

## COMP2160 - Game Development Task 2

Group: **GDT2-C**

Name	Student Number
Nathan Chen	45956146
Joshua Lukas	47712651

Itch.io link: <https://nafanchen.itch.io/gdt2-c>

This document may include images. To insert an image into your documentation, place it in the "Images" subfolder, then place the below text where you want the image to appear:

![Place any alt text here](Images/<IMAGE NAME AND FILE EXTENSION>)

### Features implemented:

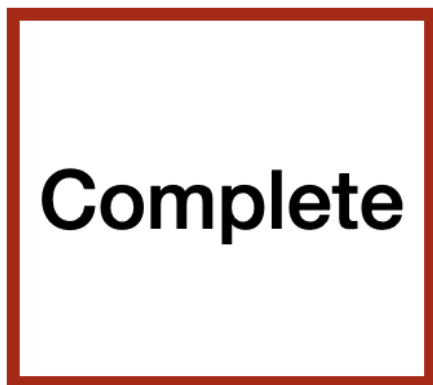
Feature	Requirements	%	Complete (YES/NO)
Levels	Level 1	2	Yes
	Level 2	2	Yes
	Level 3	2	Yes
Horizontal Movement	Movement with WASD / Arrow Keys	2	Yes
	Turning with mouse	2	Yes
	Collisions	2	Yes
Vertical Movement	Falling	2	Yes
	Jumping	2	Yes
	Coyote time	2	Yes
	Jump buffer	2	Yes
Controller Support	Movement	1	Yes
	Turning avatar	2	Yes
	Jump	1	Yes
	Camera pitch	2	Yes
Camera	Camera dolly	2	Yes
	Third person camera	2	Yes
	Pitch control	2	Yes
	Dolly control	2	Yes
	Saving between levels	3	No
Goals & Death	Lag	3	No
	Goals	3	Yes
	Death zones	3	Yes
Level Features	Launch Pads	3	Yes

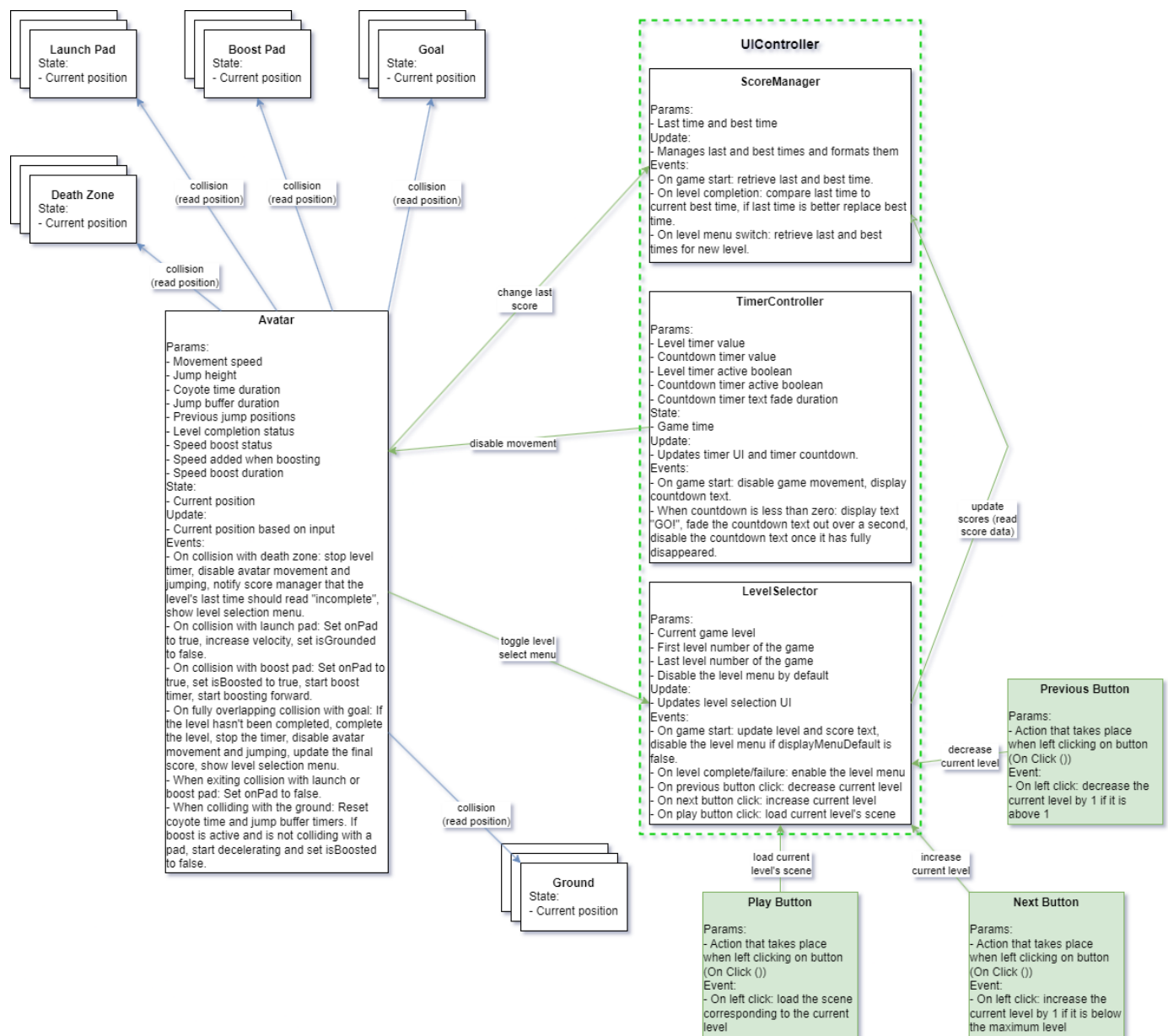
	Boost Pads	3	Yes
In -Game UI	Custom Font	1	Yes
	Countdown	3	Yes
	Timer	3	Yes
Level Selection	Selection UI	2	Yes
	Scene loading	2	Yes
High Scores	Updating scores	2	Yes
	Saving between levels	2	Yes
Debug Gizmos	Collision points and normals	3	Yes
	Jump points	3	Yes
	Avatar path	3	Yes
WebGL Build	Working build on itch	4	Yes

### 1. Entity Relationship Diagram (5%)

Please include an ERD for your designed software architecture for the game, following the guidelines and examples in the lectures. Make sure all elements of your ERD are legible. Marks will be deducted for unreadable diagrams.

Your diagram should match your code as much as possible, however you can indicate unimplemented elements with a dotted line border as shown below:





## Project Management (5%)

Complete the table below indicating which tasks have been assigned to each team member and whether it was completed.

Tasks may include:

- Designing the code architecture
- Developing code for specific features or components
- Conducting QA
- Writing documentation
- Other important development or production tasks

Note that both team members are expected to contribute to both the writing of code and the documentation, as a key learning-outcome of this assignment is to practice managing a multi-developer project.

You should commit and push a version of this report *at least one week* before the final deadline containing a draft task assignment. This can be updated in your final submission.

Task	Assigned To	Completed On
Clone GIT repository	Nathan	21/10/2024
Create a unity project with a simple scene	Nathan	23/10/2024
Clone GIT repository	Josh	24/10/2024
Add avatar movement and collision	Nathan	25/10/2024
Add rigidbody avatar movement and collision	Nathan	28/10/2024
Add camera script with camera controls and rotation	Nathan	29/10/2024
Add coyote and buffer to avatar script	Nathan	30/10/2024
Create level 1 for the game	Nathan	30/10/2024
Add controller support	Nathan	31/10/2024
Add debug gizmos	Nathan	31/10/2024
Create word document for documentation	Nathan	31/10/2024
Designed Level 2 and 3	Nathan	04/11/2024
Add in-game UI	Josh	04/11/2024
Add level selection	Josh	05/11/2024
Add goals and deaths	Josh	05/11/2024
Add Level Features	Josh	05/11/2024
Add high scores	Josh	05/11/2024
Add WebGL compatibility	Nathan	05/11/2024
Add 2 QA Plans	Nathan	05/11/2024
Add ERD for project	Josh	05/11/2024

## 2.1 Rationale

We have utilised this form of work allocation as it allows us to have an even workload where we can balance the work with time. It also allows us to work on separate functions of the game where for example the in-game UI can be made and structured separately to

the movement controls. This allows a more efficient workflow where we can both zone in on the features we are working on, without it having an effect on the other person's ability to implement the feature.

## 2.2 Version Control

Utilised GitHub desktop to organise and push work onto the online repository from the local repository. Would push to the online repository from the local repository every time a feature was finalised or worked on. With each update we would add a description so we could track our progress on which features we worked on as well as having a timeline of features worked on allows us to plan which features need prioritising and which can be placed on the backfoot for later.

## 3. Quality Assurance (QA) Plan (5%)

Choose two of the high-level features that you have implemented (e.g. 2. *Horizontal Movement* and 5. *Camera*) and write complete the QA Plan template below indicating how you would test each of these features.

### QA Plan for Horizontal Movement:

Test ID	Requirement	Test Scene	Task	Expected Result
HM-01	The avatar should move horizontally as well as forwards and backwards based on player input using W/A/S/D keys, arrow keys and the left stick of the game controller	Will be tested in the HM QA Testing scene	Test the movement of the avatar by pressing W/A/S/D and arrow keys individually, as well as different combinations to ensure proper diagonal movement. Additionally test the left stick of the game controller for smooth movement	The avatar moves forward, backwards, left and right based on keyboard and controller inputs. Diagonal movement is achieved when two directional inputs are pressed simultaneously. The speed of the avatar should be consistent
HM-02	The avatar controls should be relative to the direction the camera is facing	Will be tested in the HM QA Testing scene	Test the forward movement of the avatar by pressing W or up arrow on the keyboard, as well as left stick in the upwards direction. Then use the mouse to move the camera 90 degrees to the right and test forward movement again	The avatar should move forward on both tasks where the changing of the camera angle should not change how the controls of the avatar are applied. Forward movement should always be in the direction the camera is facing
HM-03	The avatar must not pass through or get stuck in obstacles and	Will be tested in the HM QA Testing scene	Move the avatar horizontally into the obstacles by pressing the W/A/S/D keys,	The avatar should immediately stop upon colliding with an obstacle

	should stop horizontal movement upon collision		arrow keys as well as the left stick on the game controller. Do the same but with the camera direction changed	and not clip or penetrate through the wall in any form. The avatar should move smoothly along the walls with no jittering or getting stuck
--	--	--	--	--

### QA Plan for Controller Support:

Test ID	Requirement	Test Scene	Task	Expected Result
CS-01	The left stick of the game controller should move the avatar smoothly in all directions	Will be tested in the CS QA Testing Scene	Test the movement of the avatar in all directions including diagonally with the left stick of the game controller	The avatar should move smoothly in the direction of the left stick. The movement speed should be consistent and not exceed the moveSpeed value
CS-02	The right stick of the game controller should adjust the camera direction horizontally	Will be tested in the CS QA Testing Scene	Test the left and right movement of the camera horizontally with the right stick of the game controller	The camera should move in the direction of the right game stick without affecting the pitch. The rotation should be responsive and at an appropriate speed
CS-03	The “south” button should cause the avatar to jump	Will be tested in the CS QA Testing Scene	Press the “south” button on the game controller once to make the avatar jump. Also test repeated presses to check jump functionality	The avatar should jump when the “south” button is pressed. Jump height should be consistent with the jump settings. Coyote time and jump buffering should also work as intended
CS-04	The vertical axis of the right stick on the game controller should adjust camera pitch when the right shoulder button is pressed	Will be tested in the CS QA Testing Scene	Press the right shoulder button on the game controller while adjusting the pitch of the camera with the vertical axis of the right stick	The camera pitch should adjust smoothly and proportionally to the vertical input from the right stick. The pitch should stay within acceptable limits to avoid flipping upside down
CS-05	The up and down axis of the D-Pad should	Will be tested in the CS QA Testing Scene	Press the up and down buttons on the D-Pad to adjust the camera dolly	The camera should move closer or further from the avatar respectively. The

	adjust the camera dolly			dolly movement should be smooth and consistent without exceeding the limits
--	-------------------------	--	--	---

Note: The *Test Scene* column should refer to specific Unity scenes in your project, specifically designed for QA Testing of individual features (i.e. not your the three levels included in your final game).

#### 4. Data Management Plan (5%)

Imagine a scenario in which you plan to publish this game on Steam. You are deciding what analytics to include in the game. Your designers want to gather playtesting analytics to assess the appeal and difficulty of the game. Your marketing team want user data to be able to better assess the market niche for the game and direct market the game to players who like similar games.

The data you plan to gather includes the following.

For every time the game is played:

Question	Data Source
Who is playing the game?	The system user name given by the C# property <a href="#">Environment.UserName</a>
Who is playing the game?	The player's Steam Persona name via the <a href="#">Steamworks API</a>
What machine is the game being played on?	The IP address of the computer given by the C# method <a href="#">Dns.GetHostEntry</a>
When did the player start the game?	The date & time given by given by the C# property <a href="#">DateTime.Now</a>
When did the player quit the game?	The date & time given by given by the C# property <a href="#">DateTime.Now</a>
What other games do they enjoy?	The list of games owned by the player. via the <a href="#">Steamworks API</a>
Who else might like this game?	The Steam ID & names of the players friends, via the <a href="#">Steamworks API</a>

For every level that the player attempts:

Question	Data Source
What path did they take?	A sample of the avatar position taken once per second

For the sake of this DMP, assume you have published the game on Steam and are retrieving this data from multiple users who have bought your game and signed an appropriate End User License Agreement consenting to this data gathering. Data is being gathered via the Unity Analytics service.

Complete the following Data Management Plan questions as it pertains to this project. Consider the sensitivity of the data you are collecting, and how this influences your storage and potential usage. Refer to the ACS Code of Professional Ethics, Australian Privacy Act 1988, and the GDPR where necessary.

## **4.1 Data storage and disposal**

### **4.1.1 Data storage and location**

Document where the data will be stored and what system/s will manage the data. This information should include:

- storage locations both internal and external
- how data will be accessed over time and the systems used
- back-up and recovery plan details or a link to your internal documentation.

Data will be stored internally on local secure servers, where they will have full-disk encryption and will be thoroughly protected with measures such as two-factor authentication. Data will be stored externally with Unity Analytics. Only authorised users will be able to access data such as developers and team members where such user permissions will be managed through a role-based access control system. Daily backups will be performed on the internal data servers and a full recovery plan will be enacted in the case of a loss. Alerts will be enabled for when data storage reaches a certain threshold in order to maintain data storage flows.

### **4.1.2 Data disposal (keep, destroy or transfer data)**

How will you destroy data if necessary?

We will use secure and irreversible methods to dispose of data to ensure that it cannot be recovered or reconstructed.

### **4.1.3 Privacy**

Summarise the privacy implications of data created, collected or published. Refer to the lecture notes for some hints.



Information may include:

- listing key data that contains private and sensitive information (if any)
- listing key data that is identifiable – where an individual’s identity can be reasonably ascertained.
- 

The collected Steam ID, their respective IP address and system user name are all considered sensitive data as they can be used to identify the player.

#### **4.1.4 Ethics**

Consider any ethical issues not covered elsewhere in the DMP. Refer back to the ACS Code of Ethics.

### **4.2 Using Data**

#### **4.2.1 Data Analysis**

How will the data be analysed? How will you represent the data and compare different data points with one another?

#### **4.2.2 Data Implementation**

How will the data be used to improve the game? What changes might occur based on using this data to the game’s design, code base, etc.?

#### **4.2.3 Data publication**

Will the data be published? Information here may include:

- Publishing locations such as online blogs, social media.
- Planned showcases of the data at conferences, meetings or other venues.
- 

#### **4.2.4 Data sharing**

Document any parties that you may share the data with, such as other game developers, advertisers, etc., and any expected challenges or identified risks with this sharing.

### **Third Party Assets Used**

Please indicate any third-party assets used in your project including the details below

Asset Name	Author	URL	License

### **Generative AI Use Acknowledgement**

Use the below table to indicate any Generative AI or writing assistance tools used in creating your document. Please be honest and thorough in your reporting, as this will allow us to give you the marks you have earned. Place any drafts or other evidence inside this

repository. This form and related evidence do not count to your word count. An example has been included. Please replace this with any actual tools, and add more as necessary.

**Tool Used: ChatGPT**

**Nature of Use** Finding relevant design theory.

**Evidence Attached?** Screenshot of ChatGPT conversation included in the folder "GenAI" in this repo.

**Additional Notes:** I used ChatGPT to try and find some more relevant design theory that I could apply to my game. After googling them, however, I found most of them were inaccurate, and some didn't exist. One theory mentioned, however, was useful, and I've incorporated it into my work.

**Tool Used: Example**

**Nature of Use** Example Text

**Evidence Attached?** Example Text

**Additional Notes:** Example Text