Nýr: The Last Stand

Final Report

Maxime SOARES CORREIA
Liam Alexandre ABOUROUSSE
Anita SELVARANGAME
Mathieu DUBRULLE

June, 29th 2021

Contents

1 Introduction

As part of the second semester IT project, EPITA's students had to develop a game from scratch, using knowledge acquired during classes. Our group, FENRIR was born in this context. We decided to present a game named *"Nýr: The Last Stand"*, a viking RPG-game created using Unity engine and developed in Csharp. Before getting into the game details, let us present you the birth of Fenrir and the project *"Nýr: The Last Stand"* from the group presentation to the organization choices.

1.1 Fenrir

I am **Anita**, my knowledge in programming was limited by what we have done in class so far. Thus I was very excited to do this project and learn everything about making a game. I'm not really into playing games but making one was a lot more fun. I really wanted to play a part in the game's aesthetic, I also find it interesting to save things so that we can find it exactly as we left it. That's why the map and the saves part were the most attracting to me.

My name is **Liam**, i was familiar with unity a few years ago when doing games on my own, but i never achieved to get a final version of my game. This year at EPITA i had to memorise all that I've learned previously and going further into programming and using unity's potential. Doing a game in a group was an amazing experience, i really appreciated the way it has gone from the very beginning. I felt the group was solid and advancing in the correct direction. Since I've really enjoyed programming classes this year, i took a part in the "scripting" process. I was responsible for the statistics of the game that make players evolve, the item looting system and the multiplayer part.

Hello! My name is **Maxime**, and my job was to make sure our player behaved well, and that UI looks great. Before starting the project, I already had knowledge in Unity, thanks to their beginner courses available for Students. Since my early years, I always had been a video game enjoyer, and going backstage and to see how a game is made was a great experience. Seeing the very first foundation of our game to become a complex game was amazing.

Hi! I am Mathieu. I'm in charge of the AI. My goal is to create enemies which behave well and can interact with the player as well as being interacted with. I'm also in charge of the website. It has to promote our game and host the installer of the game but also the reports. I think those part will help me for later and that's why i chose them.

1.2 Work breakdown

	Anita	Liam	Mathieu	Maxime
Web site				
Jekyll				
Hosting on GitHub				
Map				
Map design				
Scene management				
NPC's				
Level generation				
Player				
Movement				
Combat				
Levelling				
UI				
Menu				
Dialogue system				
Health bar				
Enemy health bar				
Network				
Multiplayer				
Enemy AI				
Classes				
Movement				
Combat				
Saves				
Save & load				
Interactable Objects				
Items pickup & Inventory				
Chests				
Stats				

Table 1: Work breakdown

2 Book of specification roll-up

2.1 Origins

Although Nýr: The Last Stand is an original game imagined and developed by the FENRIR Group, we, of course, have been inspired by other games and work of arts. In order to understand how Nýr came to our minds, here is a non-exhaustive list of our influences and inspirations.

2.1.1 Influences

The Vikings setting and their mythology is a common theme in nowadays popular culture. We've seen them through many media, from series with Vikings (2013), to video games such as God Of War (2018) by Santa Monica Studios, Northgard (2017) by Shiro Games, and even more recently Assassin's Creed: Valhalla (2020) by Ubisoft Montr´eal. At FENRIR Group, we all love the vikings era and everything that comes with it. Setting up a game with RPG mechanics is the perfect opportunity to explore this deep and vast universe.

2.1.2 Inspirations

When we first had the idea of Nýr: The Last Stand, we first thought about the setting, how the universe will be, and what we could do with it. For the graphics, we wanted to go for a sober and simple environment. Low-poly was the best fit for us. We were mainly inspired by Project Winter's (2019) graphics. For the gameplay, we took different sources of inspirations. The idea of having to explore an area, fighting wave of enemies, in a 4-player cooperation, came from the recent Minecraft Dungeons (2020) from Mojang Studios, which bring back the long-forgotten dungeon crawler genre in video games in a fun and epic experience. Moreover, the idea of completing an island to hundred percent with a crew, finding loots through chests, discovering abandoned structures, vikings camp, and more, came undoubtedly from Assassin's Creed: Valhalla (2020). We chose to implement RPG (roleplay game) mechanics because that is what best fit the game, and that's a genre we all played. Players will have to loot, upgrade their stuff, get better and better in order to scale with the enemies which will become more and more difficult throughout the levels, and even more for the boss fights that players will encounter at the end of each island.

2.2 Game concept

The aim of Nýr: The Last Stand is to produce a multiplayer, fun, and epic, up to 4 players experience. Players will embody vikings from a forgotten age and will have to fight enemies, and creatures on multiple islands.
Players will disembark on a newly discovered island at every beginning of a game. They will have to find every mysteries, treasures, and wealth in order to end the game. Enemies will be there to try to stop players in their adventure.
Players will eventually get to upgrade their stuff to defeat higher-level enemies, for instance to deal more damage to them. Better stuff will be unlocked by increasing level during their play-through.

2.2.1 Scenario

As the Ragnarsson wreak havoc in England, the fangs of the cold still bite unlucky clans of Norway. Vargfell is one of them. Within two winters, supplies will be exhausted and the population condemned to join the foggy Nifelheim realm. In his last stand, the Jarl chose his best drengr to go and explore the islands to the West. Play as one of the Jarl's warriors and discover new lands aboard your ship. With three of your friends, plunder and fight islands to save Vargfell before it's too late.

2.3 Development

2.3.1 Target system

Nýr: The Last Stand is intended to be played on the recent versions of Windows.

2.3.2 Software

Nýr: The Last Stand will be developed using Unity 2019.4.17 engine, and Visual Studio. Graphics assets for the User Interface will be made using Adobe Photoshop CC 2020 and Illustrator CC. The website will be realised with Jekyll (version 4.2.0), a static site generator. It will be hosted on Github.

2.3.3 Costs

Developing Nýr: The Last Stand was not supposed to raise additional costs. Though, the group decided to buy assets to avoid wasting time modelling 3D assets, which require a lot of time and learning. Here is a board summing up the group's expenses.

Assets pack name	Price (in euros)
POLYGON - Adventure Pack	17.87
POLYGON - Vikings Pack	26.79
Total cost	44.66

Table 2: Fenrir expenses

2.4 Work breakdown & Planning

- Work breakdown

  Here is a table resuming how the tasks were supposed to be broken down. There has been some slight changes given who finished their parts faster or what we actually preferred to do after starting to see what every task consisted in. Please refer to the distribution table in the introduction for the more detailed and final work breakdown.

Tasks	Maxime	Liam	Anita	Mathieu
Saves		a	R	
Player	R		a	
Items		R		a
UI	R			a
World / Map	a		R	
AI		a		R
Multiplayer	a	R		
Website			a	R

Table 3: Original work breakdown

  - R: The tasks which the person are responsible for.
  - a: The tasks which the person are going to assist.

- Planning

  Here is our planning with the expected advancement in black and the actual progress represented in colors. For our official starting date, we used the validation date of this document. And for the end, the final date in June. We used the defenses as milestones to have good time markers.

Tasks	1st defense	2nd defense	3rd defense
Saves	20% 15%	50% 40%	100%
Character	75% 75%	100% 100%	100%
Items	30% 30%	75% 80%	100%
UI	30% 50%	80% 80%	100%
World / Map	35% 45%	85% 90%	100%
AI	40% 30%	70% 70%	100%
Multiplayer	40% 45%	80% 80%	100%
Website	30% 60%	80% 80%	100%

Table 4: Advancement planning

3 First Milestone

3.1 Saves

The saves goal is to save the games characteristics when the player leaves it and to load it when he comes back. Therefore the player will find his game exactly as he left it and will not need to go through levels he already did.

3.1.1 Saving the player's position

The script saving and loading the player's name and position uses serializable data, JSONUtility class and a class GameData I created. The class JsonUtility handles the work of serializing and deserializing data into and out of a JSON format. It uses the ToJson() and FromJson() methods for these tasks. However, in order to read data and understand the use of values, FromJson() also needs a type to know what it should be trying to read. This is where GameData comes into use with the method.

```
public void Load()
{
    // Check if the file exists
    if (File.Exists(saveFile))
    {
        // Read the entire file and save its contents.
        string fileContents = File.ReadAllText(saveFile);

        //Deserialize the JSON data
        gameData = JsonUtility.FromJson<GameData>(fileContents);
    }
}

    0 references
    public void Save()
{
    // Serialize the object into JSON and save string.
    string jsonString = JsonUtility.ToJson(gameData);

    // Write JSON to file.
    File.WriteAllText(saveFile, jsonString);
}
```

Figure 1: Sample of the GameManager.cs script

3.1.2 Implementation

The script was done but not implemented at this time because there was nothing else to save so it was not useful to the game behaviour. The implementation was planned for the next defense because then we were supposed to have more things to save like the player's health and equipment.

The saves were not what I expected. I really thought it would be simpler and more similar to the savings we did in some practicals. I ended up spending a lot of time doing researches on how we can save our game with the JSON format.

3.2 Map

The map aesthetic is in low-poly viking style. I used some assets we imported as it was said in the book of specifications. The village itself was made with the "Viking pack" and the "Adventure pack" and for the sky I used another assets pack called "Simple sky". This last pack has been really helpful to simulate a sky and it can be usefull in the future to implement a day/night cycle system. The goal for the first defense was to do the main island and start thinking about the procedural generating for the other islands.

Figure 2: First sketch of *"Vargfell"*

3.2.1 Vargfell

The first island, called ”Vargfell” is the one the player will spawn in at the beginning of the game. It’s his ”homeland” the player will be able to craft equipment and buy some food there. Therefore ”Vargfell” is very detailed and thought to be welcoming.

- The village:
  The village was quite easy to do given that I already had the assets but it was still really time consuming. Placing every object meticulously and making sure that everything was homogeneous was the real challenge.

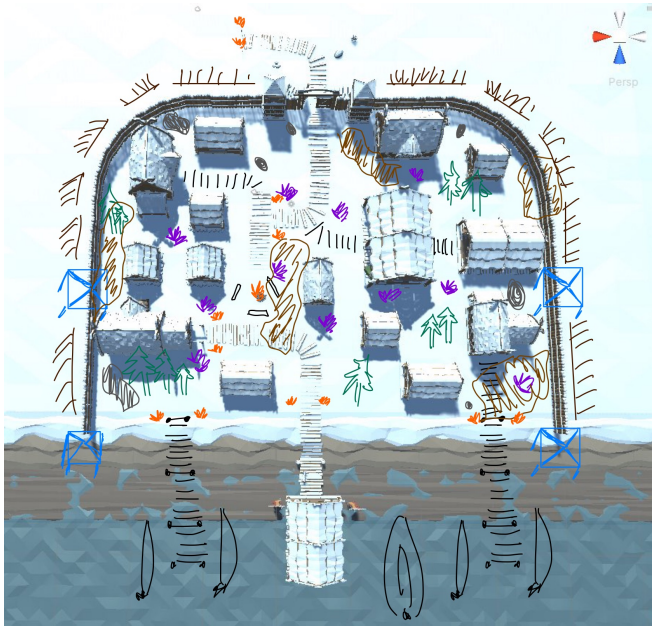

Figure 3: Sketch of the map details using what was already done.

- The environment:
  The island name means ”Wolf fangs” in old Norse so I tried to make the mountains based on this shape although the assets were not auspicious to this shape. The forest surrounding the village was made with the poly brush but it created some issues like levitating trees, the trees were also way too inclined. So I had to shift everything by hand. The sea was simply made with the assets but we’re thinking about upgrading it for the final defense.

- The lightning:
  The lightning was more intricate to set up than I thought. I had to change the lightning settings of the map and put two main lights to make it look realistic. I also added several lights for the torches and the campfire to make the map look more alive.

- The NavMesh:
  I used a NavMesh map to define which component of the map are walkable or not because obviously the player won’t be able to walk through the objects or on things like clouds. I also made the mountains not walkable because we didn’t want the map to be to vast and the goal is not to explore the "home" village but to explore other islands.

The village was not entirely finished I had to do the fire effects and the start the non-playable characters. The fire effects were supposed to be done with a shader but I encountered some

obstacles that you will learn about in the next chapter. The non-playable characters will not really take part in the game, except for some of them, they will by there as a ”decoration” to make the village look more alive and inhabited. I hoped to do the main NPC’s animation for the next defense, namely the boat keeper and the merchants.

3.2.2 Level generation

- Method:
  I also did a lot of research on the random generation of objects. As they are many ways to do it we decided to implement the easiest to avoid being short on the time. The method I will be using consists in spawning objects randomly but at predefined positions. This method is really easy to code but it implies a bit less diversity. So I will to my best to make it look as diversified as possible.

- Steps:
  The procedural generating is something that needs to be broke done in little problems. Therefore, I have already defined the different elements that are going to be on the islands created by this process. The next thing to do will be to define a hierarchy of the different elements that will go on the islands to go from the biggest to the smallest.

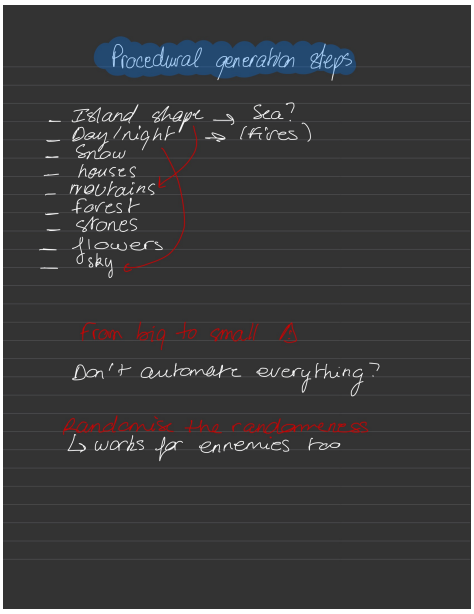

Figure 4: The different elements that has been differentiate for the procedural generating.

As a feedback of our journey’s first part I would say that the only real problem encountered while working on the map was using the polybrush. I needed to inform myself more on the

subject in order to use it correctly for the next time. Overall making the map was a great experience even though it was really time consuming it also was really amusing and I really enjoyed working on it. I was a little bit disappointed that I didn't had the time to do the flame shader before the defense but I was still in agreement with what I planned so I was satisfied with my work until then.

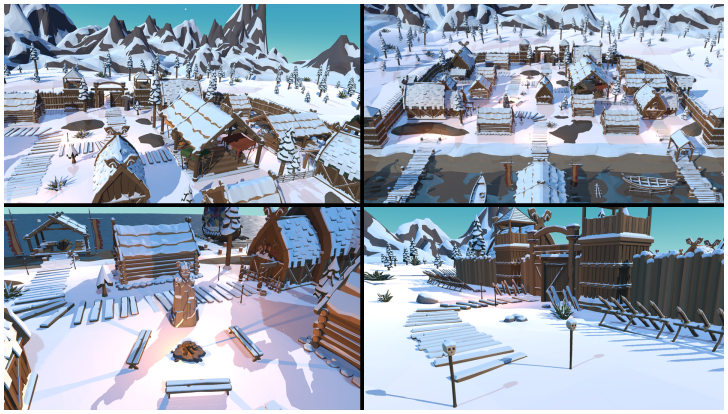

Figure 5: Screenshots of the version of *"Vargfell"* presented in the defense

3.3 Player

3.3.1 Player movement

The Player part main goal was to have a functional player able to walk, to chose the best path and to reach a destination with animations. This player needed to be the frame for most of the other tasks. In fact, a Player was needed for the Multiplayer, the UI, the AI, and of course the World. This part was the first one with the website which we really focused on. It needed to be ready fast, and it was.

Finally, I decided to use an built-in system from Unity: Navigation Mesh. After a bit of research, I finally managed to create a smart movement system. If the user clicks on a A point, the character will, alone, finds the quickest route to get to its destination. Using Navigation Mesh, we can easily define "Walkable" and "Not Walkable" areas.

Following this system, I created a Target lock system. While left mouse button click allows the user to chose a destination for the player to reach, the right mouse button click allows the user to focus a targetable object (i.e. an enemy, a container, or an interactable). This way, the user don't have to click everywhere in order to follow a moving target for instance. Instead,

the user just have to right click on the target, and the character will travel to its destination and follow it until it reaches it.

Finally, I decided to use an built-in system from Unity: Navigation Mesh. After a bit of research, I finally managed to create a smart movement system. If the user clicks on a A point, the character will, alone, finds the quickest route to get to its destination. Using Navigation Mesh, we can easily define "Walkable" and "Not Walkable" areas.

In order to have something else than just a character T-posing, I had to animate it. Again, with a bit of research, I finally managed to get something which looked exactly the way I wanted. I downloaded animations from mixamo.com, and then I used a bit of maths to get a float between 0 and 1 representing the current speed of the player, updating at each frame. Using Unity Animation Blend Tree, I managed to play animations depending on the speed.

3.3.2 Inventory

I also worked on making a working inventory, just to have an idea of what it will looks like in the game. This mechanic is closely linked to the UI part, which I also worked on. Every player has an instance of an inventory, which was primordial since it's a multiplayer game. I had to make sure that players didn't share the same inventory. The inventory is also very modular and we can easily modify the slots in it.

The inventory is displayed in-game (see UI) by pressing the corresponding hotkey, using Unity new Input Manager. This way, it's easy to make possible modifiable inputs. Whenever an Item is picked up (see Items), it will appear in the inventory and disappear from the scene. If the user clicks on the item, it will use it whatever it does (healing, restoring stamina, equipping sword, whatever.).

3.4 UI

Even if UI wasn't an important part for the first defense (about 30%), and since I was ahead of the schedule for the player part, I allocated a lot of my time into it. User Interface is the whole graphical part allowing the user to interact with our game, it includes: Menu, Health bar, Stamina, Inventory, Gamertags, various displays, effects, etc. This part was mainly done using Unity UI and Photoshop CC 2021.

3.4.1 Main menu

The main menu is basically where the player can press "PLAY" to start a game, "OPTIONS" to change the settings of his game, or "QUIT" if he wants to leave us. I've gone through a lot of designs before having a menu that didn't feel wrong.

Figure 6: Screenshot of the very first menu. The "PLAY" button spawned the player on a test scene, and didn't include Multiplayer. The animations on the buttons were very fat, and didn't look that well.

Unfortunately, this first menu couldn't handle multiplayer because of its structure. Moreover, the texts fields which I were using were low-res one and weren't fitting on a bigger screen (i.e 4K resolution). I needed to find something else, which we could use on any resolutions. I start designing a menu on Photoshop and then I transposed it on Unity UI. I also decided to go for more visible animations, which felt, again, more user-friendly.

Figure 7: Screenshot of the second menu in 4K resolution.

Figure 8: Screenshots showing how the animation works. When hovering with your mouse, the button scales up to be more visible, and kind of bounce.

As said previously, I also needed to be able to create a multiplayer game on this menu, and even join one. To do so, I created a new menu which is enabled whenever the user clicks on "PLAY". This menu allows the user to input an Username which needs to be at least 3 character long. When the username is valid, the "CREATE" and "JOIN" appears on the screen, allowing the player to chose what he wants to do from here.

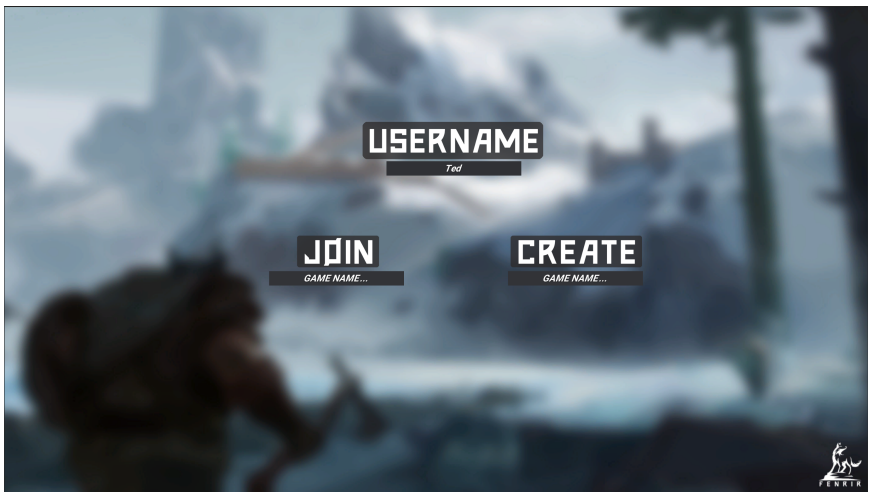

Figure 9: Screenshot of the Connect menu. When the username is valid (Ted in the example), the user can "CREATE" or "JOIN" a Multiplayer game.

For the needs of the project, this menu will be upgraded and a new design implementing our newest functionalities is already being worked out.

3.4.2 Loading screen

To ensure that every players load correctly the same, I had to make an asynchronous loading. Basically, it means that if Player 1 has finished loading, he can already start playing, while Player 2 & 3 are still loading the map. To make this kind of loading was mandatory for the implementation with Photon, else, it will have caused a lot of issues. The "Scene Loader" prefab was made to be easily applicable on any scenes. It was a pain to make at first, but now we can reuse it whenever and wherever we want. The "Scene Loader" allow a lot of customisation, i.e the name of the scene which is loading, tips at the bottom of the screen, a loading bar, audio, etc.

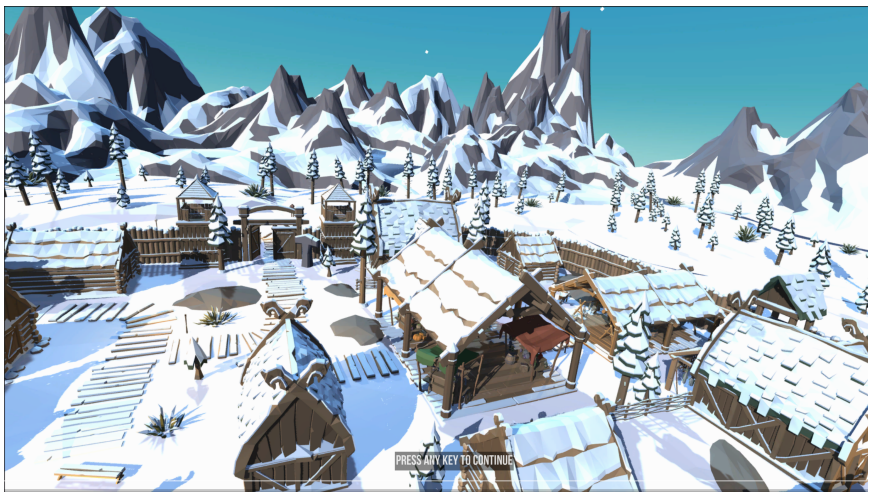

Figure 10: Screenshot of the loading screen. When the loading is done, a "PRESS ANY KEY TO CONTINUE" appears at the bottom of the screen, indicating the players what to do.

To finish with this part, I also implemented a "Spawn Menu" which makes our lives a lot easier when developing. In fact, the "Spawn Menu" appears when the loading is done and a key pressed. A big "SPAWN" button is now present, and whenever a Player clicks on it, it instantiate a PlayerPrefab in the scene, syncing it with other instances of Photon (basically with the other players).

Figure 11: Screenshot of the Spawn Menu.

3.4.3 Inventory

Not much work were done on this part since it weren't necessary for what was planned. The only thing done was for clarity and it's the gamertag. Basically, it's just a Text floating above the player's head displaying his chosen username, with a specific color. Whenever the player rotates the camera, the Gamertag follows it in order to always face the Player camera.

Figure 12: Screenshot of the Gamertag.

The other thing which is pretty important for the behavior of the game is the Inventory. I've done a simple inventory tab on Photoshop which is probably going to get modified for the next defenses. Here is a glimpse:

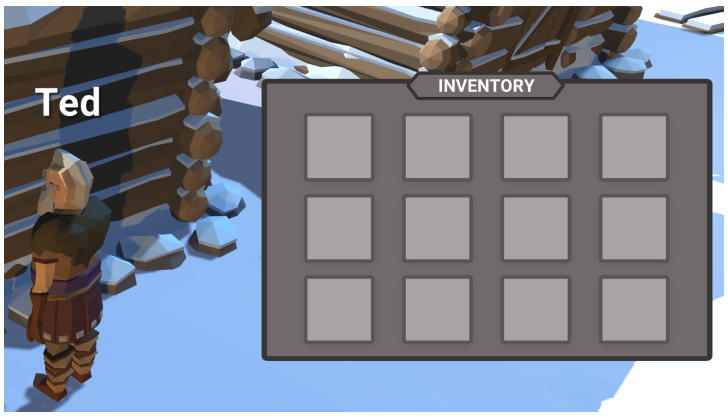

Figure 13: Screenshot of the Inventory.

3.5 Items

• Picking Up Items:

Picking up items was made instantaneous thanks to a C# Script called ItemPickup.cs that says whether or not the item can be picked up the player. If the item contains this script, then the player will be able to store it inside his inventory.

• Equipment:

For the first defence, we focused on the equipment side of items which meant working on the inventory side of things and the statistics. We decided that there will be 4 statistics that can be modified by pieces of equipment. These 4 stats are Armor, Damage, Stamina and Speed. For now, every equipment piece will be able to fill one of these 7 Slots : Sword / Shield / Necklace / Right ring / Left Ring / Armor Rune / Damage Rune.
In order to create faster pieces of equipment, we created a new piece in the asset menu called "Equipment" that load a fresh piece of equipment with all the components we need to modify by hand.

• Statistics:

Statistics and Equipment are two complementary part of the work. When launching a fresh

instance of the game; the player spawns with base stats. It's supposed base stats are 100 Health, 100 Stamina, 1 speed, 1 Damage and 0 Armor. For this first defence we only managed to go as far as creating those stats and modifying the stats for armor and damage. Armor will have a direct impact on health. For example, if the player takes 10 damage but is equipped with a rune of protection that gives him 5 armor, he will only lose 5 health. We subtract the damage received by the amount of armor the layer is equipped with.
Yet, in this part there were two problems. "What if the players has more armor than the damage given ?" and "What if his health is inferior or equal to 0 ?" To resolve those problems we had to create a moment where the players dies (for now it's only signaled in the unity console). And if the damage is inferior to the amount of armor, the player could have received Hp since subtracting a negative number is a positive sum. We then had to create a stopping case to prevent this problem.

3.6 Multiplayer

At first, the multiplayer we wanted to implement into Nyr was supposed to be built around the asset called Mirror. It created a local server on a machine and other machines could connect and join. Yet, there was a major problem: the server was hosted directly by a player and not by a third-party server. In fact, this meant that the player had to open the ports of his IP so other players could connect. They could also use Hamachi to emulate a LAN. We found that really annoying because it was combined with optimisation issues. The result wasn't intuitive and that's why we worked on the multiplayer using the brand new Photon Engine. The main product they propose is PUN. A reliable multiplayer integration specifically designed for Unity. Compared to Mirror, PUN is more straightforward in it's way to create and join multiplayer games because all of them are located on the Photon cloud which guarantees low latency and no punch-trough issues.

3.7 AI

We might be a bit late on the AI compared to what was planned on the book of specifications (40 percent). This is due to the fact that the AI has to be developed at the same rate as the Character and the UI because the AI interact with the players, and we didn't thought about it when we redacted the book of specifications. However we are satisfied with the actual advance of the AI and we are confident for the realisation of the next tasks. There is a lot of resources on the internet and specifically on YouTube for the AI. By following tutorials, i manage to

create a basic AI easily (which can move and has a detection range). However i encountered a major problem : How is it possible to access to the player object if there is none on the scene? (the player are instantiated when they connect to the room). After some researches, i found the GameObject.FindObjectWithTag() which helped me a lot to access to the player object. I chose to give to the AI a logical behavior: it follows the closest player in his range of detection. Then Maxime helped me with the Animation of the model. It glitched at first as i kept applying the root motion to the animation but i fixed this.

3.8 Website

I think that I'm really in advance on this part, the last thing which could take much time is the modification of the header. As I'm developing the website with Jekyll, it will be easy to host the website on GitHub, and the updates of the information about the game on the website are really easy to do. It is said in the book of specifications that the website had to be finished at 30 percent for the first defense, i would say it is finished at around 60 percent. Also a small precision, for the moment the website is hosted locally, we wish to host it on GitHub when the work done will be satisfying enough to have something presentable to show, that is why the website will be hosted on GitHub for the next presentation.

What took me the most time is the setup of my work environment. I had to install and configure WSL Ubuntu, then install and configure Jekyll and finally install VS code and his extension for WSL. Then i started working on the website. The text/images/links integration part was quite easy as I'm using the Markdown language to write text on the pages. However, it took me time to understand fully how the software works in his page management. Also as I don't have a previous experience in web development, i chose to use an asset. The link of the GitHub page of this asset is on the 3rd page of the website. I made a selection of good looking assets and then showed my selection to my team. We agreed on one of them, however i quickly understood that this one was really dense and didn't have documentation, and it was the same for most of the assets i chose in first place until i find the asset "Alembic". It is quite simple but aesthetic and it has a GitHub page with documentation (short but it was enough for me). Then i had to look for the customization of the website (which is not completely done for now). I'm not feeling completely comfortable with CSS right now, but I'm working on it. Also the fact that the assets also include Flex Box CSS doesn't help me to understand the asset's CSS code.

4 Second Milestone

4.1 Saves

So far the saves were going way slower than expected. However it didn't impact the project because it was not necessary to the game yet. We realized that the saves were really dependent of other parts of the project that's why it can hardly go at the same rhythm than the rest of the project.

4.1.1 Implementation

The script done for the last defense didn't work out well when implemented. As for the next three tries, it didn't go any better. I thought it had something to do with the scripts I wrote but actually I found out there was also a problem in my UI. I did a "SAVE" button and a "LOAD" button and since I wasn't really at ease with the UI because it was my first time doing this I realized that I didn't assign the buttons correctly to the corresponding functions. Since we thought it was a problem of compatibility between the multiplayer and the saves we started to think about an other way of doing this and maybe save less things at first to see it was one characteristic in particular that didn't work.

For the last defense I had to find a way to make the saves work but since the problem was in the UI I actually just had to correct this. Thus for this defense I had to save an load in priority the health, the inventory and the level of the player and if possible the player's position.

This task has been the most difficult one for me because I tried so many things that didn't work out as expected so it was hard to stay motivated. I had to frequently alternate with the map part to avoid being constantly disappointed. I'm confident that I will find a solution in the near future and not finding a solution on the first try taught me to not give up and keep digging until I succeed.

4.2 Map

4.2.1 Vargfell

As explained in the previous report I was on time for what I planned for the last defense. I had the main map as planned, namely *"Vargfell"*, minus some details like fire effect and NPC's. As for the random generation I started working on it and thinking about how I'll proceed to

generate random levels. Thus, for this defense I had to finish up the details on *Vargfell* (fire effect, NPC's and some fix in the NavMesh), do the tutorial island as explained in the last report and finally I also planned to start the random generating to at least have each basic component of an island that spawns randomly.

- Fire effect:
  In the first defense's report I had some ambitious project for the fire effect. I wanted to create a flame shader to have a realistic effect but I was actually too ambitious. In the first report I explained that the flame shader might take me a lot of time because I still needed to do some research in order to install the right packages. It turned out that installing the right package involved converting the project into HDRP template instead of 3D which I did. However, it didn't work out because the HDRP template is very costful and my computer couldn't handle it. That's why I couldn't do the shader, my unity editor was crashing every now and then which made my work conditions really uncomfortable. Since I lost a lot of time on trying to convert the project I decided to go on with the fire effect included in the assets package we bought (i.e "Polygone Viking" which actually looked really good with the map since it has a low-poly aesthetic).

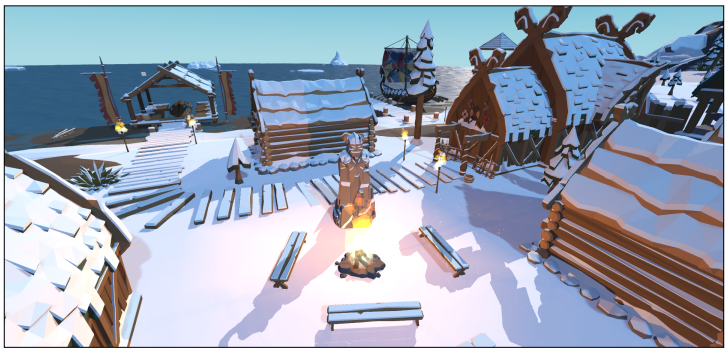

Figure 14: Fire effects example in *Vargfell*

- NavMesh fix:
  For the first defense I explained that I used a NavMesh map to define which component on the map are walkable or not. Soon after the defense we realized that the player couldn't go until the edge of the docks and he could walk on the water. So I had to fix this for the second defense, the player can now look at the horizon peacefully without drowning.

Figure 15: The view from the edge of the docks

- NPC's:
  The non-playable characters (NPC) are necessary to make the game more alive. For the NPC's we planned to do some with whom we can interact with and maybe as a bonus if we have time some wonder AI. Thus for this defense we started to work on the ones who are made to interact. We have three of them: a boat keeper and two merchants. They are respectively named Nora, Bob and Karl although those names are temporary because Bob is not really credible as a Viking. Bob and Karl will be done for the next defense. Nora is the NPC that will allow the players to explore other islands. As the boat keeper, Nora will send them to undiscovered lands or a training island depending on the players' desire. For now we can access the "tutorial island" by right clicking on her, there is no dialogue yet. Bob and Karl will sell food and drink to the players so they can restore their health. The players will also need to craft some equipment and no one do that better than Bob and Karl! Those three NPC's are already on Vargfell with a simple idle animation. We can also interact with Nora to go on the "tutorial Island" by right clicking on her.

For the final defense the goal was to have the multiple choice dialogues and maybe some talking animations for the three NPC's. Nora will have a short dialogue asking the player whether he wants to trains on the tutorial island or go plunder some islands. Bob and Karl are going to have a basic merchants role to sell food and drinks that can restore the player's health and if possible they will allow the players to craft some equipment. The making of the NPCs involves other parts of this project as the items and the UI. Thus we are not certain of how much we can develop the NPCs. For sure the dialogue of Nora will be done and Bob and Karl will be able to sell food as for the details they are our hopes but we can't be sure it will

be perfectly done in time.

4.2.2 Tutorial Island

The tutorial island is the map that has been used to define the steps of the random generation. It is also here to have a second map while the random generation is not implemented in the game. Since it's a first template to help create the random generation we kept it simple. It is not meant to be really detailed like Vargfell. Therefore the island is only composed of a plain village and simplistic environment.

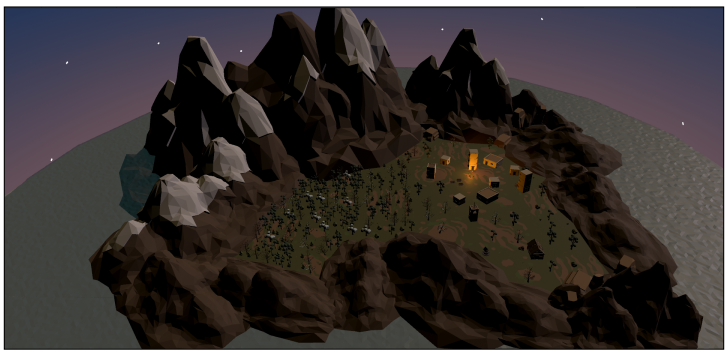

Figure 16: Screenshot of the Tutorial Island

- Village:
  The village is compounded of houses and a campfire. The houses are surrounded of small props like weapons, barrels or carts to animate the map a little bit. Then the same fire effect as in Vargfell was used in order to lit the campfire and a point light has also been added so that the lightning looks realistic. The assets used for this island are from "The Adventure Pack" and "Simple sky".

- Environnement:
  The island is surrounded of mountains and the village is located near a forest. The terrain is flat to make the random generation easier and to use the polybrush. In the random generation the trees will be spawned randomly but here the forest was made with the polybrush. The skydome comes from the "Simple sky pack". The sky and the light imitates the dusk.

- NavMesh:
  As for Vargfell the tutorial Island possess a NavMesh so the the player can only walk on the ground and doesn't walk through the different elements of the map. He will not be able to go beyond the mountains.

4.2.3 Random generation

For the first defense I concentrated on Vargfell but I did started my researches on the random generation. The goal for this defense was to have a proper hierarchy to avoid getting lost and be efficient. We also wanted to have a first whole island spawned by the random generation script so that we would just have to implement it and upgrade it for the last defense.

- Hierarchy:
  In order to do the generation I had to prioritize the steps. The big elements such as the shape of the island had to come first, the forest and the village come next and at the very end we will take care of the details such as little rocks, flower etc...

- Prefabs:
  Doing the prefabs to spawn is the longest part so I decided to only do some "test" prefabs and do the script first. So for now we only have one island shape, three forest and a village. It was essential for this defense to concentrate on making everything work to spawn at least one island. The forests are composed of a hundred trees spawned at a specific position but with a different tree each time. The village uses the same process. Finally the global shape spawns one of the three forest with the village.

- Process:
  The method used here consists in placing empty game objects in a scene to "mark" the position where the prefabs will be spawned. The script chooses a random prefabs from a list that we define and spawn it at the position of the empty game object it is attached to. We can also destroy the island by clicking on "escape" (this key might change or be replaced by a trigger for the implementation). I also added another script to randomize the sky color by changing the texture's offset at each new generation but the script modifying the lightning is more complicated than I thought. Thus the randomization of the daytime may not be implemented in the game since our priority is to have various islands.

By the next defense I was hoping to do many more prefabs to have a great variety of islands but this goal seem a bit too ambitious in light of my delay on the save part and all the other things to do on the map. But I still tried my best to do as many different prefabs as I could. I also wanted to make the sky color and the lights random so that it is a different time of the day at each generation. Then the player should be able to go on those islands so I had to implement the generation in the game through Nora and some buttons to travel easily. And finally the last thing would be to do the NavMesh for the island which I was certain would be the hardest part since the NavMesh has to be bake at each generation on runtime.

As for the last defense doing the map is the part that I enjoy the most doing. Now that it is not only resumed to placing prefabs on a scene it is even more interesting, I'm really happy we chose to do randomly generated islands. The NPC's also add a challenge since it is a bit related to other parts like UI for the dialogue. I'm quite satisfied with what I've done so far, I learned lots of things and I can't wait to go further on this journey. If I finished everything that I've planned for the next defense I might keep adding bonus features and more prefabs to randomize the generation to the maximum.

4.3 Player

4.3.1 Player movement

This mechanic received a few tweaks, mostly fixes and polishing. Before those fixes, the player would eventually bump into enemies, and would never stop in front of an enemy. Instead, the player run indefinitely, desperately trying to literally enter the enemy, which was not possible because of the box colliders. What I did to correct that was simply to change some settings in our navigation mesh agent, for the player but also for the enemy. On the current version, players and enemies stopped when they're close enough from each other without looking wrong.

4.3.2 Player combat

Player's combat was probably the most important feature to be added onto the player for this defense. I started working on it really early, and managed to get something right quickly. Basically, when a player right-clicks on an enemy, which is tagged as an interactable object, the Player runs to it and when close enough, it starts dealing damage to it. Damage are based on the Damage modifiers, which are explained in the Stats part. But when close enough, the enemy also deals damage to our player, and again, taken damage are calculated based on the Defense and Health modifiers. And when one of them health reaches zero, it triggers a death animation and makes it impossible for the player to move. He can also clicks on a RESPAWN button which allows him to be respawned and go fight again. What I also added was the combat's animation to avoid having a game looking like nothing. On the current version of our game, the player swings his arm two times, striking without pity the enemy. And just for the game to look better, I also added an axe in the player's hand.

Figure 17: Ted, wielding his axe

4.4 UI

4.4.1 New menu

The new menu was the main feature in the UI part for this defense. The third menu looks a lot cleaner, and professional. When you first launch the game, you have a quick animation displaying the logo of our game done only in Unity.

Figure 18: The logo animation on the main menu.

For this menu, I wanted to go for a less cartoon style, and I was aiming for a more serious style. The logo was also changed to look better, and again, more professional.

Figure 19: The main menu.

What I also added for this defense was the Settings menu. That's a feature common to every game in the market, and we had to have one. What you can do in this menu is basically change the game's resolution to better fit your game, toggle fullscreen or not, change graphics' quality (you can chose from Low, Medium, High and Ultra), and finally change game's volume.

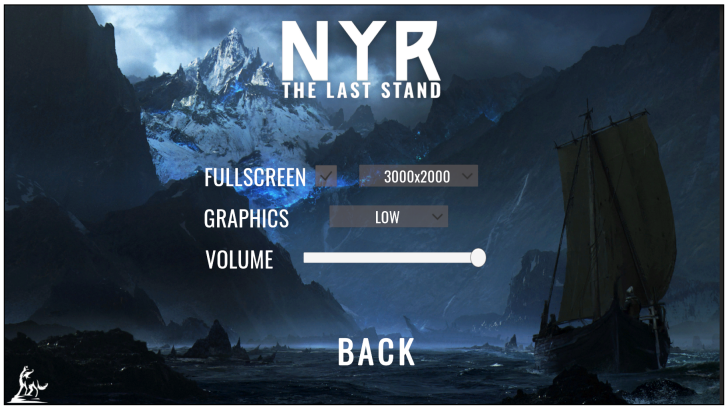

Figure 20: The Settings menu where you can change the settings explained above.

Finally, I reworked the Play menu to fit the new look of our menu.

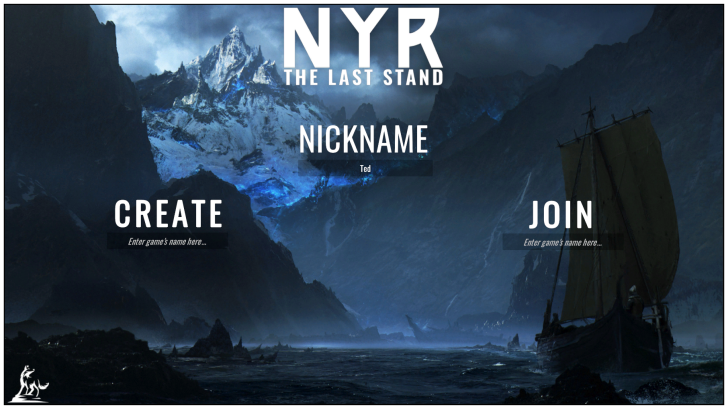

Figure 21: The Play menu.

4.4.2 Healthbar

Now that the game includes Statistics, my job was to be able to visually represent those values. I started by taking inspiration from one of my favorite games for the healthbar.

Figure 22: A screenshot from the The Elder Scrolls V: Skyrim's GUI from Bethesda Studios.

With that idea in mind, I came really fast with a design that exactly fits the game's spirit. A simple healthbar, displayed at the bottom of the screen, which is updated everytime a player takes damage.

Figure 23: Ted, and the healthbar at the bottom.

4.4.3 Loading screen

Because we added a new scene to our game, I also had to manage how to transition between Vargfell, our main village, and this whole new map. Just like the very first defense, I came up with a simple but efficient loading screen, telling the player to "PRESS ANY KEY" if loading is done.

Figure 24: The loading screen between Vargfell and the Tutorial Island.

4.5 Items

• Statistics:

The most important part of the work made on statistics is the levelling system. The player can reach a maximum level of 10. To level up, he has to gather up experience points. So, the work done on levelling an be devised into two main points: The level system and what happens when the payer reaches the next level.
To create a level system that could work with a UI we had to design a code that calculates, every time the player gathers experience, how much experience is needed to skip onto the next level. In fact, reaching level 2 from level 1 doesn't take the same amount of experience as reaching level 6 from level. So, the algorithm also takes this into account, that's why it works indexed on the current level of the player. Once the player has gathered enough experience, he can reach the next level.
But, there's one main issue we faced: What would happen if a player only needs 10 experience to reach the next level and gathers an experience orb of 100? To fix this issue, we had to adapt the algorithm so that at the next level, the payer keeps the extra 90 experience that we wouldn't want to disappear into the abyss but normalize it.

Next up is the part where the player gets to the next level. We promised in the last defence to make stats evolve according to level and we delivered. Whenever the player gets to the next level, it's amounts of health grow of a certain amount depending on the level he reached. The player will go from 100 to 120 health when he gets to level 2. But he will go from 300 to 400 when he reaches level 40.

- Chests:

Chests are an essential part of RPG games. They allow the player to get stuffs that will improve its statistics. From the basis we wanted to implement a chest system into Nyr. It should have been a system where players can store and select items into the chest but we decided that It the chest should only be open once, and the stuff will drop at random position around the player that interacted with the chest.

We decided to get into that direction to make sure there we no conflict with multiplayer and networking. It could have cause trouble whenever two player got access to inventory at the same time.

I really enjoy the challenge of working on scripts and see the results into a video game. It really feels pleasant when it works. But sometimes theorizing a system can take a lot of time until you find the correct formula. Working on chests was very challenging because I had to think of what I would like it to do at any time. Yet, I finally reached my objectives.

4.6 Multiplayer

We said that we would implement a lobby system where player can access a room trough a server browser. We failed to do that due to the limitations fixed by Photon.

Even tough we faced this problem; we have improved the multiplayer experience in Nyr. Every three out of four times, the camera of one of the players was fixated on another's character from another player. This was due to some errors we had made in the player Prefab that spawns whenever a player connects. It happened when two players would join in a short span of time.

The bug Is now corrected and networking is flowing and pleasant for the player, thus making the experience even better.

Players can create a room that they give a unique name by clicking on the button "Create" on the menu and the next scene is asynchronously loaded. Then, other players can join by entering the name of the room previously created under the button "Join" on the menu. For every player spawning, he has his own player prefab with all the scripts needed and it's own camera that the players can individually control.

4.7 AI

We have done the hardest part on the AI, as we now have a solid bases with plenty of scripts. The only thing that remains to be done now is to create other classes of enemies by duplicating the object, adding an other model, and finally modifying the scripts so it has a different behavior in the way it fights or detect the players. We didn't fix an objective yet for the bosses (how they look, how much damages they give, how they attack and so on...) but we will figure this out. I am feeling much more at ease using unity now. I understand how the scripting part works. I started working on this part late compared to the planning as we faced difficulties in the organisation. However i implemented everything i said i would, except for the multiple classes of enemies but it isn't a problem as it is easy to do from now. Finally i want to explain briefly how the combat system works : there are two ways of implementing combat for the AI : the first one is to have the same scripts for the player and the AI. However this method isn't interesting when it comes to implement levels for the enemy side, indeed as i didn't imagine the level and statistic system for the AI like Maxime and Liam designed it for the player, I duplicated the scripts for the player and created my own methods and variables management for the enemy. This way is longer to implement but it will make the task easier for the following work on the AI. Finally, there is not much to say on the animation part, it is just a system

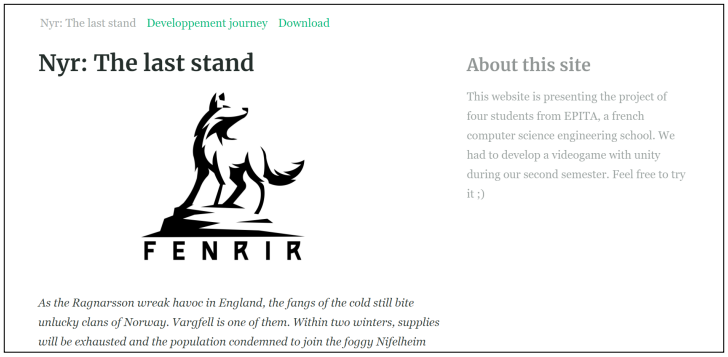

Figure 25: The website

with Boolean variables used to know when the animations should play. As a choice I made the enemy disappear once the death animation played.

4.8 Website

You can now visit our website at the following address: https://mathieu-cs.github.io/nyr_the_last_stand/. The hosting took more time than i thought, as the asset i chose isn't supported by GitHub. Therefore i had to skirt this problem by defining the theme i chose as a "remote theme" in Jekyll. This little tweak allows one to host a website with any theme. However, by installing the plugins required for this, i lost the graphic work i did on the website as the files were updated when i downloaded the plugins. I'm still satisfied with the way it looks and this made me realise that i am not a great graphics designer because the base theme looks better than what i previously did. I just adjusted the size of the navigation bar, because it looked to small to me. Also i updated the information on the website with the progress of the team. This hosting part allowed me to understand few things about the way Git works. I had to read the documentation of GitHub pages to solve my hosting problems and the i am feeling better using git now, as i understand what a repository really is, the way branches and commands work and so on... There is not much to say more, this was a pretty short part but i am happy that it works fine.

4.9 Expected progress for the last defense

4.9.1 Expectations by parts

- ***Saves:***
  The saves will have to be implemented by the last defense. We should keep track of the player's progression so that his stats are the same when he quits and comes back to the

game.

- ***Map:***
  The NPC Nora presented in the report will be upgraded by adding a dialogue system and two other NPC's will be added. As for the level generation it will be implemented in the game and randomize to the maximum.

- ***Player:***
  What needs to be done is a bit of polishing for the combat animations to make them more realistic. What is also planned is the behavior for when the player decides to leave the game.

- ***UI:***
  A multiplayer chat which will let players talk to each other, but also tells them about important events in the game. A pause menu is also in the works, and more visual effects for combat and levelling too.

- ***Multiplayer:***
  If we manage to rework our networking system, we would like to implement that lobby browser we failed to make work. The rest is just making sure everything is correctly instantiated.

- ***Items:***
  For the last defense, i plan on making unique classes of characters that specialize themselves on certain attributes.I will also make sure XP brings more depth to the game. It's objective isn't to make easier but more challenging.

- ***AI:***
  For the last defense, we will have three or four classes of enemies with different behaviors. We will also have a boss (a bigger enemy harder to kill).

- ***Website:***
  The information on the website will be completed and we will have a download link for the game, the light version of the game and the final report.

4.9.2 Planning for the last defense

Tasks	Passed defense	Current defense	Last defense
Saves	20%	50%	100%
Character	75%	100%	100%
Items	30%	75%	100%
UI	30%	80%	100%
World / Map	35%	85%	100%
AI	40%	70%	100%
Multiplayer	50%	80%	100%
Website	30%	80%	100%

5 Third Milestone

5.1 Saves - Anita

5.1.1 Characteristics to save

- Stats:
  We wanted to save the player's stats namely the health, the xp and the level of the player. Unfortunately I only managed to save the health of the player.

- Inventory:
  The inventory can also be saved when the player saves his game with some equipment on him, when he load the game again the same equipment will appear in the inventory.

- Position:
  The player can also save his position on the map so that he can re spawn at the same last point where he saved his game.


```
{"actors":
    [{"pos":{"x":3.705707550048828,"y":0.01256561279296875,"z":48.2286376953125},
        "inventory":[],"health":100
    }]
}
```


Figure 26: Sneak peek of the saved file

5.1.2 Implementation

- Canvas:
  So that the player can save his game I chose to do a canvas that appears when he hit the key "S". This key opens a canvas with two buttons on it.

- Buttons:
  The two buttons are "SAVE" associated to the save function that will save the player's

characteristic in a .json file. The "LOAD" button correspond to the load function that will take the information of the .json file previously saved and apply it to the player in the actual game.

Figure 27: The saves button

5.2 Map - Anita

5.2.1 NPCs

- Dialogue:
  The dialogue system is fully done and functional. For Nora there is a small dialogue and when she is finished talking the player can choose to go on the training island or a randomly generated island by clicking on a button.

Figure 28: Nora's dialogue

Figure 29: Travel choices

Bob and Karl also have a small dialogue each. Bob allows the player to craft equipment and Karl sells food and drink. The player can choose what he wants on a dropdown menu that will pop at the end of the conversation.

Figure 30: Karl's dialogue

Figure 31: Food & drink choices

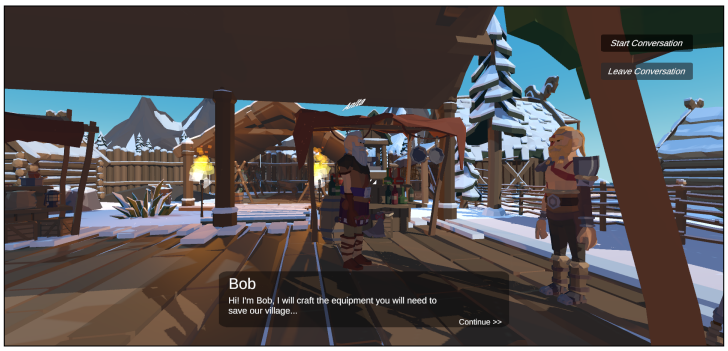

Figure 32: Bob's dialogue

Figure 33: Equipment choices

- Animations:
  I put some small animations for the dialogue, the dialogue box floats in and out as the conversation starts or finishes. There is also an animation for when the dialogue is displayed, the letters appears one by one so that the text is slowly displayed.

5.2.2 Random Generation

- Prefabs:
  I tried to do as much prefabs as I could, unfortunately I didn't manage to do much of them because creating different island shapes and layouts is really long thus I didn't had enough time to do what I wanted. However I did a correct amount of them, the islands seem a bit different each time and the sky generation adds a bit of illusion too.

- NavMesh:
  The NavMesh for the random generated islands was a hundred times more difficult than the previous ones. The NavMesh had to be generated during the run time since it change each time an island is generated. Thus I did a first try by using the namespace UnityEditor.AI but this namespace doesn't work on the build of the game. So I tried to do it with the NavMesh components package provided by unity but this method didn't worked out well with our game. So now we have the navmesh on the editor because with the first method it works perfectly when tested on the editor. But there is no NavMesh on the

build of the game.

```
using UnityEditor.AI;
using UnityEngine;

0 references
public class NavMeshGeneration : MonoBehaviour
{

    // Update is called once per frame
    0 references
    void Update()
    {
        if(Input.GetKey(KeyCode.B))
        {
            NavMeshBuilder.BuildNavMesh();
        }
    }
}
```

Figure 34: Code sample of the NavMesh generation

- Sky & lightning :
  I chose to also do a random generation for the sky and the light so that there is even more diversity between each generation. Thus there are ten types of skies that have the color and the lightning of different time of the day.

- Implementation:
  When the player start a conversation with Nora at the end he can choose to go on a generated island. From here the island is generated thanks to the generation script and the different prefabs used to create islands then the player is spawn the same way as for the training island and Vargfell. In addition to that the players can also change from island to island by the button on the top right corner when they are not on Vargfell.

Figure 35: An example of a generated island

Figure 36: An other example of a generated island

5.3 Player - Maxime

Since everything was done for the previous defense, the only I had to do was to optimize as best as I can the code, to find more efficient ways to achieve our goal. In the final version, the player is able to move freely around the map, right-click on interactables (including chests, NPCs, and enemies), and fight for his life.

5.4 UI - Maxime

5.4.1 Experience bar

Following what Liam has done, and since UI is my job, I implemented the visuals for the experience bar, showing the player how far he is in the current level. The experience bar is displayed at the bottom of the health bar, in blue, along a little text displaying level in that format: "Level X".

5.5 Items - Liam

• Statistics:

Levelling is now done, whenever you kill an enemy, you gain experience that make your stat go up.Thus making him stronger. Afterwas, we had to shop the experience system and it's progression. We made the experience bar appear under the health bar.

5.6 Multiplayer - Liam

Nothing needed to be done for the last defense in multiplayer.

5.7 AI - Mathieu

5.7.1 HealthBar

To know how much life the enemies have left we needed a health bar for the enemies. Maxime already made a health bar for the player, but this one doesn't fit for the enemies, as its design is too complex. I chose to use a simple design for a better readability in game. The health bar is then a red image scaled to a rectangle. It is controlled with a slider going from 100 to 0. This slider is controlled by a script which sets its value according to the current health of the AI with the following formula : currenthealth x 100 / maxhealth. This formula allows us to use the script for any enemy. I also used a script to make the Health Bar face the player's camera at any time.

5.7.2 Enemies classes

To diversify the gameplay, we needed multiple enemies. This implies different statistics, models, animations, and behavior. Therefore we decided to create 3 type of enemies : the "Warrior", the "Glass canon", the "Tank" and a "Boss"

The Warrior is the simplest one. Its the final version of the enemy we presented the last defense. It as normal statistics and rewards few xp points.

Figure 37: The basic enemy

The Glass Canon is fast, deals a lot of damage, as a huge detection range, but is weak (it has only 25 health point). It rewards a decent amount of xp points.

Figure 38: The glass canon

The Tank has a lot of health and armor, deals few damages and has a small detection range. His attack phase is based on a cycle of 8 seconds : he attacks for 4 seconds then blocks attacks for 4 seconds. It rewards a lot of xp points.

Figure 39: The tank

The Boss is a big enemy, 2 times bigger than the Warrior, it is very slow in everything, but it hits strong. It has a lot of health and rewards tons of xp points.

Figure 40: The boss

5.7.3 The Animation

We needed different animations for the enemies so we can differentiate them. Therefore i had to create multiple animator controller which all have the same structure (except for the tank) : They're composed of a Blend Tree, which help us have smooth transition between being static and walking/walking and running, and then an Attack animation and a Death animation, all controlled by Boolean variables. The animation for the Tank is different as it uses an other animation : the block animation. The transition between attack and block is controlled by a Boolean variable which changes every 4 seconds in the script which controls the attack. All the animations have been downloaded from mixamo.com.

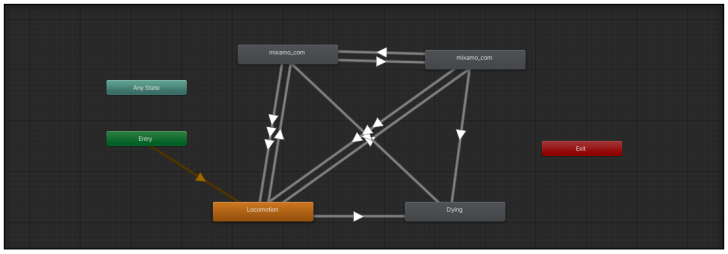

Figure 41: The animator controller for the tank

5.7.4 The scripts

I had to change the scripting system if i wanted to have multiple enemies. Indeed, as all the enemies don't have the same behavior, i had to use sub-classes for my combat script. I first thought about doing different classes, but realised it wasn't possible because it would make the interaction with the player hard to code. I thought about using the same process for the statistics scripts, but realised it wasn't useful. I just reused the same script on all the prefabs and modified the stats manually for each enemies.

5.8 Website - Mathieu

5.8.1 The first page

The fist page has been updated with all the information a new player should know about the game. It makes the promotion of our project and explains the history of the game, the gameplay, etc...

5.8.2 The second page

The second page has been updated as well, you can now find a complete overview of our personal experience during this project. The last report can be found as well at the end of the page.

5.8.3 The last page

You can know download the installer on the last page to play the game. The Credits and Resources used to develop the game have been updated as well.

5.9 Prides and disappointments

5.9.1 Mathieu

I'm pretty proud of the "Tank" enemy. It wasn't easy to code and animate and it works fine. I'm disappointed by my Boss though. I didn't feel like modeling and animating something special (like a big wolf as an example) to make a Boss which would have been cool. Instead it is a bit simple for a Boss in my opinion.
I'm not really disappointed for the website as it looks good in the end. However if i had to do it again i would do it from scratch without using software like Hugo or Jekyll. The experience would have been more rewarding and i would have developed more skills in HTML and CSS. I know I'll have to use these languages later at EPITA so it isn't a big deal.

5.9.2 Maxime

I am proud of what I have done so far for the project. I had the occasion to learn a lot of new things while being able to produce something real, and playable. C#, Unity, and Git management are of course great addition to my knowledge.
At the very beginning of the project, I had a lot of ideas and of course some became true. Although, my biggest disappointment was to be unable to fulfill all of my goals. And even without that, we have a great project, and besides the "coding experience", this was also a great social experience. Having to work like that, on such a project, with other people was somehow an introduction to how will be our future work environment.

5.9.3 Liam

I am very proud of how my scripts are looking. They are well optimized to. They work fine together and there are no conflicts. The part that I took the most pleasure to realise is the levelling system. I had to to go beyond my limits in Object Oriented Programming. Syncing the level bar and the stats was lots of fun even tough it took a lot of time.
The only downside is the multiplayer part. It made me fail all the object pickups scripts. I couldn't find a way to sync the inventories and the objects you pickup which made me very frustrated.

5.9.4 Anita

I am proud of my work on the map even if I didn't do as much as I would have wanted. I did everything that needed to be done I don't have anything late for the map in respect of the book of specification. Moreover I really like what I did with the different maps and my team mates

agree to say that my maps look really good. The non playable character is also something I'm really proud of since it depended on other parts such as the UI and I managed to do it well even if I had to learn everything about the UI. The part I'm really disappointed in is the saves as I said in the report, the saves were not what I expected and I had a really hard time doing it. I'm also a bit sad I didn't get to put more diversity in the generated islands

6 Recap of our journey

6.1 General feedback

This project was mostly positive and taught us a lot about working in group. We had some minor altercations but we always found a way past them. We learnt to communicate better and listen to each other. The decisions on the project were only made with everyone's agreement and no one was left aside. This is the part our group was really good at, we always listened at each other's opinion and most of the time we agreed which made the decision-making easier and faster. We also started to have more and more mutual aid as the project went on. We understood that to make it work and do our best we needed to help each other. This was something essential to the good running of our group work.

6.2 Personal experience

6.2.1 Liam

The project finally came to an end. The knowledge i acquired during these 6 month is monstrous.I learned to use Git, which makes my life simpler now. Sharing files and organising them with your team is very valuable. Talking of the team, they were all very nice to work with. I felt like the pace of the group was fluid and development wasn't going to fast or to slow. I personally learned a lot on teamwork and communication. It felt very professional but also really pleasant.

As far as my personal knowledge in programming is concerned, i had to learn tons of things. Comparing to what i was capable to produce in a certain amount in time before the first defense and know is clearly impressive.

6.2.2 Mathieu

I'm happy that this project has came to it's end. I learned a lot in scripting but also in human relations management. It made me realize that i find this type of work not interesting as i spend most of my time searching for information and trying to understand how the software works, instead of thinking about how i can make my code work and optimize it. Unity is made for artists rather than developers and the experience i had scripting the enemies was boring and long as i had to understand Unity and how the scripts work.

I'm really happy that this project forced me to learn Git. I have a better understanding of what it is and how it works now. It is a very powerful tool for team work if it is well used.

There are good and bad practices of push and merge and by making mistakes, I know feel good using this tool.

I am so exited for the S3 project. I hope this project will demand more coding skills and will be closer to what we do in practical work. I want to try to be a group leader for this project. It should be my last project before the engineer cycle so i hope it will go well.

6.2.3 Maxime

Seeing this project coming to an end feels weird, I'm not going to lie. It became a kind of habit of having to work on that game, that was only an idea at the beginning. Seeing our work finally looking like something is amazing.

Nyr is a video game, for sure. But Nyr was also an experience. An experience that allowed me to discover how to work with a group, the pros and cons of such a project. At some point, we were up and down but finally we managed to do this together, as a team, and we made it.

What I will retain from such an experience is how to manage my time when working in a group, but also how to maintain social links, professionally talking. I will also retain that when it comes to such a project, I have one thing to remember: do not be too ambitious. I had a lot of ideas for Nyr, and a lot got skipped because of time or because of our lack of experience. It made us lose time, and I will not do that again.

6.2.4 Anita

I'm really sad this project has came to it's end. I really enjoyed working with the members of Fenrir and this game was really interesting to develop. I would have liked to continue working on Nyr because we had so many ideas to upgrade the game afterwards, but every good thing has an end right?

Working on Nyr with the group was an unforgettable experience. It taught us all a lot on working in group and we learnt so much about unity and everything we used to create our game. Now personally I have the feeling that I also became more autonomous and this project also helped me develop my self confidence.

Being a girl who liked science I always had a hard time mingling with boys in school. Mostly because they used to underestimate me when it came to group work. I was always left aside. Gladly the members of Fenrir considered me as much as anyone, they listened to me and treated

me equally. That was my biggest fear with this project and more generally with epita but it appears that I found great friends here more easily than in high school.

In brief this was a great experience that brought me knowledge on code and team work, self confidence, autonomy and friends. I'm happy we did this game together although I would have liked to develop the game even more. Since I can't get enough of making games I actually found a internship in this area so I'm really looking forward to use and deepen the knowledge I acquired during this 6 months.

7 Conclusion

Members of Fenrir thank you for reading this report and hope it made you want to buy to Nýr: The Last Stand.

List of Figures

List of Tables