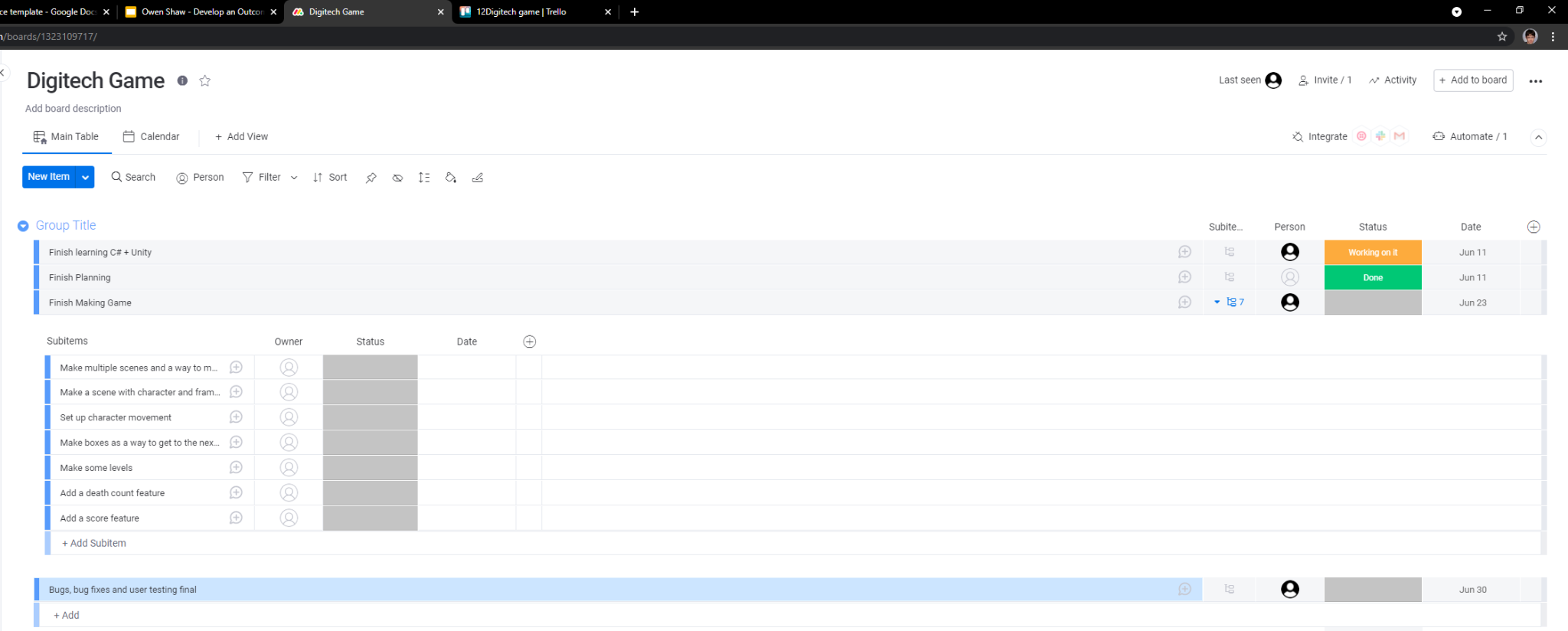
**What project management and version control tools will I be using?**

I used a combination of waterfall and agile project management techniques. I would have just used agile but for my program, most things needed to be done in order, so waterfall fit my program a lot better. Agile is used in that some of the aspects of the game can be interchanged and that the development is adaptable in the section with aspects of the game. Some of them are interchangeable eg. death counter and extra features, and any of them can be edited individually most of the time. The game can also be ‘finished’ early if need be, aspects 5 and over can be ignored if needed (due to me being sick, this came in useful as I was only able to add one extra feature - score, when I would have liked to add more. I was able to finish the program well enough without those extra features). Testing and user feedback after each section is another feature of agile development, which I have done. Despite this, due to the nature of the program, most of it still needed to be done linearly as many later features rely on earlier features being complete in some way first. I have used a mix of the 2 instead of just waterfall because waterfall is not flexible at all. If I want to change the program at a later point then it would have been a lot more work. I used Trello to manage my project as it fit well with this style of separating the program into different components and different components having small development cycles of their own.

[**https://trello.com/invite/b/KX5NXbpE/fd197fc2d8ae98807c3dcf79d71f5744/12digitech-game**](https://trello.com/invite/b/KX5NXbpE/fd197fc2d8ae98807c3dcf79d71f5744/12digitech-game)

****

I could have also used [monday.com](https://monday.com/), but I decided not to because it follows a more linear planning style with less flexibility.



For version control I only really had 1 option, saving different copies of files of my game at different stages of development. I put the files onto google drive to make sure that I would still have copies if something happened to my computer. I could have also saved files onto github, but learning how to use github would take time out of developing the game, so I decided to just use drive instead as there is no real benefit to using either one in this case. I am not going to be looking back on this project in the long term future.

**General Information**

I learnt C# and Unity from a course on Udemy ([here](https://www.udemy.com/course/unitycourse/)) with Ben Tristem and Rick Davidson as instructors. Most techniques used in the program were learnt from this course (that I wasn’t able to finish due to being sick, but got far enough through and learnt enough to make a game myself).

Feedback was given by my family members (Brett Shaw, Tessa Phillips, Thomas Shaw) and some friends (Ethan Chester and Anand Krishnamurthi). Feedback was taken at all stages highlighted in red and the game was adjusted according to feedback as best as possible.

At the user testing stages, my brother (Thomas Shaw) and friends mentioned earlier looked for bugs that I couldn’t find and told me about them.

To share the game across multiple computers, the game had to be built and sent only in game form. Code could be put into a notepad or a document and sent that way. Without Unity installed, it would be very difficult for other computers to see what was going on in the engine, so that part couldn’t be shared across to different computers without Unity.

The Trello Board could also be shared across different computers if needed.

**Link to full evidence file and game:**

<https://drive.google.com/drive/u/0/folders/1Q2jW1mNSJEGNNPavhYvNLIgkR2WIKhX1>

**Decomposed list of the components of my project:**

* Having multiple scenes and a way to move through scenes
* A scene with character and frame in, collision between the 2 and physics on character
* Character movement
* Have boxes as a way to get to the next scene and have frames kill the character
* First and second level and explain how to play the game
* Death count feature where total deaths are added together
* Score feature in some form
* Extra levels

| **Component:** Having multiple scenes and a way to move through scenes | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | Either have scenes on one ‘scene’ on unity and move the camera to each one on trigger or have each scene on a different unity ‘scene’ and switch between the unity scenes on trigger. I chose the second option as having every scene on one ‘scene’ would drastically decrease the performance of the game if too many scenes are added. As I want the game to be able to be expanded upon in the future, it would be much better to have separate scenes on separate unity ‘scenes’. (‘scenes’ refers to unity scenes while scenes without the quotation marks refers to the visual scenes).  Either have buttons or use keys to switch scenes. I chose to use buttons as keys will be used for movement as well which may confuse the user. | | |
| **My testing and trialling** | Debugging/testing: The code here is relatively simple and worked first try.  Code validation: Unity will not let the game run if there are any compile errors within the code, so code will always be validated. (same for all components)  User feedback/testing: At this stage, users found arrow keys to be nicer to use than buttons, but this is most likely only because there was no gameplay involved yet. More feedback will be given on this later when the game is more fleshed out.  Evaluating against requirements: Make sure to have the way to move through scenes be easy to use, which I have done. | What did I do as a result of this?  Note that I will need to gather more feedback about this later when the game is more fleshed out. | |
| Game Evidence:  <https://drive.google.com/drive/folders/1U17xUCeAZcGXOA0oYNV1qfU12h7RmesB>  Project Management tools at this point:  [Trello board](https://drive.google.com/drive/u/0/folders/13gYIwo4aYQLqfBKflssscKi-9DvvtA7s)  (Game evidence included alternative trials for components) | | | |
|

| **Component:** A scene with character and frame in, collision between the 2 and physics on character | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | There is only really one feasible way to do this for me - use the unity physics engine, but technically making my own physics engine would also be an option. I cannot trial this though and chose to use the unity physics engine because I am not adept enough with unity and C# to make my own physics engine. | | |
| **My testing and trialling** | What testing and trialling did I do?  Debugging/testing: No problems with collision between the 2: the character fell, hit the frame and didn’t roll, which is all that was needed at this stage.  User Feedback/testing: Not applicable here.  Evaluating against Requirements: Not applicable here | What did I do as a result of this?  There was nothing to do, everything worked just as I wanted and not much could be changed. | |
| Game Evidence:  <https://drive.google.com/drive/folders/1uzTjqKDJ46rvrRasHwlNakumMxDDQOnj>  Project Management tools at this point:  [Trello board](https://drive.google.com/drive/u/0/folders/16zkfCYh55jd6xMP7bfJpJS3Agbhw1Bgs) | | | |
|

| **Component:** Character movement | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | As I wanted the game to have in-air controls, I could either allow movement as an in-air jump or have the character move straight in the direction that is pressed. In the end I chose to have an in-air jump as the main method of movement as I thought that it would allow some more interesting design decisions than the character simply moving straight in a direction eg. having to time the jumps well in a tunnel to get through rather than just holding down an arrow key.  I could also allow a vertical jump along with jumping to the left and right, but I chose not to do this for similar reasons above eg. in a vertical corridor - having to time the jumps right instead of just spam clicking the up arrow. | | |
| **My testing and trialling** | What testing and trialling did I do?  Debugging/testing: Found out that the collider for the character becomes out of place when the character turns, as the model of the character moves slightly when turning. This results in the hitbox for the character being too far to the right when the character is facing left. There were no problems that I found with the character actually moving  User Feedback/testing: Got help from users to find a good ratio of vertical-horizontal distance when the character ‘jumps’.  Evaluating against Requirements: This type of movement can be done either in air or on the ground. | What did I do as a result of this?  Changed the hitbox to be at the center of the character model and changed it to be a thinner oval shape. This means that the hitbox will not extend in front of the model, and instead is inside of the front of the model. This way it also gives players some leniency for close misses, where the model can clip into the wall slightly but the character still survives, which will feel good for the player. The hitbox will also extend behind the model only very little.  I made the ratio for horizontal:vertical distance be 4:5. | |
| Game Evidence:  <https://drive.google.com/drive/folders/1YmpR5oHaU8dlTGGRJvgefpxtNmNAWeWh>  Project Management tools at this point:  [Trello Board](https://drive.google.com/drive/u/0/folders/1g5nvdJPEUxO42LfQi1Y3AvsbJNUnattD) | | | |
|

**Testing Table for Movement**

| Test Number | Input | Expected Outcome | Actual Outcome | Pass/fail |
| --- | --- | --- | --- | --- |
| 1 | Click right arrow key | Character jumps right | Character jumps right | Pass |
| 2 | Click left arrow key | Character jumps left | Character jumps left | Pass |
| 3 | Hold right or left arrow key | Character jumps once in the direction pressed | Character jumps once in the direction pressed | Pass |
| 4 | Any key other than right or left arrow key pressed | Nothing happens | Nothing happens | Pass |
| 5 | Left arrow key pressed while right arrow key is being held down | Character jumps left after the initial jump right from holding the right arrow key (same happens for keys other way round) | Character jumps left after the initial jump right from holding the right arrow key (same happens for keys other way round) | Pass |
| 6 | Both keys pressed at same time | Unsure | Keys cannot be pressed at the same time, the character will move in the direction of whichever key is pressed second | Pass |

| **Component:** Have boxes as a way to get to the next scene and have frames ‘kill you’ | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | Having all frames kill you vs having some frames that don’t kill you. I chose to add some frames that don’t kill you in the end, both because it allows some more level design decisions, and because with them, the character is just stuck in an endless loop of spawning and dying as it falls to the ground if the player doesn’t click anything.  I also trialled multiple different designs for the frames, but ended up choosing the frame with a thicker outside on all sides (third arch down on the left). Based on design alone, me and others decided that the second arch down on the right looked the best, and the arch that I ended up choosing looked the second best. Both were also easy to see for the user. I chose the frame that I chose as the best looking frame would be a lot harder to implement (14 different sprites would need to be imported and used to make structures - bottom left) and take up a lot of extra time. This is why I chose the second - best frame, as it would take a lot less time to use. | | |
| **My testing and trialling** | What testing and trialling did I do?  Debugging/testing: I found that the weird hitboxes weren’t very noticeable on the frames that kill you but when you hit the safe frames the character noticeably clips through them. The boxes worked fine.  User Feedback/testing: Users thought that safe frames didn’t look different enough from the unsafe frames.  Evaluating against Requirements: Boxes are the way to move between levels and frames are the way that your character dies. This turns the game into a precise ‘platformer’ which has the focus of avoiding obstacles. | What did I do as a result of this?  To stop this I extended the hitbox of the safe frames so that it looks like the character sprite is hitting the outside of the frame sprite and stopping, but in reality the frame hitbox is extended and is hitting the hitbox of the character in the middle of the sprite.  Changed the colour of safe frames to be more noticeable.  Old:  New: | |
| Game Evidence:  <https://drive.google.com/drive/folders/1YjddlNTGC6q0C3PaZWQZbBX2XYIRVyAd>  Project Management tools at this point:  [Trello Board](https://drive.google.com/drive/u/0/folders/1c_OdSf3iKEpTGMtozfwEjMe4Q11oeYei) | | | |
|

| **Component:** First and second level and explain how to play the game | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | I designed 3 levels at this stage. The first I used as the first level in the game. The second one that I made I was going to use in my game as the second level but after feedback and some thought I decided to leave it out. The third level that I made ended up being the second level.  Level not in game: | | |
| **My testing and trialling** | What testing and trialling did I do?  Debugging/testing: Not applicable here  User Feedback/testing:  I got users to look at each level that I made, and people generally didn’t like the second level.  Evaluating against Requirements: Multiple levels have been made. | What did I do as a result of this?  Not use the second level that I made and instead used the third level that I made as the second level in the game. | |
| Game Evidence:  <https://drive.google.com/drive/folders/1VdGLKMhfeRyyejdYF2rLQyRGMgI8lh12>  Project Management tools at this point:  [Trello Board](https://drive.google.com/drive/u/0/folders/1oZ1S8tDclcGv5Jl_EicYUwOsRQDqFEwp) | | | |
|

| **Component:** Death count feature | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | I could either collect the death count of individual levels and then show the death count of each level to the user at the end, or I could add up the user’s total deaths. I decided to just so a total of all deaths it would be too much information to be shown if more levels were to be added in the future. | | |
| **My testing and trialling** | What testing and trialling did I do?  Debugging/testing: On my first attempt the death count didn’t save across levels. This is because gamestatus (the object keeping the death count), got deleted and remade every new scene.  User Feedback/testing: When testing, users found out that it is possible for the character to hit 2 frames at once. This causes the death count to go up by 2 when the user only dies once.  Evaluating against Requirements: This component does not fulfil any requirements but is an extra feature that improves my game. | What did I do as a result of this?  To stop gamestatus from being deleted each scene I added a singleton pattern to save the old gamestatus and delete any new ones.  It still wasn’t working and after some testing I found out that it was because of how functions are executed in a single frame. To fix this I first disabled gamestatus and destroyed it later (this is because disabling it will happen earlier than destroying it).  I tried many different methods to fix the death count going up by 2, including coroutines to make it so that deaths could only be added every half a second, but nothing seemed to work. From trying to fix it I found out that the score is added at exactly the same time (as coroutines didn’t work) and I couldn't think of a fix. | |
| Game Evidence:  <https://drive.google.com/drive/folders/1wUN9I56QwP58t6-gF5lMAcbA2TyBOVy8>  Project Management tools at this point:  [Trello Board](https://drive.google.com/drive/u/0/folders/10smr3HMHVv6xBkHZjWAt0r8mAgh5RMz6) | | | |
|

| **Component:** Score feature in some form and extra levels. | | | |
| --- | --- | --- | --- |
| **What alternative techniques or components did I trial? Which did I choose and why?** | There are 2 main things that I could do for score. I could have it integrated with the death count so at the end of the game, you receive a score for how many deaths you have: lower deaths = higher score. The other thing that I could do is have collectables in the levels that give you points when you collect them. I chose the second option because of a few reasons. 1, it feels more interactive for the player collecting these items rather than just receiving a number. 2, the second option allows for some more interesting design in the levels and an option for extra challenge other than ‘do the same levels but die less’. 3, the first option may make the player feel bad if they receive a low score, whereas the second option is simply an optional thing that players can do if they want.  I made 1 more level and used that as level 3. | | |
| **My testing and trialling** | What testing and trialling did I do?  (This was also my final stage of testing and feedback as nothing more was added afterwards)  Debugging/testing: There were no major bugs when testing, but lots of little bugs such as score not resetting properly when dying on a new level. It took a lot of time to fix them all and make this component.  User Feedback/testing: Main user feedback was on where the points orbs should be placed, whether they were too easy to get to or too hard.  Evaluating against Requirements: This component does not fulfil any requirements but is an extra feature that improves my game. | What did I do as a result of this?  I fixed all the bugs that I could find, and changed the location of 2 point orbs.  Old:New:Old:New: | |
| Game Evidence:  <https://drive.google.com/drive/folders/10HYjAF24K2PjsyIXWW8y5LbtGKJLyf_Q>  Somehow or other I have lost the component trial where death count is counted for each level and displayed at the end, so that isn’t on here.  Project Management tools at this point:  [Trello Board](https://drive.google.com/drive/u/0/folders/1eWkPghdaVrspLMh4dl2oXpZBAeUP6GSt) | | | |
|

| **3. Finishing up** |
| --- |

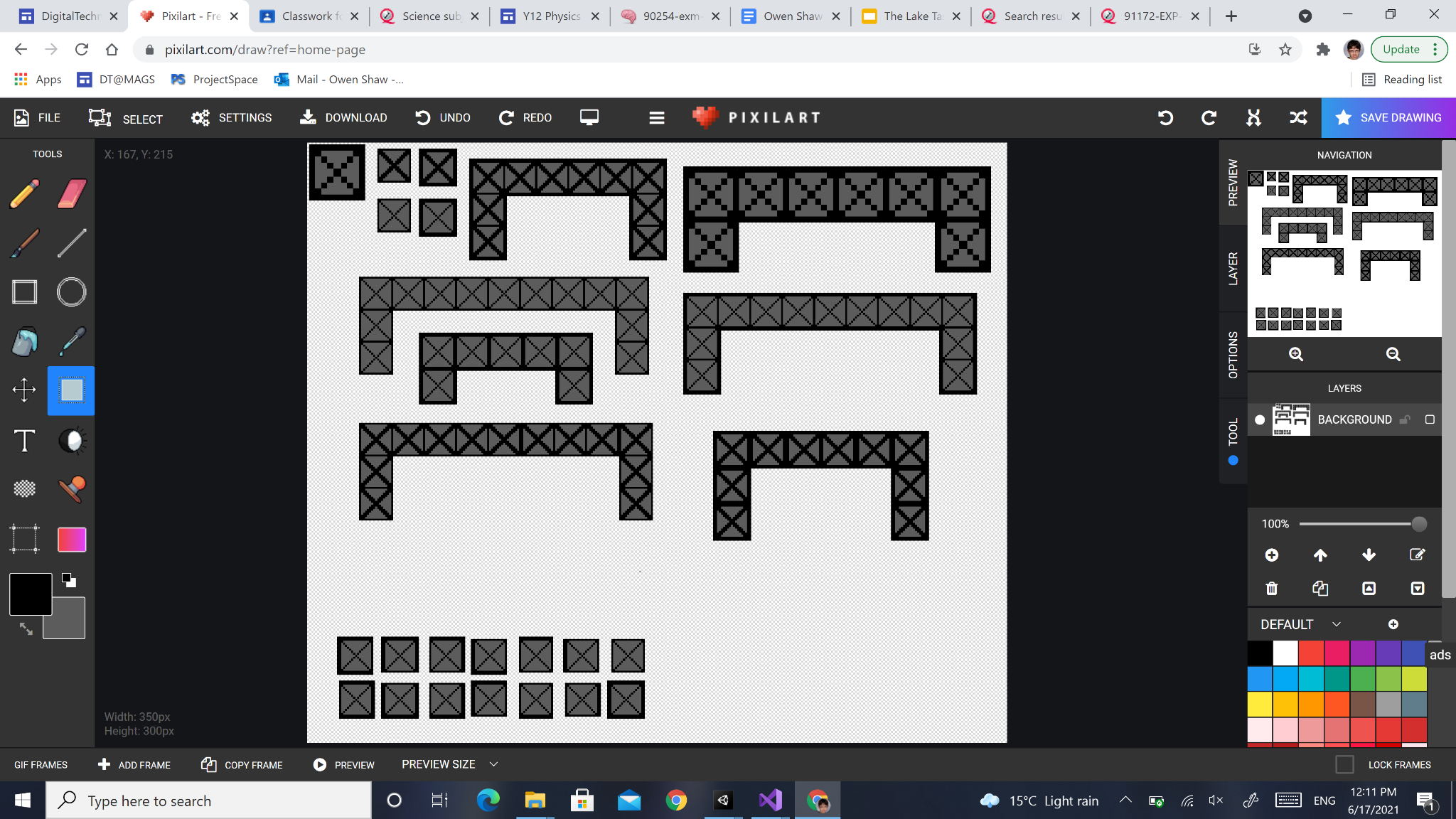
**Link to my final outcome (or screenshots):**

<https://drive.google.com/file/d/1E8m8vFhGu92g0tdHn7nZ9ZOuPVf4O3VI/view>

**Relevant implications:**

Sustainability and future proofing - It is important for the game to be expanded upon later, especially because there currently isn’t much in the game. More levels could be added, a proper tutorial, more features such as movable and moving objects, enemies, doors with triggers, other features and general tuning to the game. I have made it so that future features like these wouldn’t be hard to implement into the game in multiple ways. New levels/scenes could be easily added wherever into the program (except before the 3rd scene/first level) because I haven’t used numbers when loading new scenes (except when loading the first level on the third scene. This is the only time that numbers are used though and only 1 value would need to be changed when adding scenes before the first level). Instead I have retrieved the current scene index number (placement in the scene array) and used that to load new scenes. Eg. SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1); is used to go to the next level instead of manually going to the next level such as SceneManager.LoadScene(5); to load the 6th scene. I have also used tags for things like the character and frames so that in the future if more components are added, they won’t affect things that they aren’t supposed to eg. separate moving object picking up a point orb instead of the character and giving you a point. I have also almost exclusively used variables in the code instead of direct numbers so that if I want to tune some values and parts of the game, I won’t have to go and change each individual value. I have also used [SerializeField] so that I can do this in the unity editor instead of in code.

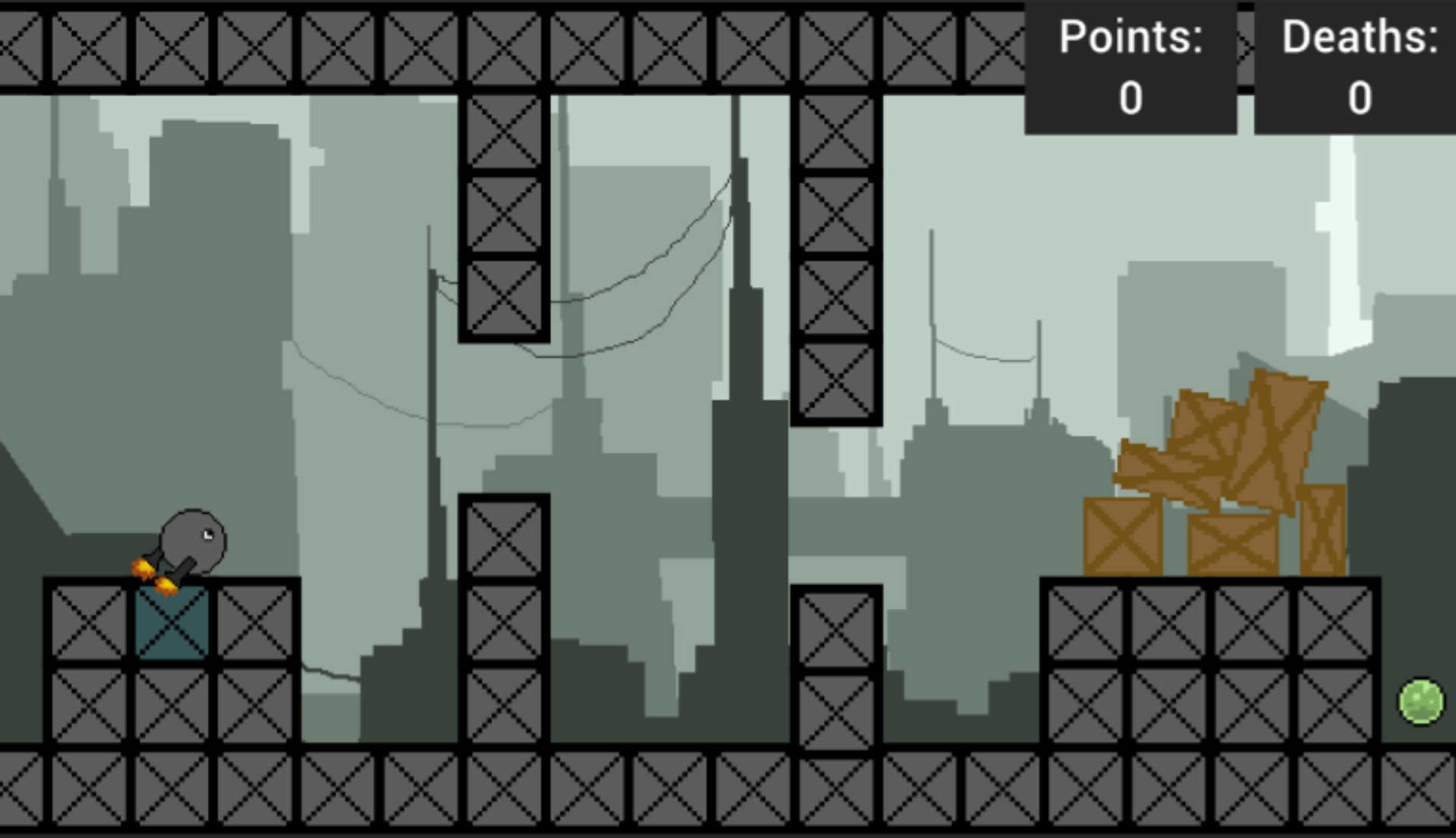
Aesthetics/design - As my project is a game and is meant to be something that people will do in their free time, it should look good and I should make it so it is engaging to play. I have tried to make all of the sprites in the game look good, and have gathered feedback on how to make them look better until they look good enough to be in the game. One example of this is with the frames (this was talked about in the component ‘Have boxes as a way to get to the next scene and have frames ‘kill you’’).



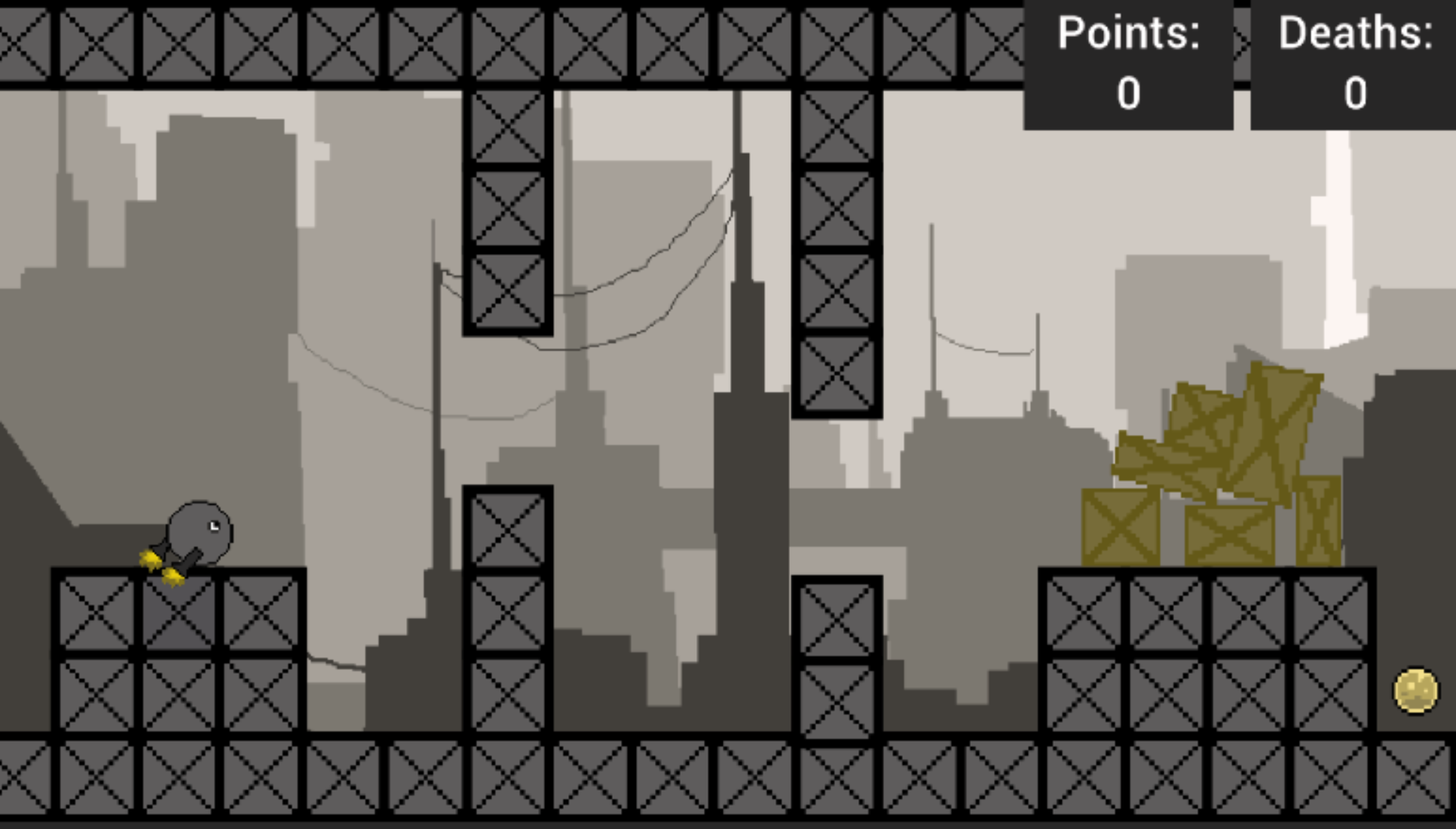
I made many different designs for the frame and asked for feedback for which ones looked best. Eventually it came down to between 2nd to bottom on the left and 2nd to bottom on the right, and I decided to use the one on the left as it would take a lot more work for the one on the right. For the one on the right, I would need to add 16 different sprites into the game to build structures with the frames to make it so only the outside of the structure would be boldened. It would take up a lot of time and I would rather use that time to create other features in the game, so I chose the left one instead.

Accessibility - The game should be able to be played by almost everyone, including those with disabilities including colour blindness or other eyesight problems, or those with minor movement problems with their hands. For those who have minor movement problems with their hands, I have made the buttons for moving between scenes big enough so that they shouldn’t struggle to click on them. The rest of the game only involves clicking 2 buttons, and doesn’t require much movement at all. Those with slow reaction times or find it difficult to act precisely will find it very difficult to play the game due to the nature of the game. Large buttons and text are also useful for people whose eyesight isn’t great. For colourblind people, I have tried to make the game as accessible as possible. In the end, I managed to make it so that all kinds of colourblind people except those with Red-Blind/Protanopia.

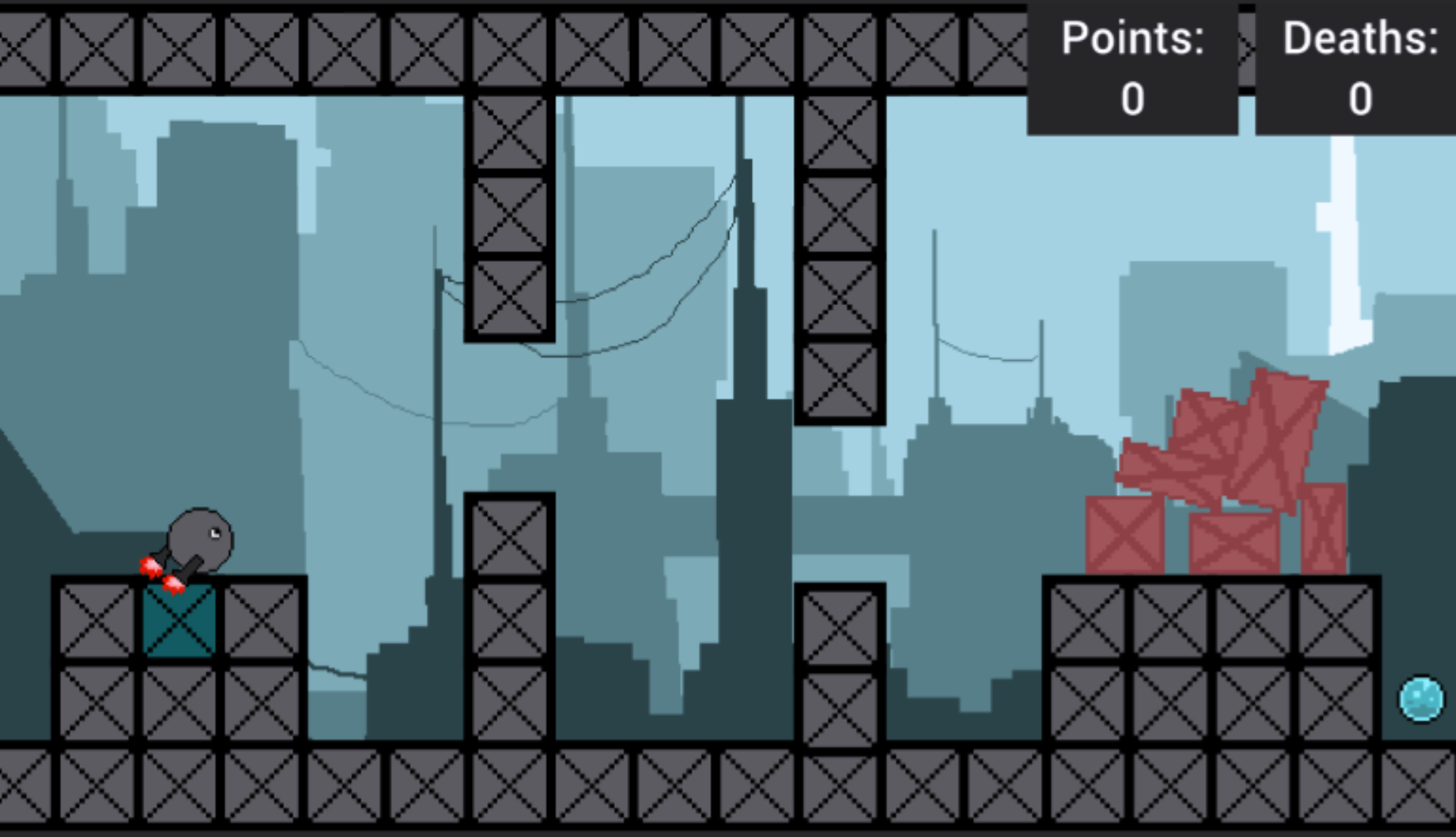
Red-Weak/Protanomaly:



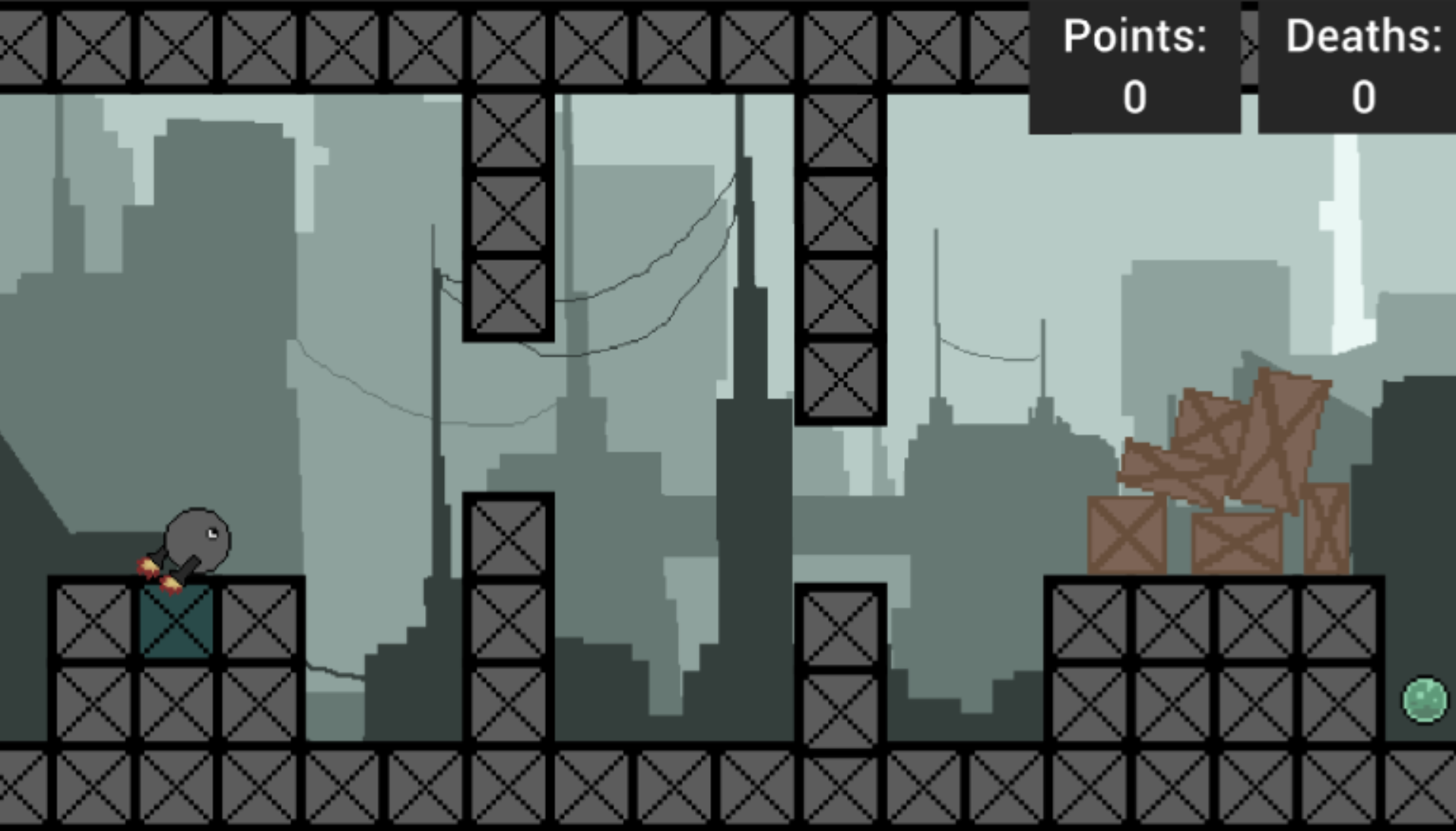
Red-Blind/Protanopia:



Blue-Blind/Tritanopia:



Blue Cone Monochromacy:



(I tested many different kinds of color blindness, these are only some of them. To get these images and test what my game looked like to colourblind people, I used [this](https://www.color-blindness.com/coblis-color-blindness-simulator/))

On all types of colour blindness that I tested, all types of color blindness except Red-Blind/Protanopia could see everything clearly, and the only thing that Red-Blind/Protanopia couldn’t do was make out the difference between safe frames and unsafe frames.

Legal/intellectual property - The game's background art was not created by me. It was created by another user who submitted the artwork onto [pixilart.com](https://www.pixilart.com/). Any artwork/pixel art that is listed as public on the website can be viewed and used in any way by anyone else (public domain).

“By submitting content from the pixel drawing application (located at "pixilart.com/draw") to Pixilart for inclusion on the website, you grant Pixilart worldwide, royalty-free, and non-exclusive license to reproduce, modify, adapt, and publish and advertise the content without prior notice. Drawings you create on Pixilart may be used for public/commercial use.”

(<https://www.pixilart.com/terms>)

The user has since taken the image off pixilart but I downloaded the image before doing so, and it is legal to continue to use the image as the user had previously put it on pixelart.

All other images/assets used in the game were created by me on pixelart and any copyright issues are coincidental. I had no inspiration from anything else to make them.

“You may use the services provided by Pixilart for your own internal, personal, commercial use outside of Pixilart services.” (<https://www.pixilart.com/terms>).

The game does not try to obtain or store any of the user’s personal information at all, not even their name.

**How did planning assist in the development of a high-quality outcome?**

Planning assisted in the development of a high-quality outcome as it helped me organise my work. It let me split the game up into different sections to be done at different times which helped me be flexible with how the game was developed. Each section had different parts to it from programming to testing to feedback which helped me locate bugs and problems with the game and helped me make sure that the game was as polished as I could make it at each step. All this proved to be useful when I had 3 weeks off due to being sick and I couldn’t really work. This made it so that I only had a very small amount of time to make my game but due to my planning I was able to produce a high-quality outcome in that time. It helped me manage my time and work so that, while the finished game didn’t have as many features as I would have liked, I was able to finish the game and give it some features in the 1 and a half weeks I had to work on it.

**How did testing and trialling assist in the development of a high-quality outcome?**

Trialing different components allowed me to see which ones I liked and which ones I didn’t. They allowed me to decide between components not just by what I think it would be like but what it will be like. An example of this is the death count: I was originally going to display the deaths of each level but when I tested it in game, I realised that it would be hard to find the space for it if more levels were to be added after seeing the space taken up on screen.

Testing after each component helped me considerably when finding the source of a bug. If there was no bug in the previous component and there is now, it must be from the new component. This was very helpful when it helped me figure out why the death count wasn’t saving across scenes. It let me know that the problem had to be something to do with gamestatus and from there I was able to work it out.

User feedback was also very helpful in improving my game and making it feel better to play, which was prevalent when trying to figure out the best height and length of the character’s jump. Users tried both large and small jumps and tinkered with the values, as did I, until we found a good horizontal and vertical distance for the jump. User feedback was also helpful when finding bugs with the game, such as finding out the bug where death count would go up by 2 if the character hit 2 frames at once. My planning and decomposition used with testing and trialing at each step worked together to help me make a high-quality outcome.

| **4. Final checklist: Make sure my evidence includes all of the following** |
| --- |

Achieved: Use *advanced processes* to develop a digital technologies outcome.

* using appropriate project management tools and techniques to plan the development of a digital technologies outcome
* decomposing the outcome into smaller components
* trialling the components of the digital technologies outcome
* testing that the digital technologies outcome functions as intended
* explaining relevant implications

and at least two of:

* saving back-up copies with a logical file naming system
* using collaboration tools
* using simple version control software applications
* using tools or systems to plan tasks and milestones
* adjusting key actions and tasks where appropriate

Merit: Use advanced processes to develop an *informed* digital technologies outcome.

* effectively using project management and version control tools and techniques to manage the development of a digital technologies outcome
* trialling multiple components and/or techniques and selecting those which are most suitable
* using information appropriately from testing and trialling to improve the functionality of the digital technologies outcome
* addressing relevant implications

Excellence: Use advanced processes to develop a *refined* digital technologies outcome.

* discussing how the information from planning, testing and trialling of components assisted in the development of a high-quality outcome