First presentation - Project Rukmaksii

Romain LE MIERE Lucas TILLY Alban NAULIN
Emmanuel VOUILLON

S2 project

Contents

1 Introduction

1.1 Origin and nature of the project

Our project is a video game, more precisely a MOBA (Multiplayer Online Battle Arena) with a TPS (Third Person Shooter) perspective. With it, we wish to create an experience centered on three players versus three players matches to fight for the control of objectives. We want to create a game where strategy, good positioning and accuracy are key to success and which is very dynamic, even for the prejudice of realism. As for the general ambiance and artistic direction, the game will be held in the not too distant future and the scenery will be composed of both abandoned buildings and a jungle environment. We do not plan of creating a rich storyline around our game, focusing instead on the gameplay and leaving the player free to imagine the surrounding world. The game was inspired by many games we enjoy playing in our free time such as Anthem, Paragon, League of Legends, SMITE or Overwatch.

Figure 1: Screenshot of Anthem's gameplay

1.2 Narrative of the realisation

1.2.1 Technical workflow

To keep a clean workflow, we decided to keep 2 main branches:

- develop: the base development branch shared among the members and supposed to be kept clean (without any bugs) at all time
- master: the branch kept for releases merged before every presentation or when a humongous feature has been implemented

Furthermore, we opened some issues every end of the week to specify what we had to do for the following week: an issue for each feature. With every issue opened, a branch was created, rebased on the branch develop, and added to the Github project board.

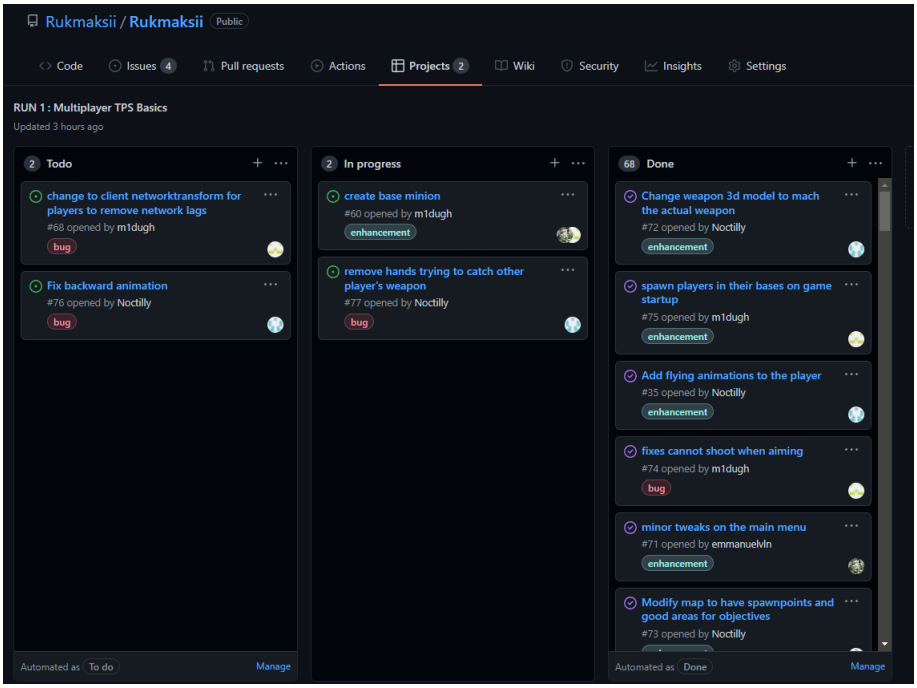


Figure 2: Screenshot of our project board with the opened and closed issues alongside their status

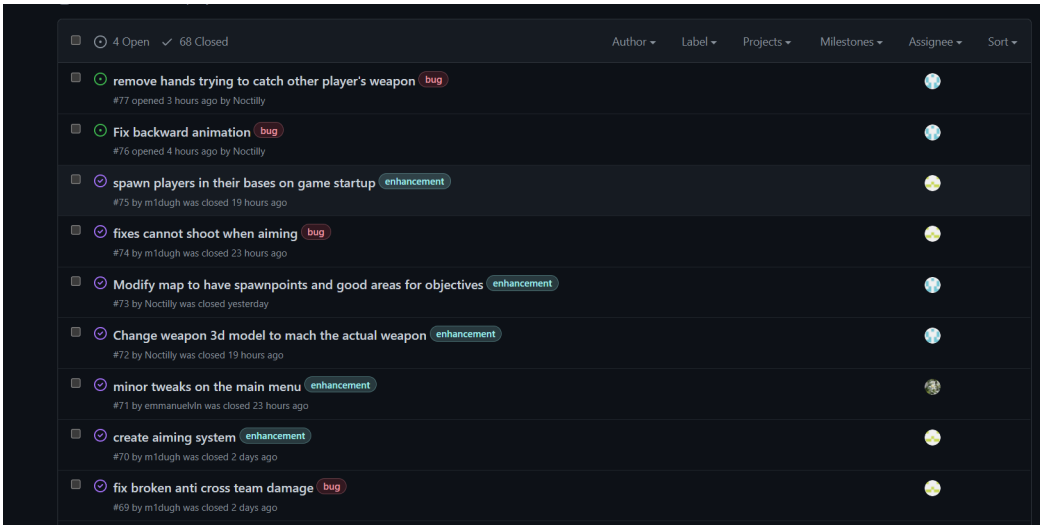


Figure 3: Screenshot of the issues opened for the project

To avoid merge conflicts on the branch develop, we often rebase the branch we work on, named after the issues (e.g. bug#42). When it comes time to merge, we first merge with develop on our branch before merging in the develop branch. This allows to resolve merge conflicts outside of the develop branch, leading to a clean workflow.

1.2.2 Experiences

From the beginning of this year, we worked every Friday night together. Since we have great consideration for machine rooms we ate in the corridor in front of the class. We code every Friday for at least 7 hours.

Figure 4: Photograph from one of our late night coding session

2 Advancement

2.1 AI

The AI will be implemented in the minions which will be summoned by the player. The monsters of the jungle will also have some kind of AI in order to attack the player.
For the moment, the monsters spawn all in about the same place. They are able to chase a player from a distance and if they catch up the player they inflict damage. If the player manages to escape, the monster returns to its starting point. For the next presentation it will be necessary to add a texture, and add an attack animation. And maybe adapt the life points they inflict and their life points, as the game progresses.
For the minions, we have an action model that we will implement. The minions will be based on the following model:
Mission:
Does its actions in a circular range around a reference point defined by the switch case.
Its actions:

- If nobody is in the range, it does nothing
- If an enemy is in the range, it attacks him
- If a teammate is in the range:
  - If the teammate is low life, it heals the teammate
  - If the teammate is fighting, then there is an enemy in the range, so it attacks the enemy
  - Otherwise, it does nothing

Switch case:

- "onPoint" : does the mission in relation to a point on the map
- "onMe" : does the mission in relation to my position
- "onTeamMate" : does the mission in relation to the position of a teammate

2.2 Multiplayer

The multiplayer section is about the implementation of local and online multiplayer.
All multiplayer has already been implemented using unity supported **netcode for gameobjects** preview package. Due to the fact that the package is still in preview, there may be some lags in the game. Still, as the project will go, the package will go on getting more stable which will increase the overall stability of the game. Currently, most of the player stats either are or will be synchronized on the server, allowing the game at its current state to be played as a multiplayer TPS (like Quake for example).

Figure 5: Screenshot several players paying together.

2.3 Sound

The sound section is about, as its name suggests, the soundtrack of the game, the shooting, moving and ambient sounds, etc.
For the next presentation we expected the sound to be started. Shooting and movement will do some sound. And for the last presentation the sound is expected to be finished.

2.4 3D

This section concerns the implementation of all the 3D assets, with their respective animations.

The map now has one base per team in which the players spawn. It also contains 3 objective points that can be captured by the players by standing on it long enough. Finally, the map has mountains acting as world borders as well as hills to create more variety. To implement that, we used the Terrain tool from unity which allowed us to modify the elevations and made placing all the trees and bushes a much easier task than if we would have had to do it manually. The 3D models of the trees are low-poly and were found on the unity's assets library for free. Finally the two bases have a temporary basic shape made using the 3D-modelling software named Blender.
We still have to adapt the size of the map to the player's speed, to improve the 3D assets of the objectives and the bases. We will also split the map in different areas like a jungle and a flat area for example.

Figure 6: Screenshot of the 3D map.

The player has an asset found on cgtrader.com as a free asset. It now has more than 10 distinct animations such as walking, running, flying, jumping, etc. Most of them were created using Mixamo but some were created by us. Furthermore, most of them have been modified to match our expectations. All of these animations are connected via the Animator tool from unity to create a smooth transition between each of these.
We have to change the player's color according to which team it belongs, fix some transition between animations.

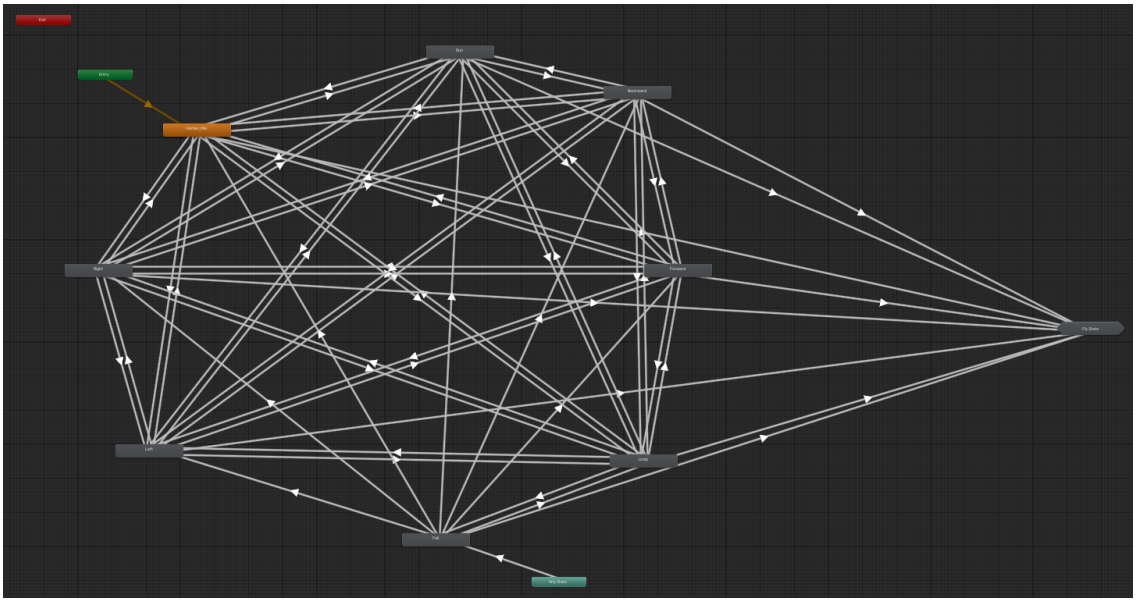


Figure 7: Screenshot of the Animator from Unity.

The weapons have their own 3D model found on the official Unity's assets library for free. their model is synced with the functional weapon the player is holding.
We have to fix the placement of the hands of the player according to the weapon it is holding.

The monster also have its unique asset that may change by the next defense. It has only one animation for the moment.
We have to add color to its model and add all of its animation as well as the transition between them.

2.5 Arena

2.5.1 Objectives

There will be 3 control points which will be activated one at a time and in random order every few minutes and after a global announcement. When they appear, the players can capture the point by standing on it and the team with the most players capture the point after some time.

If captured they allow to access a shop and strategic points plus they deactivate the enemy's shield which allow the players to attack the enemy base. Therefore, they play a central role in each game and are the main objectives of both teams.

A first implementation of objectives is already available; they are represented by white circles on the ground. If a player stands on it, an indicator appears on his screen and he slowly captures the point. The larger the number of players on the point, the faster they can take over it. If the players leave the point while capturing it, their progress is saved and they can resume by stepping back on it. Once the point is captured, it changes color and cannot be captured again.

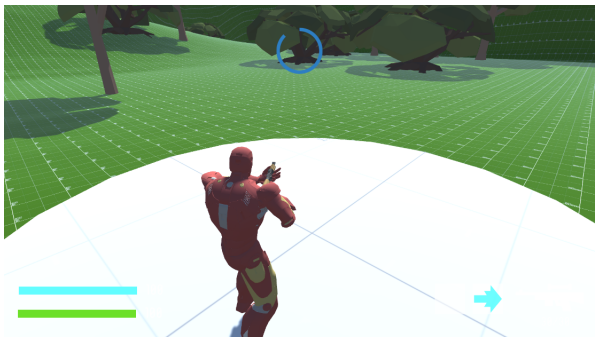


Figure 8: Screenshot of an objective being captured.

2.5.2 Incoming

Future work on objectives will consist of implementing team-based capture, create rewards for taking control and sync the three capture areas with the

rest of the game. We will also have to implement other structures such as working bases and shops.

2.6 TPS

The TPS (or Third Person Shooter) section concerns the very basics of the game, in short the common ground between every modern shooter. As of the first presentation, this whole section is already in a very good shape; each of the following items are either completed and in need of minor changes or well started.

2.6.1 Classes

Advancement by 1st presentation: Base class has been implemented as pure test, but made fairly easy to create new classes. All attributes linked to base class (movement speed, health points ...) can be modified easily without changing the inner working of the base player.

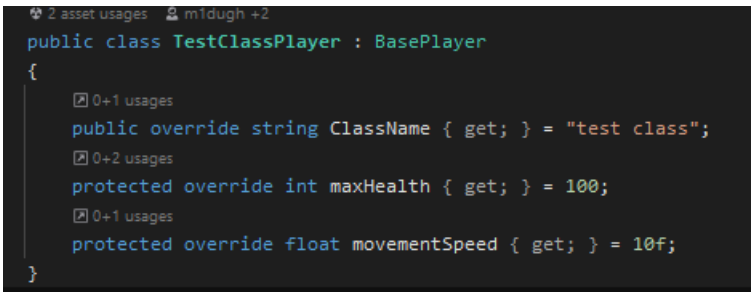


Figure 9: Screenshot of a player class implementation.

2.6.2 Shoot

The shoot section is about everything purely related to shooting enemies. This includes technicalities such as bullet tracing or registering hits. Advanced shooting has already been implemented; the player is already able to shoot players from other teams, monsters and destructible objects. Future additions will be implementing the option to add travel time to bullets and shooting interactions with some structures.

2.6.3 Weapons

The weapons section is about all the different kinds of weapons which will be added to the game. It will consist of creating a large weapon pool with rifles, handguns, shotguns, etc. with for each, different firing speed, damage or precision. A few weapons have already been added.

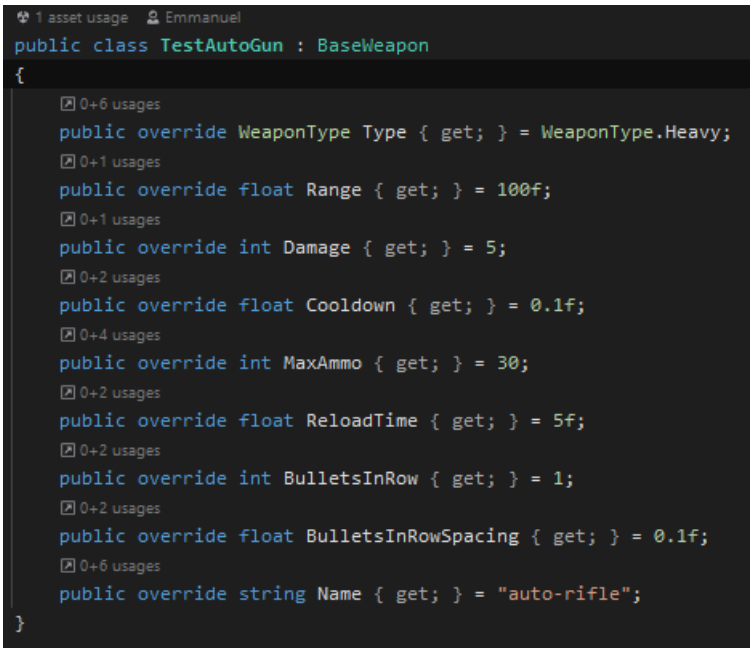


Figure 10: Screenshot of the weapon's implementation.

2.6.4 Items and abilities

The items category is about all the consumables which will be available during the game.
By the first presentation, we did implement all the technical logic for both items and abilities but they are not yet integrated in the game. Our final goal is to have an ability tree for each classes. Items will be sold in shops during the game and can be gained by killing monsters.

2.6.5 Player death

When the player is killed, he will lose all of his bought weapons and start again with his starter weapons after a small timeout. While he respawns, he will be able to buy a weapon to improve from the base available weapons.
As for the first defense, a basic respawn system has been added: when a player's health reaches 0, he disappears from the map and his screen becomes black. After a few seconds, he is summoned back, his weapons are reset and can resume playing.
Future additions will be implementing a shop interface during the player's death alongside linking it with the in-game economy.

Figure 11: Screenshot the current death screen.

2.6.6 Movement

The movement section is about all the player's movements. This includes fundamentals such as walking and sprinting but also other features to make the game more dynamic:

- a dash: on pressing the A key, the player dashes very quickly in the direction he is going
- a jetpack: when double pressing the space key, the player can freely fly around as long as his fuel reserves are not depleted

Consequently a fuel reserve has been added. When the player is on the ground, it slowly fills up and when the player is in the air, it quickly empties itself. The filling up and emptying speed of the fuel reserve are parameters which may be affected by classes or items. The amount of fuel remaining is displayed on the screen in the HUD (cf HUD) alongside the health bar.

2.6.7 HUD

Up to that point a functional HUD (or Heads-Up Display) has been implemented and is composed of fundamental elements such as:

- a crosshair: in the middle of the screen to allow the player to aim
- a fuel bar: in the bottom left-hand corner to show the remaining player's fuel
- a health bar: in the bottom left-hand corner to show the player's health
- a dash indicator: in the bottom right-hand corner to show whether the dash is available
- a weapon indicator: in the bottom right-hand corner to show which weapon is held and the ammunition remaining

All of those elements are displayed on the screen at all time and are constantly updated to show correct the information.

On top of that, some other elements may be displayed while playing the game such as:

- a hitmarker : an indicator on the crosshair displayed when an enemy is hit
- a capture indicator : displayed when on a objective and showing the progress

Figure 12: Screenshot the current HUD.

2.7 Money

In-game currency can be gained by killing players or jungle monsters. When players are killed they also give money depending on the weapons they bought instead of leaving their weapons. All kind of shops in the game will depend on this currency.

2.7.1 Shops

When a team will have captured control points, it will unlock a shop for the players to buy items like shields, health regeneration packs or hand grenades. Player bases will feature shops but with far less items than in mid-map shops. Shops will be represented later on by a building next to the capture point that will not be accessible unless point captured by the team. As the game will go, content in shops will get better.

3 Role distribution

Task	Romain	Lucas	Alban	Emmanuel
Website	X			o
AI		o	X	
Multiplayer	X		o	
Installer			X	o
Sound		o	X	
3D				
Map		X		o
Characters	X		o	
Weapons	o			X
Minions	o		X	
Arena				
Map		o		X
Objectives			X	o
Events		X	o	
TPS				
HUD	o			X
Shoot	X			o
Weapons	o	X		
Items		o		X
Movement	X		o	
Classes		X	o	

For each task we have appointed someone in charge and a substitute.

- X: in charge,
- o: substitute.

We have made sure that everyone in the team has about an equal work load.

4 Schedule

The following table shows our updated schedule; for the most past we managed to reach our expectations but there have been some changes. In some areas we overestimated the amount of work and in others the opposite. For instance, most of the visuals still are in an early stage whereas the HUD, the movements or the weapons are already better implemented than planned.

Task	**First pres**	**Second pres**	**Final pres**
Website		XX	XXX
AI	X	XXX	-
Multiplayer	XXX	XXX	-
Installer		X	XXX
Sound		X	XXX
3D			
Map	X	XX	XXX
Characters	X	XX	XXX
Weapons	XX	XXX	-
Minions		XX	XXX
Arena			
Map		XXX	-
Objectives	X	XXX	-
Events		XX	XXX
TPS			
HUD	XX	XX	XXX
Shoot	XX	XXX	-
Weapons	XXX	-	-
Items	X	XXX	-
Movement	XX	XXX	-
Classes	XX	XXX	-

For each task we have evaluated the necessary advancement at the First presentation, the Second presentation and the Final presentation.

- X: started,
- XX: well started,
- XXX: finished,
- -: previously finished.

5 Tools

5.1 Game

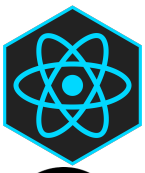

Unity has been the core element of our project since we used this game engine to create the diverse scenes needed. We also used native Unity components such as colliders to ease our task.

Rider has been used as our main IDE over VS Studio since we already use it in prog classes. Moreover its integration of git and unity has greatly simplified our experience.


Blender has been used to edit assets we found alongside creating new 3D assets.


Mixamo has been used to import animations such as running animations or flying animations for our players.

5.2 Website



React

NextJS

Tailwind CSS

6 Conclusion

Making this game allows each one of us to practice programming a lot and to learn lots of new things. Furthermore, since we all are working in group, it is a first sight at what projects could be in a later work. We all learn from this months of project how to work together as a team and how to manage a team for the project manager. Finally, we worked well during this first run and mainly succeeded in our goals but there are still a lot of things to do. (cf Schedule)