Ultimate Fighting Chickenship

Final Report

Tempête group

May 2019

charlie.brosse@epita.fr
philippe.lefebvre@epita.fr
david.horozian@epita.fr

Contents

1 Introduction **3**
- 1.1 The roots of Tempête . . . 3
- 1.2 The idea of the game . . . 3
- 1.3 Goal of the game . . . 4

2 Book of specifications review **5**
- 2.1 A short history of fighting games . . . 5
- 2.2 Artistic direction and philosophy of the Project . . . 7
- 2.3 Technical aspect . . . 8
- 2.4 Economic aspect . . . 8
- 2.5 Task distribution . . . 9
- 2.6 Timeline . . . 10

3 Game Mechanics **11**
- 3.1 Comments . . . 11
- 3.2 Movement- Charlie . . . 13
- 3.3 Attacks - Charlie . . . 16
- 3.4 Other mechanics - Charlie . . . 19

4 Sound design **21**
- 4.1 Comments . . . 21
- 4.2 Music - Philippe . . . 21
- 4.3 Sound Effects - Philippe . . . 23

5 Graphics and User Interface **25**
- 5.1 Comments . . . 25
- 5.2 UI and Design - David . . . 25
- 5.3 3D modelling - Philippe . . . 26
- 5.4 Animations - Philippe . . . 27
- 5.5 Map Design - David . . . 27

6 Miscellaneous **29**
- 6.1 Comments . . . 29
- 6.2 Website - David . . . 29
- 6.3 Multiplayer - Philippe . . . 30

7 Production of the project **32**
7.1 Pace of work and difficulties 32
7.2 Joys 32
7.3 Disappointments 33

8 Conclusion **34**

9 Appendix **35**

1 Introduction

1.1 The roots of Tempête

The beginning of Tempête

Tempête is a video-game development studio born in late 2018. It is the union of three EPITA students from the same class INT2: Philippe, David and Charlie. We are first and foremost a group of friends which loves to imagine and create. Creating things pleasures and amuses us. As Tempête members, we are forever curious to see to what extent we can fulfill our atypical goals and carry out our peculiar ideas. To make it simple, we are our first and most devoted audience, therefore we believe the creation process is part of the enjoyment. Thus, the goal of the project is to share our games with a broader audience. The project in our opinion represents an opportunity to highly improve our skills and leave our comfort zone.

Through this presentation we introduce our first project: ***Ultimate Fighting Chickenship***

Tempête Group

Tempête has three members. Everyone is versatile, creative and hard-working: those qualities ensure the good listening of everyone's ideas and thoughts about the project. Furthermore, everyone brings a singular asset to the studio. The members are young yet willing to improve their programming skills and their teamwork abilities.

1.2 The idea of the game

Ultimate Fighting Chickenship is a PC fighting game where two characters - chickens - fight each other until one's health is reduced to zero. The gameplay we aim for is similar to ***Street Fighter***'s and relatively close to classic fighting games where the two foes are facing each other. However, we want to enable the players to move, attack and jump faster than what ***Street Fighter*** allows the players to: the goal is to enhance movements and agility. This would be our contribution to the evolution of fighting games.

A major aspect of the game is that it aims to be fun and is, thus, purposely "fatuous". The idea of taking chickens to fight each other illustrates this aspect very well: as we want this game to be multiplayer oriented, we are making sure it is a game a group of friends could play to have fun and relax. One should not be surprised by the unexpected features or artistic choices of the game.

1.3 Goal of the game

As said before, Ultimate Fighting Chickenship is a 1 versus 1 fighting game, multiplayer oriented it is designed to be played online or in LAN.

With a funny environment and a fluid and dynamic gameplay, two players fight using basic attacks such as punches and kicks, abilities such as fire balls and a wide set of movement abilities to knock out the opponent.

2 Book of specifications review

2.1 A short history of fighting games

The first game of the genre is ***Heavyweight Champ*** *(1976)*, a two-player game that was released for the arcades. The game has a simple black and white color palette and consists of two characters shown on a two dimensional plane. The players have a very limited set of attacks at their disposal: each character can shoot high and low punches, the goal being to knock out the opponent before they knock you out.

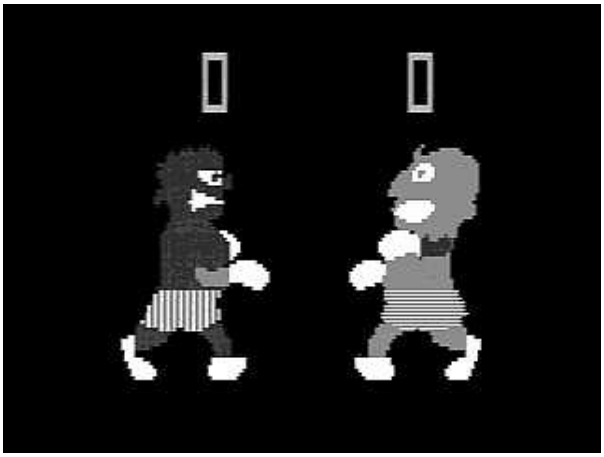

Figure 1: ***Heavyweight Champ***, the first fighting game

The genre as we know it today is the result of an evolution of fighting games. The first game to introduce less crippled movements was ***Karate Champ*** *(1984)* which marked the beginning of the popularization of fighting arcade games. Then, ***Yie Ar Kung-Fu*** *(1985)* introduced a system where every character has their own fighting style.

The first major milestone in the evolution of the fighting genre is the ***Street Fighter*** series, specifically ***Street Fighter II*** *(1991)* which quickly became popular thanks to its great variety of characters and attacks, as each character has as many as 30 unique moves / combos and several special attacks. The game also gained popularity thanks to its wide availability on arcade as well as home gaming consoles.

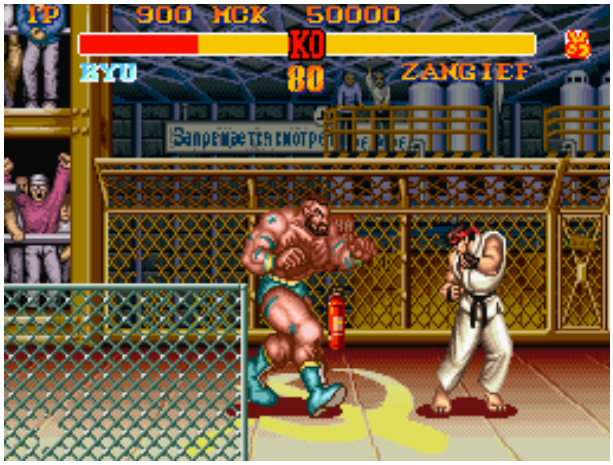

Figure 2: ***Street Fighter II***, the most popular fighting game

As time went on and technology got more advanced, newer games similar to Street Fighter appeared, the most important in the scope of our project being:

- ***Mortal Kombat*** *(1995)*, for its usage of violence and special attacks (*fatalities*),
- ***Super Smash Bros.*** *(1999)*, for its fast-paced gameplay and percentage-based damage mechanic.

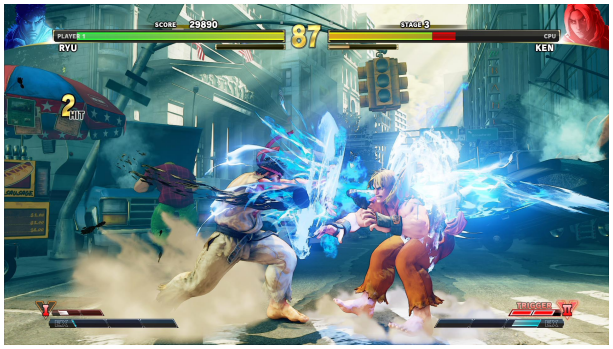

Figure 3: ***Street Fighter V***, the most recent Street Fighter game

Review

With patience, inspiration of the great games of the genre and much documentation, we are proud to have created a game that not only has the codes of a fighting game but also assimilated them to innovate.

2.2 Artistic direction and philosophy of the Project

The main guideline we wish to follow is to keep the visuals simple and uncluttered. The assets will consist of simple colors, with photo-realistic touches. The final goal is to get a consistently "wacky" style, with inspirations from the ***Raving Rabbids (Les Lapins Crétins)*** games and the ***Deadpool*** movies. UFC is meant to be funny and unexpected. Moreover, we wish that not only the gameplay, but also for the visuals, sounds and music to play a fair part in the player's enjoyment.

Review

Until the end of the project, we kept developing this game with this philosophy in mind. Yet with the loss of Maxime Mugniot of our group during the year, designing characters and maps was a time-demanding task that was not primary.

Yet we did not lose face and kept on developing the game after reviewing our priorities. We decided to make only one character and make sure that with this one we could show everything we could do.

The gameplay looks pretty much like the final version we imagined. The gameplay is dynamic and fluid as we wanted. Furthermore, all the characteristics were added. Of course, only having one character makes it impossible to have different types of characters.

The attacks also match our expectations. There are Power-Ups, basic attacks, Projectiles and Combos. The implementation of combos were a real challenge yet we are proud of the result.

Not suprisingly, the Sound Design was the most rewarding part. Audio is really important in a game and thus it brought life to our project. The plurality of sounds and voices went beyond our expectations.

2.3 Technical aspect

The project will be brought to life with the following software:

- **Blender**: 3D modelling software,
- **Unity**: game engine and all-in-one development utility,
- **Git and GitHub**: for collaborative work.
- **Ableton**: music making software to create our soundtracks.

The reports as well as the book of specifications will be done on Overleaf. Communication is key in a group project, thus we will use a Discord server that we created.

The project will also include 3D animations. In order to implement them, we will use **Mixamo**, a website where thousands of animations are available for free use.

As documentation, we will use the Unity forums as well as YouTube channel Brackeys. Other sources such as Stackoverflow might be of help.

2.4 Economic aspect

In terms of economy, we do not intend to spend any sizeable amount of money into the project, as we already have all the software we require. The game will be free of cost. All members of the group have friends that want to test our game. If they like it, they may share it with their own friends, who might do the same thing and so on. This is how TEMPÊTE will extend its community, and how the distribution campaign will start.

2.5 Task distribution

	Philippe	**David**	**Charlie**
Game mechanics			
Movement	⊕		✓
Basic attacks		⊕	✓
Combos	✓	⊕	
Sound design			
Music	✓	⊕	
Sound effects	✓		⊕
Voices		⊕	✓
Graphics			
Map design	⊕	✓	
Character design	⊕		✓
3D Animations	⊕	✓	
User Interface		✓	⊕
Other			
Multiplayer	✓		⊕
Website	⊕	✓	
Artificial Intelligence		⊕	✓

✓ stands for "In Charge", ⊕ stands for "Substitute".

2.6 Timeline

Presentation	**1st**	**2nd**	**3rd**
Game mechanics	40%	70%	100%
Movement	70%	90%	100%
Basic attacks	25%	60%	100%
Combos	0%	30%	100%
Sound design	20%	50%	100%
Music	30%	60%	100%
Sound effects	0%	50%	100%
Voices	0%	50%	100%
Graphics	20%	60%	100%
Map design	50%	75%	100%
Character design	15%	60%	100%
3D Animations	15%	60%	100%
User Interface	0%	60%	100%
Other	40%	70%	100%
Multiplayer	50%	90%	100%
Website	30%	60%	100%
Artificial Intelligence	0%	25%	100%

3 Game Mechanics

3.1 Comments

Charlie took care of the major part of Game Mechanics receiving Philippe's help for the multiplayer and network programming and David's help for assembling all parts.

Implementing Game Mechanics into the project was our first contact with Unity and, as a result, took a long time to be operational. We first had difficulties understanding how C# programming changed from our Programming's projects.

Indeed, programming movements and basic attacks were not instinctive at all and really different from what we did before. This is mostly due to the significant amount of features Unity provides the user to write scripts or to configure the scenes.

Having difficulties taking a start on the project, Charlie took a few time to think and build himself an idea of how he would conceive each and every mechanic. This is how, during the first phase of the project, he established a good base for the following implementations.

Then the whole group had become more familiar with Unity, its basics features and C# programming in this environment, thus Charlie started working for the second phase of the project with more confidence.

We understood at this point that few features such as Ultimate attacks would not be possible to implement. Without being overwhelmed by this sign of weakness, we kept on working to make our game as close to the expectations as possible.

Of course, he still had numerous features to discover and learn how to handle. Thus, he entered a more advanced phase of video game developing.

After fixing few features, he started adding complexity to the gameplay. Switching between reflexion and code, he tried to find the best way to implement each task in order to optimize the game.

Yet, patience and perseverance were involved in this process because everything did not work on the first try. However, with much documentations most of the goals have been achieved during the second phase.

At the end, with much documentations, tutorials and patience (because of the bug resolving), we pretty much achieved what we wanted to do.

3.2 Movement- Charlie

1st deadline

For the first presentation, a Ultimate Fighting Chickenship player was able to realize two different movements: jumping (Space bar) and moving left or right (Q/D keys).

In order to implement movements, Charlie created a C# script called Player Controller (see figure 4). This script took care of the physics laws applying on our players : gravity and drag. Back then the players are represented by two cubes : a small one representing the head and a bigger one for the torso. By using Unity RigidBody and changing its position according to the inputs of the user : the player can move pressing the specified keys.

For the jump movement (see figure 5), we also implemented a boolean variable checking if the user can move or not (regarding its position with the ground) : if it is on the ground it can jump. This is to prevent the user from flying.

With a prefab, a problem occured: start positions (or spawn points) had to be created. Indeed, both players started the game at the same position. This is why we added two network start positions : the host will spawn on the first one and the client on the second one.

2nd deadline

In addition to the already available basic movements, Charlie implemented a dash movement, the ability to double jump and the rotation of the player on each turn for the second presentation. Thus, a player had and still has multiple movement possibilities during a game of Ultimate Fighting Chickenship. All these changes were implemented in the C# script Player Controller.

At that point, the game was not far from its final form in movement possibilities. These three main new movements were more complex to implement as they are not as basic and common as moving left, right or jumping.

The double jump (see figure 6) has been implemented using a jump counter and a reset check. Every time one touches the stage ground the jump counter is reset. On the other hand, each time one jumps pressing down the Space bar, the counter is incremented by one. Finally, the counter must be below a certain limit to enable the player to jump (the limit is 2 for double jump). Another system would have been to implement a boolean array where each element checks for a specified jump. However, this system is too bulky and won't allow simply to increase the limit of jumps. An important condition is to check for the Space bar input when it is **down**. Otherwise, there is a risk that the key is pressed during more than one frame and thus the counter will reach its limit too soon.

Furthermore, the player turns towards the direction it is moving (see figure 7). Charlie implemented it using a boolean informing on the direction the player is going. Then on each left/right movement input the rotation is changed if needed. A problem was that it also inverted the internal axis of the character, thus the movement vector had to be inverted.

The dash movement works like a small teleportation towards any direction. The 'dash' does not change rotation and only provides an additional mechanical possibility for the player. It is quite complex to perform one, we aim to make it a move accessible to trained players. Lastly, the 'dash' demands energy called "stamina" to be used: we will get back to this new system.

Those movement abilities enable a more fluid and fast gameplay. The experience is far more dynamic.

3rd deadline

By the time of the second presentation, we almost already finished creating the movement abilities. As a result, for the last deadline, we made the scripts clearer, optimized the implementation and adapted the values of ground speed and jump speed to match our expectations around the game's dynamics.

Charlie also created a new movement ability called Protect Move. By pressing the matched key, one is able to enhance its defense abilities. One can not move nor attack as long as one presses the key. This ability provides a discount for any attack that hits one's character, it reduces the damage by a percentage (80%) that decreases by 15% every second as one keeps the protection on. An interesting feature to take care of when using this move is that it demands a lot of stamina to prevent players from turning into cowards.

The implementation of such a move demanded preparation and reflexion, yet there were no bugs afterwards. Charlie demonstrated the patience and experience he acquired during those 6 months of game developing. Specifically, this move is two variables implemented in the Fighter C script, a boolean value changed whenever the key is pressed and released and a float representing the damage reduction percentage. Putting this percentage in a variable allowed Charlie to make it decrease in the Update function called every frame. Finally the Protection Move avoids the dizziness periods after being hit.

3.3 Attacks - Charlie

1st deadline

For the first presentation, Charlie took care of the implementation of the first basic attacks : a punch and a kick. The method used is : an array of BoxColliders (boxes able to detect collision) was added to the player prefab as well as a Fighter C# script.

Each of those boxes represent a part of the body able to hit the opponent. The Fighter script contains a unique method LaunchHit taking into parameter the part of the body involved in the attack. The method detects every collision (OverlapBox method) with this part of the body and calls a function to make the other collider take damage. One can only attack by touching the Torso or the Head (multiplying by 1.5 the damages), the "Attack Boxes" do not act like "Hit Boxes" which means they can not take damage for the whole body.

For instance, at that point we only had a punch and kick dealing different amount of damage. If one 'punches' one's opponent : if it hits the head or the torso, the opponent will take damage. But you could not take damage by being hit on the arm.

Of course, the inputs were updated in the Player Controller script : by pressing the specified keys, one can punch or kick.

As a result, a Health C# script has also been added to the prefab. One is able to take damage and a system of health points proper to each character has been implemented.

2nd deadline

To match expectations of the second deadline, there were a lot of changes in the system of attacks. Keeping the BoxCollider system for the attack boxes, a new C# script Attack was created. This script only contains a structure Attack establishing its components and its constructor. Thus, the Box array was replaced by an attack array in the Player Controller script.

On the specified input, an attack is chosen from this array and calls the LaunchHit function in the Fighter script. Attacks have multiple variables :

- *name* of the attack, at this point 'punch' and 'kick'
- *damage* it deals on hit
- *hitbox* : the attack box it uses to trigger the ennemy's body
- *knockbackX* and knockbackY: the knockback components
- *staminacost*: the cost in energy

Charlie also implemented a method GetKnocked() called with the attack's knockback components as a vector each time the player is hit. This method constitutes in famous mechanic functionality of fighting games: the Knock Back or Push Back. When one is hit, one will be pushed away in a certain direction depending on the type of attack.

One of the major novelty was the creation of a new type of attacks: the abilities. We aimed to have different types of abilities. At that point, Charlie implemented the Projectile type of abilities in a new Projectile C# script. Projectiles almost have the same components as Atttacks yet it is Network-Behaviour inherited Class. It takes into account three new variables:

- *VelocityX*: the constant velocity at which the projectile moves
- *Prefab* to be spawned in the game
- *SpawnTransform*: the position where it will be spawned

Projectiles have their own collider that will detect collision and call On-CollisionEnter() method. This method decreases the health in the HealthBar script and destroys the projectile on hit. Projectiles deal much more damage than basic attacks and offer the advantage of the distance. However, they can be easily dodged and cannot be used as often as punches and kicks. The Player Controller script was updated to have an Projectile array and to be able to call the LaunchProjectile method in the Fighter script on the right input. This method was implemented as a Command. It means that it is a call from a client to the server. Indeed, to make sure the projectile spawns on

each client, the server must be in charge. At that point, Ultimate Fighting Chickenship had one Projectile Attack: a Fire Ball (see figure 8).

3rd deadline

For the final deadline, we aimed to add combos and new attacks.

First, we added new abilities. After introducing Projectiles in the second presentation, we created a new type of abilities: the Power-ups. Power-ups enhance a characteristic and allows one to have better skills in melee or distance combat.

All Power-Ups have their own way of working, some work along time, others last for few actions. Every Power-Ups have their own way of being performed: they need a specific combination of keys. Furthermore, they cost more stamina than a basic attack but still less than a Projectile. We added multiple Power-ups, here are few examples.

- The *Strength Up*: a strong Power-up that will increase one's damage, costs a lot of stamina and lasts for five attacks.
- The *Speed Up*: lasts for a certain distance, easy to perform and does not cost much stamina. Great to surprise the opponent.
- The *Jump*: allows one to make a triple jump. Does not cost much stamina.

Power-Ups represent the first combos we implemented in the game and demand a certain experience to be performed in-game.

Second, we added a new Projectile Ice Ball that slows the opponent on hit. To script it we check for the hit, reduce the speed variable and block the jump counter to 2 for the duration of the effect.

Finally, all the hitboxes were replaced by the real 3D modelled character implemented by Philippe.

3.4 Other mechanics - Charlie

Stamina system

A whole new system was implemented for the second deadline: the stamina (or energy) system. Some actions cost stamina and can only be performed if the current stamina points amount is sufficient. Players have start at a maximum amount and cannot go above this value.

All attacks and abilities have a stamina cost, the dash movement also costs stamina. The stamina was updated by calling a Command taking a float in arguments: the current stamina amount is decreased by this float and the stamina bar is updated.

Additionally to these punctual calls, this command was called each frame by the Update() event to increase the stamina amount: in that way it acts like a regeneration. The amount of stamina regenerated each frame is based on the number of frames by second of the game to make sure the regeneration amount is the same over time in any environment:

regeneration = Time.deltaTime*10

User Interface and Experience

Charlie implemented a stamina bar (see figures 9 and 10) that updates during time. He integrated it with the Health Bar. The objective was to make them both work on network so that both players can see the health and stamina bar of the opponent. It has been done using the SyncVar attribute of Unity enbabling the synchronization of the amount of health and stamina and of the update of the bars. Both bars were implemented manually on the player prefab to make them appear above the player. Another script has been created for both bars to make sure they do not rotate with the player prefab.

A script ExitGame has been created to come back to the Main Menu. Pressing the Escape key leaves the game.

For the next deadline, we wanted to create a protection move: by holding a key one will reduce the damages taken during the holding, it will cost a significant amount of stamina. One will not be able to use any other attack or to move during the protection

Cooldown & dizziness

For the final deadline, we aimed to create dizziness after each hit and attack. In addition to the stamina, we wanted to create a cooldown for each attack to prevent the players from spamming.

First, we implemented a very simple cooldown: the duration after an attack for it to be available again is the time of the animation, nothing more. After reflexion, Charlie and his fellow members of TEMPÊTE thought about cooldown in fighting games. It occured to them that most of fighting games did not have such system. Furthermore, we thought that stamina system was quite enough to keep the players from spamming.

Furthermore, to avoid having a messy gameplay, Charlie added the dizziness feature. Dizziness represents the time after taking a hit when one can not do anything. Each time one takes a hit, one's controller is frozen for a duration that varies along with the damage taken and the knock back of the Attack or Projectile. Adding this feature was for us a way to make the game harder and more realistic. It would have been ridiculous to see a game where one is insensitive to attacks.

4 Sound design

4.1 Comments

We knew from the start that the creation of the audio would be one of the most rewarding parts of our project because we knew it would bring life to our game. It is a nice way to express our personality and share our emotions, make the user experience emotions. Luckily, the implementation of the audio sources into our game was one of the most seamless we experienced. Also it was the first step in establishing the burlesque atmosphere we wanted our game to have.

4.2 Music - Philippe

1st deadline

Philippe has had the main role in sound design, and particularly in the music production. He used his previous experiences in music creation to build the main theme of Ultimate Fighting Chickenship. Along with his musical genius, he mainly used Ableton (see figure 11) along with his iPhone's microphone to record the sound-effects.

The music is essentially 1min20 loop that repeats itself. 1min20 is enough time to have interesting variations but short enough to ensure a small music file.

The standout element in the song is the chicken shouts repeating themselves in a rhythmic fashion. Along with these shouts are epic percussions with fast strings to add rhythm. He added fast-paced Hi-hats to bring out a hip-hop vibe to the song. A flute can also be heard. In the second part of the loop, the epic percussions switch with trap drums, which helps break the monotony of the loop and enhance the hip-hop vibe we wanted the main song to also have.

With a main theme done and a musical direction that is now fully defined, the amount of work needed to produce the rest of the game's music will only be facilitated.

2nd deadline

During the second period, Philippe has kept on crafting sounds in order to provide players a more fulfilling experience. He made the music that would be heard during the fights, the music that would best accompany most of the player's time spent in the game. The process is the same that the one for the main menu's music: Philippe made a 1min20 loop using Ableton.

The idea of the song came from listening to "Praise The Lord" by A$AP Rocky. The fascinating flute sample playing through the whole song stuck in our heads. We decided to use the very same sample and make a remix of the song. In the first half, percussions similar to the original song's allow us to make a pop-culture reference as well as extending the hip-hop elements brought by the main menu theme. The second part of the song, however, sets itself apart from the rest of the music. It contains almost exclusively electronic music elements and are a full part of the genre-bending mindset Philippe had when crafting the game's music.

3rd deadline

The third and last music of the game uses the same techniques as the first two. The third music is the one that will be heard in second by the player. It covers the second scene of the game: when the players it will connect players to each other in order to play a match.

To best fit the context, Philippe decided the music would be a simplistic loop. Mainly composed of rhythmic elements, similar to what can be found in the main theme music that can be heard during the main menu. The simple 1min loop, although minimalist, is supposed to hype players going into their match and serve as a transition between the main menu's music and the match's music. Reverb on the percussion us used to make the listener believe he is in a big arena, make him feel like a gladiator.

4.3 Sound Effects - Philippe

1st deadline

We knew sound effects would take place only later in the game's development because it is hard to attach sound to only a skeleton of the game. Knowing that, we didn't work much on the sound effects of the game for the first months, but the members still exchanged their ideas on how the game's sounds would be recorded and what type of sounds would be absolutely necessary. Since the beginning, we knew that our sounds would be recorded in an amateur fashion and be only lightly treated.

2nd deadline

The second deadline marked an achievement in terms of sound design with the appearance of the short sound effects.

With the help of his and Charlie's voices, Philippe established a large sound bank, containing almost a hundred sounds. This number was expected to grow as we kept on adding functionalities to the game. Sometimes sprinkled with reverb or distortion to better fit the context, almost every action in the game has its own sound-producing function in its script. It uses unity's random function to pick a .mp3 clip from the array of clips each action possesses, allowing the game to be ultra dynamic and constantly surprising.

3rd deadline

Philippe has made even more sounds for the game's actions and added them to the sounds array triggered by the random script.

5 Graphics and User Interface

5.1 Comments

Graphics/UI is obviously one of the main points that was completely shaken due to the loss of a team member. We had to reduce the goals we had set, in character design / modelling as well as map design for the first deadline, a delay that has persisted throughout the project. For this reason, the graphics part is not as advanced as we had aimed.

5.2 UI and Design - David

David was in charge of the user interface and design. He created the main menu and the different elements of the user interface, as well as the logos for the game. His advancement throughout the year is described below.

1st deadline

The user interface is a point that has evolved a lot during the development of the game. For the first deadline, the user interface was just comprised of a main menu that consisted of a blank screen that contained a play and an exit button, the main menu theme looping in the background (see figure 12). The game did not yet feature a logo.

2nd deadline

For the second deadline, the main menu was modified to feature a play, exit, and options buttons, with the main theme of the game still playing in the background. Every time a button is pressed, a sound is played before doing the action the player requested. The main menu was also redone visually, with a better background and more stylish buttons that blend into the backdrop. The inspiration came from a photograph of an old American fast-food chain called Burger Chef. David edited it as to remove any proprietary content (see figure 13). As of the second deadline, the options button is a placeholder.

Along with the main menu, a logo was designed for the game, which comes in two variants:

- The first is a 2-line design, that is featured on the website's top banner and the game's main menu (see figure 14),
- The second is a 3-line design, which is meant to be used in smaller spaces as its resolution is more square-shaped (see figure 15).

The logo was designed to be striking and fun by the choice of the fonts and colors.

On a side note, some buttons were assigned in-game to go back to the main menu, but their implementation was not final.

3rd deadline

For the final deadline, the options menu was added with an option to change the resolution as well as the volume level. To fluidify navigation between the different menus of the game, a pause menu was added with two "Resume" and "Back to main menu" buttons. As the game is primarily played over the network, the pause menu doesn't actually pause the game.

5.3 3D modelling - Philippe

1st deadline

Very few work has been done in the first period of our project because it was first attributed to Maxime. Our time wasn't gave to this part of the project in the first months. Obviously was the better choice since it was maybe not as important to have a complex 3D model in the first presentation as having a character than can move around and a multiplayer functionality.

2nd deadline

Philippe took care of the 3D modelling of the character. He learned how to use Blender, a free 3D editing software. He first designed a basic 3D humanoid character (see figure 16). He wanted to use a humanoid style of character, because it would facilitate the uses of the model in Unity, because it offers presets for humanoid models. He still wanted the model to look like a chicken, so he then buffed his thigh but kept thin leg bones, replaced the human feet by chicken feet and put somewhat of a chicken-human mix for the head. With this first character done, Philippe hit a landmark, because

making the next characters would be a breeze since we will only changes its textures.

3rd deadline

Philippe then imported the model into Unity and simply scaled it to the right size so that the player Prefab has a perfectly shaped and sized model. The player can now admire and control a basic character in his game.

5.4 Animations - Philippe

1st deadline

Without any 3D model, it was logical that no animations was made during the first period of the project besides light research on how to implement the animations with the character's moves and attacks.

2nd deadline

As for the animations, Philippe first constructed and placed the bones of the character. He then proceeded to "weight paint" (see figure 17) the 3D model to make the animations deform the character in a natural fashion. He made two basic animations: a standing pose where the chicken's arms move slightly, and a side-walking animation, where the chickens make side-steps in a bit of an on-purpose ridiculous fashion to enhance to funny side of the game.

3rd deadline

As for the last period of work, Philippe has implemented the 3D model's animations made in blender into Unity. He used the animations system provided the software to make transitions between animations states. When the model stops, the animations changes to a resting animation.

5.5 Map Design - David

1st deadline

For the first deadline, David completed the first map, Krispoul's one. The work was done with Photoshop, a video from Ionisx and icons from the

website flaticon.com. The main focus was to remain faithful to the legendary backdrop that appears in the Ionisx videos. Therefore, special attention was given to the making of the green fade and the placement of the icons on the left, some of which had to be remade from scratch as they were not available online. (see figure 18)

2nd deadline

For the second deadline, as we had to advance deeper on other aspects of the development, map design was limited to research, and testing design ideas in Photoshop and Unity.

3rd deadline

For the final deadline, the last map was added, Git Poule's one. It features a backdrop that imitates the looks of a shell. Once again, the work was done in Photoshop and was minutious to assure the best quality. This gives the game two playable maps.

6 Miscellaneous

6.1 Comments

In the first weeks of making the project, we thought that multiplayer would be quite easy to integrate to our game and would not change that much from a single player game. We were wrong. Not only every functionality of our game needed to be coded to work in multiplayer and needed a full understanding of Unity's networking features, which took most of our time, but these features were outdated during the making of our projects. Our dependency on Unity has had a negative impact on our project, especially on the multiplayer department.

Yet we are glad we have implemented the network early, it allowed us to have an idea of the final result. Furthermore, implementing it too late would be a mistake, it would have completely changed our project and would have forced us to rewrite and re-adapt every single script.

6.2 Website - David

1st deadline

David began working on the website as soon as the project started. Using his previous knowledge of HTML and CSS, he put together a simple homepage that was updated as the project progressed. The website began as a simple set of empty pages, that was transformed into a real layout:

- *Home*, a few paragraphs to introduce the website, the project and the team,
- *Project*, a page where we post advancements, screenshots and other various news for our community to see,
- *Download*, a page that will host download links to the game and various reports as they are completed and ready to be published,
- *Links*, a page that will host all information and media that was used to create the game, such as sources for images, designs, models and so on.

For the 1st deadline, only the first two pages were created.

2nd deadline

For the 2nd deadline, David continued working on the website, which was almost completely finished. The website featured all of the four pages that were specified in the book of specifications (Home, Project, Download, Links), including the last two that were not yet completed. From this point on, the only thing to do is to populate these pages with content, a process that has already started. With input from every member of the team and following the progress that was made in each field of development, David has written newsletters that are destined to inform the community of the progress that the team is making in advancing the project.

3rd deadline

Approaching the final deadline, the website was already complete for the most part. More content was added as the development progressed, and some technical details were polished. (see figures 19 to 22)

6.3 Multiplayer - Philippe

1st deadline

The first thing Philippe has done regarding the multiplayer implementation of our game is making heavy researches on every possible way to implement it. He also looked at Unity's competitors and which company had the best free offer. He then chose unity's free plan because it was the most commonly used worldwide and therefore plenty of tutorials and information could be found online.

Also, implementing the multiplayer from the start of the project is the best solution regarding the time it would have took to convert a single player game to a multiplayer one, because the way information is transmitted is entirely different in a multiplayer game. This way, we started by making a multiplayer game, and if someone wishes to play in solo, he can host his own local server natively.

Our game scene is containing a GameManager Prefab that handles the networking abilities of the game. This prefab contains the Network manager

component as well as the Network Manager HUD component. Network manager component handles the multiplayer and HUD component allows us to have a premade menu with basic features such as LAN Host and LAN Client.

Then each object in the scene that has a server-related comportment must have a Network Identity Component, and a Network Transform component.

- Network Identity will define if the object is server based or if the client will have authority on the object
- Network transform synchronize the object with the server. It has parameters for the frequency at which the information is sent or what the movement threshold is.

Before doing any action, the object must verify if the client has authority over it. This way, the player can only move his character.

2nd deadline

Philippe has done little work on the overall structure of the multiplayer functionality of the game in the second period of the project. Hand in hand with Charlie, Philippe did little adjustments. Features such as the health bar have been fixed from the first presentation. The game still features the unity original network's GUI that we planned to change for a custom one by the time of the third presentation.

We also fixed a few problems regarding the synchronization of movements between the characters that occured before the first presentation. Charlie and Philippe also encountered bugs regarding the UI: health and stamina bar would not decrease the same way for the client and the server. Fortunately, we were able to fix those problems before the deadline using the Commands (see part about Game Mechanics).

3rd deadline

Because the multiplayer functionality was already fully working in the second period of our project, this month consisted only in researches for network efficiency as well as a big chunk of bug resolving.

7 Production of the project

7.1 Pace of work and difficulties

Through those six months on the project, we encountered several difficulties. Indeed, few nights were spent trying to fix a bug on Unity or figuring out how to implement one mechanic or another. For instance, the most difficult part to understand was the implementation of the network on our Unity Project. It took a night to fully understand how scripts, players and features would get along online. Those difficulties and challenges altered our pace of work.

Our pace of work has been intensive, every member gave everything to create and add more features to the game. Yet spending hours on Unity tearing off our hair in front of a bug was not always the solution. Outside of the rushes period, we usually spent time gathering new ways of programming, information about fighting games and fresh ideas. As a result, there were no time wasted. Establishing such a pace of work was a big step toward the completion of our project.

7.2 Joys

The first joy the group had was to be able to work in such a great environment. Supporting each other and gathering ideas really helped to bring the project its final stone. Now, we are glad the game works and looks like what it is supposed to be: a game. It is a great win for each one of us who were beginners at every task we had.

What certainly makes us proud is our ability to have finished this project on schedule despite the challenge it represented at first. We were complete beginners yet mastering numerous skills in programming, game designing, graphics and even for the website allowed us to produce a finite product. Although it is not perfect, our project is not a work in progress, it is a playable fully operating game.

We are also happy that despite the loss of one member of our team, we were still able to keep up with the pace of the deadlines and compensate enough to keep things running smoothly throughout the making of our projects and our presentations.

A main success is that we achieved to create a fighting game that has a fluidity and a rapidity of gameplay between our models: Super Smash Bros. and Street Fighter. One is free of its movements and yet is not ejected out of the map after being hit.

The whole team is also glad to have established this "fatuous" atmosphere in the game. The voices and sound effects really helped and of course the chicken-based idea of the game was a good start. Creating something we liked, a project that made us laugh, was much easier and contributed to our motivation.

7.3 Disappointments

Character selection, to us, is essential to the fighting game genre. This can be observed with games like Super Smash Bros. One of our disappointments was maybe not having shifted our focus in the right places at the right times. We know that if we focused on it a few hours more we maybe would have had more characters and a better map diversity.

8 Conclusion

As members of Tempête , we have been united and invested in the project. With high ambitions and a strong belief in what we could achieve, we created a game which means a lot to us. We all played ***Street Fighter*** during our younger years, along with ***Super Smash Bros.*** Drawing inspiration from classic games that made the genre what it is today, ***Ultimate Fighting Chickenship*** has been designed to be the craft of our very own fighting game.

9 Appendix

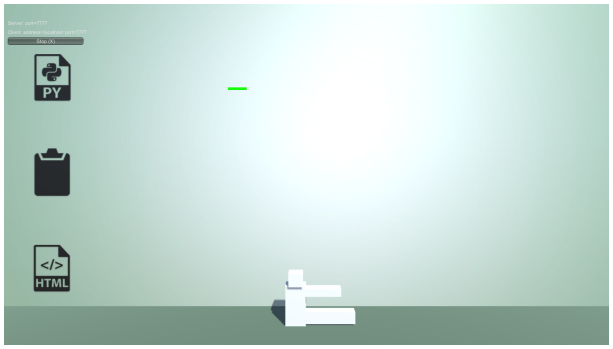

Figure 4: Normal position

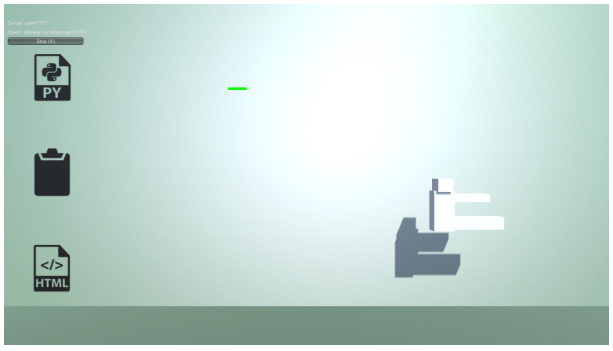

Figure 5: Jumping

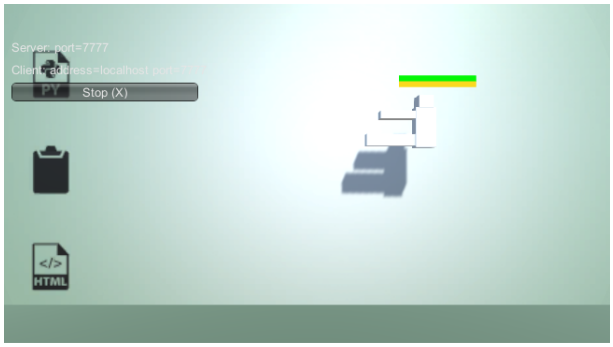

Figure 6: New double jump height

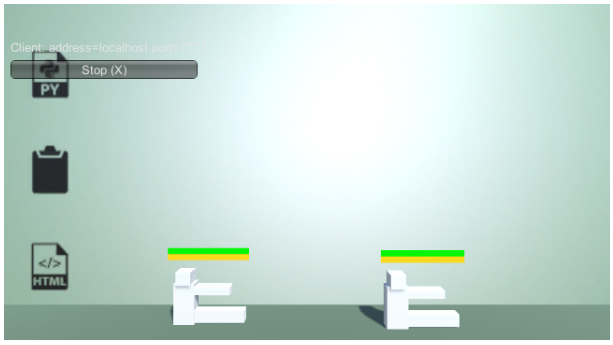

Figure 7: Both players face the same direction after rotation

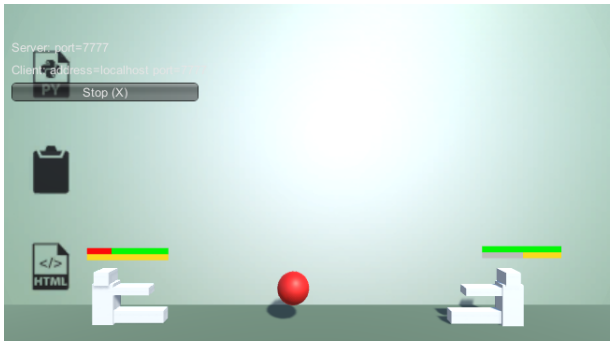

Figure 8: A moving fireball that will damage player

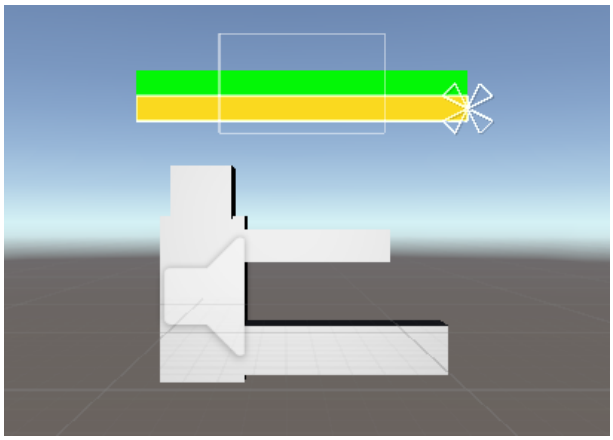

Figure 9: A full stamina bar in player prefab

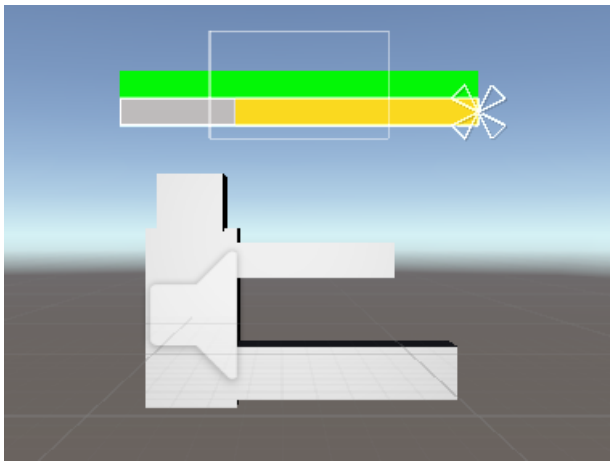

Figure 10: A filling stamina bar

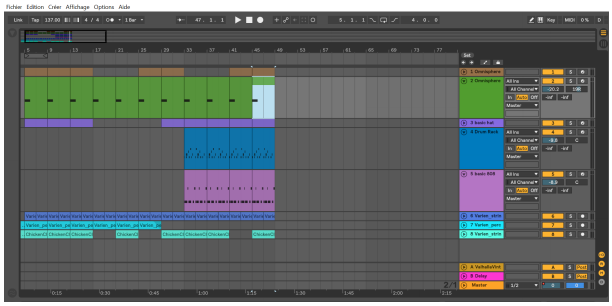

Figure 11: The creation of the soundtrack in Ableton

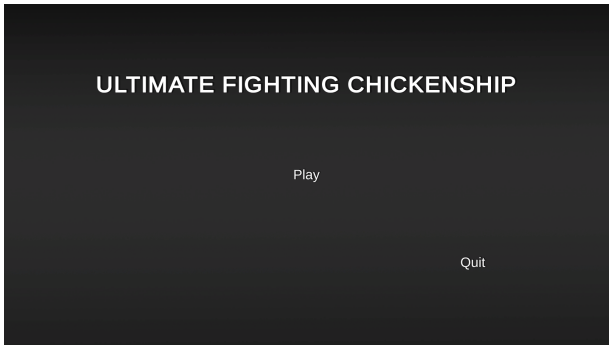

Figure 12: A screenshot of the main menu as of the 1st deadline

Figure 13: A screenshot of the new main menu

Figure 14: The two-lined logo

Figure 15: The three-lined logo

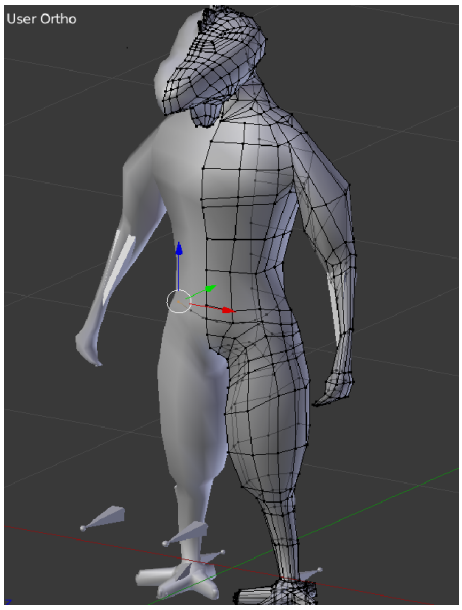

Figure 16: The creation of the model itself in Blender

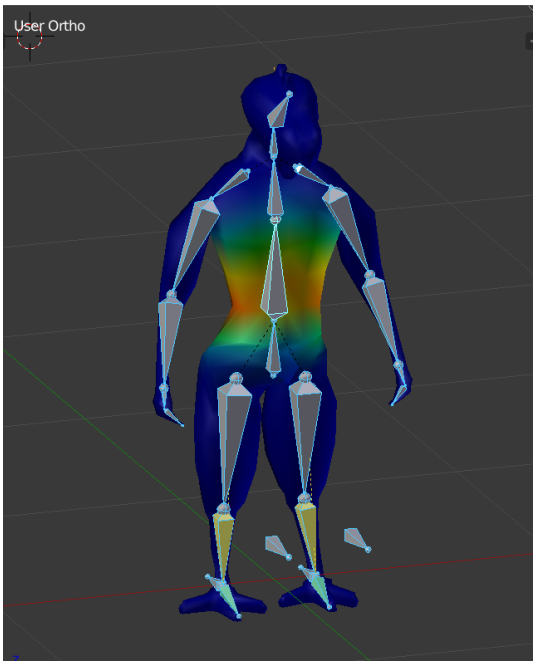

Figure 17: The weight painting

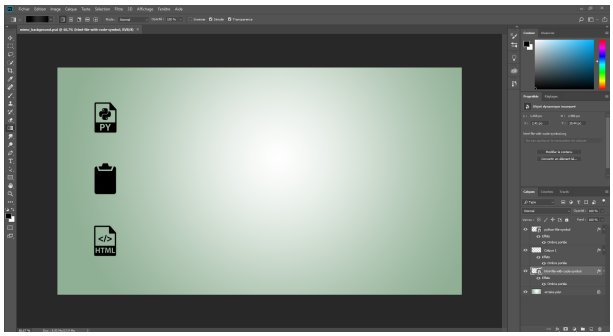

Figure 18: The creation of the Mimo background in Photoshop

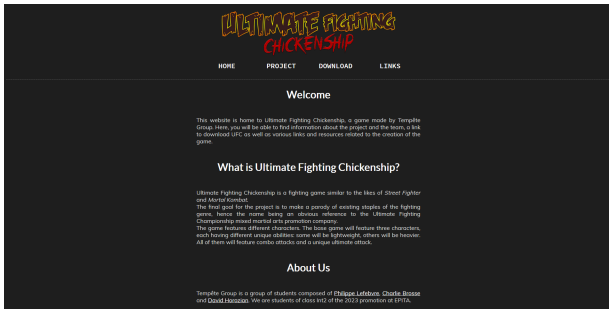

Figure 19: A screenshot of the home page as of April 2019

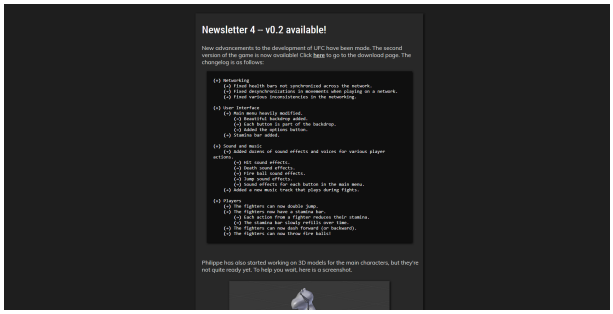

Figure 20: A screenshot of the project page as of April 2019

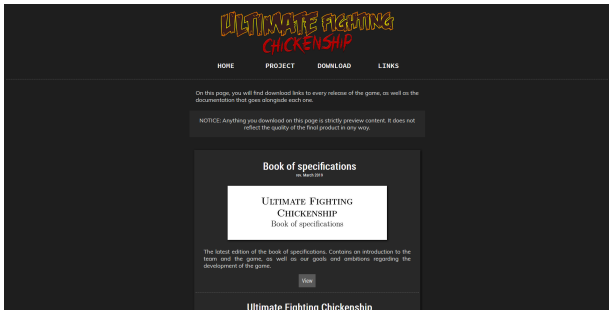

Figure 21: A screenshot of the download page as of April 2019

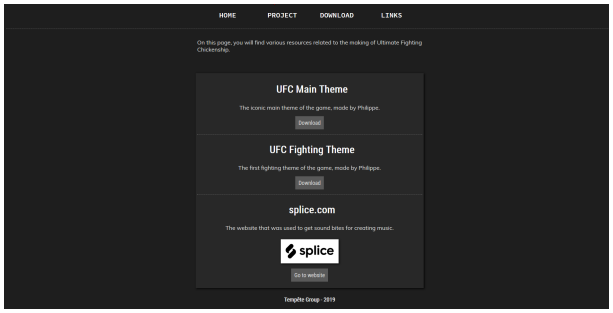

Figure 22: A screenshot of the links page as of April 2019