CIT419

**Gaming and Simulation Capstone**

***Final Project Report***

**Fight Club**

***Zac Brennan***

Fall 2020

*12/09/2020*

**Pennsylvania College of Technology**

TABLE OF CONTENTS

[1 VISION STATEMENT 3](#_Toc15666166)

[1.1 GAME/SIMULATION LOGLINE 3](#_Toc15666167)

[1.2 GAMEPLAY SYNOPSIS 3](#_Toc15666168)

[1.3 AUDIENCE AND SYSTEM REQUIREMENTS 3](#_Toc15666169)

[1.3.1 TARGET AUDIENCE 3](#_Toc15666170)

[1.3.2 PLATFORM 3](#_Toc15666171)

[1.3.3 SYSTEM REQUIREMENTS 3](#_Toc15666172)

[2 CAPSTONE REQUIREMENTS 3](#_Toc15666173)

[2.1 PROGRAMMING COMPONENT 3](#_Toc15666174)

[2.2 DATABASE COMPONENT 4](#_Toc15666175)

[2.3 WEB COMPONENT 4](#_Toc15666176)

[2.4 INFORMATION ASSURANCE AND SECURITY COMPONENT 4](#_Toc15666177)

[3 RESEARCH 4](#_Toc15666178)

[3.1 RESEARCH COMPONENT 4](#_Toc15666179)

[3.2 RESEARCH SOURCES 4](#_Toc15666180)

[4 REFLECTIONS 4](#_Toc15666181)

# VISION STATEMENT

## GAME/SIMULATION LOGLINE

A multiplayer fighting game with a fixed side perspective. The combat would be generalized and would be more focused on knocking opponents off a stage.

## GAMEPLAY SYNOPSIS

The goal of Fight Club is to allow two players to have at it in a floating arena. I’m hoping for an experience that is both casual and competitive. A mix between the two by giving the game simple mechanics, but allowing for the user to use good positioning to their advantage. Players have access to two types of combat abilities.

The uniqueness I would say would actually come from the primitive gameplay. Other fighting games focus on long drawn out combos, this game would focus on positioning. Being able to get around opponents and finding their blind spot will be impativie for the general outcome of the players experience. Even without that side of the gameplay, casual players should be able to have fun with just fighting. This mix of casual and competitive gameplay I believe will mesh well together.

A big pull to this game would be the multiplayer aspect. Two separate players can participate in a lan game. The fighting is synchronized to be able to be the same for both players. The camera would be a side perspective giving both players a view of the arena. This type of view really amplifies the multiplayer of the game. It allows the player to see every movement the other player performs and allows for them to react to the opponent accordingly.

Movement and combat would be the main gameplay mechanic for this game. Movement is limited to jumping three times in the air. This resets after the player hits the ground. It also consists of general movement of the player. The combat will also be a little generalized. The player has access to a basic punching mechanic and anyone hit by the punch will have a knockback effect placed on them based on the damage they’ve acquired. They also have access to an ability that differs between characters.

There are two separate characters that the player can choose from. They have the option of the Cactusman or the Devilman. Each character has other aspects that differ, such as speed, punch distance, damage and jump height. The Cactusman is faster on the ground and hits harder, but his jumps are lower than the Devilman and has less reach. The Devilman’s special ability is also a ranged attack, while the Cactusman’s ability is a sphere around them. Each play can choose a unique character to play as when they start the match.

The players will also have access to a UI that floats above their character head. It’s unique to each player, allowing the individual to see their current stats. These stats are their deaths and their current damage. When knocked off the stage, their deaths increase and their damage is reset. The damage is increased when hit by the opponent’s punch or special ability. The player cannot see their opponent’s stats.

The setting of the current map is a desert town map. This fits the Cactusman character I went with that I found on the asset store. It fits really well with the theme of the game and gives a nice unique vibe for this type of game. This type of area isn’t generally utilized in fighting games and more used in story games. Allowing for this type of area allows for a fun feel to the game.

To add to the feel and tone of the game, I will be pulling music from a database that fits this theme. This music fits the individual character with a bonus song from another capstone student, Troy Records. There are two songs that fit the Cactusman and two songs that fit the Devilman. These help amplify the tone of the game.

## AUDIENCE AND SYSTEM REQUIREMENTS

## TARGET AUDIENCE

The target audience for this game are teens and young adults. This demographic is where fighting game fans usually tend to be. It can outstretch from those a bit for people who like party games as well.

## PLATFORM

This game is an exclusively PC only project. The player will need to have access to a mouse and keyboard.

## SYSTEM REQUIREMENTS

For Play: The player will be using Windows 10 to play the game. They will need access to a keyboard and an okay computer. They will also need access to the internet.

For Development: I was using Unity 2017 to produce this game. I have a very good computer with great specs. I also needed powerpoint for my presentations and photoshop for some button work. I also used SQLite as my database. I also needed access to the internet to play the game.

# CAPSTONE REQUIREMENTS

## PROGRAMMING COMPONENT

Programming is one of the key aspects to this game. All other components hinge on this component making it the most important. Database, Web, and Security components are all utilizing programming. That is why it is such an emphasis for my game.

The first thing I would go over is the movement system. It is stuck on the X and Y axis. The player can move upon these axises. This means they can jump as well and fall. The player is limited to up and down movement. The issue with two players is that each can move each other, if the player isn’t limited to their own character.

The combat portion is also represented here. The synchronization between both clients is important. This is addressed with programming and allows for a smooth experience. The player also has to spawn their attack, whether it be a punch or a special ability, on the server. That is completed with programming. The damage reaction system also is done this way. It knocks the player back if they get hit and is based on their damage.

If a player is knocked out of the play area, they spawn at the starting point. This will also reset all of their stats. These stats are displayed on a floating UI above their head. This follows the players around as they move through the game.

The next aspect would be the menu system. The player stats on the main menu where they can choose a song. The game manager saves the song. Then with simple scene management they can quit, start, or look at controls. When they click start it sends them to the network management scene. This allows them to pick a character then join a lan game. They will need to input the IP of the host device to join. Then this sends them to the play scene.

The multiplayer allows them to join a game via the network manager. Code allows them to change the base player prefab they want. It also allows their attacks to be synchronized. This is all important for smooth network play.

The database programming is mainly connecting to the database. It goes in and extracts the song information needed. Then plays the song and for the individual player. After grabbing the information, it concatenates the name and the artist. It then displays that into the dropbox. Each song is made into an instance of a Song class. This asks for the name, artist and filepath. It is then put in a list and it grabs the needed information.

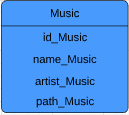
## DATABASE COMPONENT

The database that I used was SQLite. It is a locally hosted database that allowed me to get the songs I needed from it. There is only one of which is the Music table. The music table has four attributes:

* Id\_Music – This is the primary key and it increments for each song.
* Name\_Music – This is the name of the song.
* Artist\_Music – This is the artist of the song.
* Path\_Music – This is the file path in the song. These are held in the resource folder.

After a connection is made, it takes the information and places it where it needs to be. This is different depending on the type of info.

The ERD Diagram isn’t anything special because it only has one table. It would have no relations, just the one table that is in the project.



## WEB COMPONENT

The web component is allowing players to play together over a lan connection. There were many modifications I needed to make to allow the network manager to work correctly. These were mainly synchronization changes and allowing different buttons to allow for the player to join.

Synchronization wise the main thing needed was ClientRPCs and Commands. Commands would send information to the server from the client, then the ClientRPCs would take the information. It would synchronize it to the individual users.

The player is able to connect to the host via a lan connection. They must input that player’s IP and be on the same network. They will then join the other player and be able to interact accordingly.

Most changes to the network manager were to allow for the player to connect on a custom UI. There were some changes inplace to allow for the player to select their character. This worked by sending a message to the server before joining that changed their player prefab.

## INFORMATION ASSURANCE AND SECURITY COMPONENT

The security portion of the game is mainly implementing security by design. The vast majority of methods or variables are private except when needed otherwise. Something I learned after the project is that serializable variables show up on the inspector. If I were to go back, I would change more of the public variables to these.

The other portion that I focused on was the idea of my database being locally hosted. With the game already being interet connected, I wanted to keep the database in house. This avoids any other outside connections that could be a breaching problem.

# RESEARCH

## RESEARCH COMPONENT

There were two main research portions to my game. The database connection process and the multiplayer aspect of the game. These both were necessary as they added to the overall feel. Also, the web portion is imperative for the fight experience.

The multiplayer was incorporated as a main point for the game. Without it, the game wouldn’t be nearly the same. I had to restructure a lot of components that worked in single player, that messed the multiplayer experience up. This resource component defined the game as it is.

The other one was more for me to understand. I needed to do it for a database component. The main thing that this did was add music to the game. It allowed me to take the database information and connect it to the needed areas. I will say this was more for the overall tone and feel, but it was still important.

## RESEARCH SOURCES

Technologies, U. (n.d.). Using the Network Manager. Retrieved December 02, 2020, from <https://docs.unity3d.com/Manual/UNetManager.html>

Technologies, U. (n.d.). Remote Actions. Retrieved December 02, 2020, from <https://docs.unity3d.com/Manual/UNetActions.html>

KnnthRA. (2015, August 30). 1. Unity tutorial: High score with SQLite - Intro. Retrieved December 03, 2020, from <https://www.youtube.com/watch?v=laspFwXGprg>

Mwmdragon. (2019, July 15). SQLite in Unity. Retrieved December 03, 2020, from <http://www.mwmdragon.com/2019/07/15/sqlite-in-unity/>

# REFLECTIONS

I think that this helped me increase my love of game development more. It was very outside my comfort zone and helped push myself. I don’t think it made me like multiplayer games anymore than I did before though. They have a lot of frustrating quirks and I don’t really like them as much as single player games. I would be more open to working on them now and am not as afraid of them anymore.

This will help me grow for a future position. I think this type of project is the icing on the cake for my portfolio. It already had a lot of things going for it, but this shows me going outside my comfort zone. It also will help me if I need to work on a multiplayer project in the future.

I liked adding the multiple different characters the most. Although it was tedious with the multiplayer synchronization, it made me really happy to get two characters to work. I thought before that the game was good, but it really pushed it over the edge. With that said, the thing I disliked the least was the UI synchronization. I put over 15 hours trying to figure it out and had to goto an old version of the game because I broke it so much. It was really frustrating for me and is why I looked for a different solution.

I would give them the advice to work before the proposal date. I had a mass majority of my basic mechanics done and I thought the progress showed in my game. I think if you wait you will have a lot of problems with your demonstrations that could be avoided with some precaution.