End The Bully

S2 Project Report n°1

Par
hugo.camarasabirabent-martinez , louise.bendjeguelal
noe.auguste , wachirawit.fasquel , alex.chiriax

EPITA
SUP anglophone section

7 january 2025

Summary

Introduction

Our company, inspired by RogueLite Hack n Slash games such as Cult of the Lamb, is currently developing an exciting new game titled End The Bully. Set in a dark and captivating universe, the game follows the story of a protagonist who, after being bullied by the son of the world's main antagonist, takes a stand to seek revenge against his tormentor. The game features classic mechanics of the Hack n Slash and RogueLite genres, challenging players to defeat waves of enemies using unique skills while acquiring equipment through chests or rewards from vanquished foes.

End The Bully also includes a cooperative multiplayer mode, allowing multiple players to team up online via a peer-to-peer (P2P) network system. Enemies are driven by AI, ensuring dynamic and engaging encounters throughout the game.

In this report, we will introduce the advancement of our company and its team on the video game.

1.1 In depth presentation of the game

During the brainstorming phase of our project, we decided to prioritize replayability over creating a long, linear adventure. This approach led us to consider several genres, including sandbox games, PvP shooters, RPGs, and RogueLikes. Ultimately, we chose the RogueLite genre because it provides a meaningful challenge and clear objectives for players, enhancing their engagement with the game.

Our primary inspirations are RogueLite Hack n Slash games like Cult of the Lamb, Risk of Rain 2, Rotwood, and The Binding of Isaac. These games stand out due to their sense of progression, where players feel a continuous improvement throughout their journey, while also offering varied experiences on subsequent playthroughs. Additionally, these titles were developed by independent creators and achieved significant success, which is a motivating factor for our team.

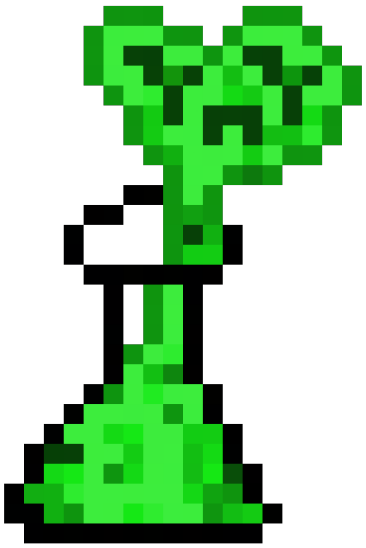

Figure 1.1: Sprite of a mob, made by the graphic design team

Our inspirations also showcase diverse art styles, including pixel art, cartoon visuals, and realism. For our project, pixel art emerged as the ideal choice due to its accessibility and alignment with our resources and design goals.

To enhance replayability and player engagement, we will implement a loot system inspired by Muck and Risk of Rain 2. This will allow players to progressively evolve their characters throughout a playthrough, ultimately becoming powerful enough to defeat the final boss.

Players will have to navigate through increasingly challenging floors filled with stronger enemies. These foes not only test the player's skills but also offer more powerful upgrades as rewards. However, players will have the option to bypass certain obstacles and move directly to the next floor. While this will speeds up progression, it will come with the risk of facing under-prepared encounters with mandatory boss fights, which will block access to subsequent levels.

Bosses will serve as gatekeepers, presenting mandatory challenges for players to advance. These encounters are designed to test the player's mastery of the game's mechanics and the synergies created by their chosen loot and upgrades.

Our goal for End the Bully is to create a game that balances difficulty with rewarding progression. Each playthrough offers a unique experience, encouraging players to experiment with different builds, strategies, and approaches. By combining engaging mechanics, strategic depth, and a dynamic loot system, we aim to deliver a game that feels fresh and exciting with every run. This design philosophy ensures End the Bully will offer both a challenging and satisfying experience, resonating with fans of the RogueLite genre while carving out its own unique identity.

1.2 Task repartition

Task	Employee				
	Alex	Noé	Hugo	Louise	Wachirawit
Art Direction / Graphic Design		R		S	S
Multiplayer / Networking	S		R		
Enemies / AI	S		S	R	
UI		S		S	R
Loot / Upgrades		S	R		S
Combat / Player Experience	R	S			S
Marketing / Communication	S		S	R	

R : responsable, S : substitute

1. **Art Direction / Graphic Design**
   The task of Art Direction (AD) and Graphic Design mainly involves drawing and designing the sprites of various characters and objects in the game, as well as the maps and levels. It also includes sprite animation and will work closely with the Marketing/-Communication team.

2. **Multiplayer / Networking**
   The members of the group handling the Multiplayer/Networking task will be responsible for making multiplayer functional and operational. They must ensure secure network connections and optimize the data transmitted through it to make the gameplay as smooth as possible.

3. **Enemies / AI**
   The Enemies/AI segment of the game involves the AI of all enemies or companions in general, the AI of various bosses, as well as the addition of new enemies with their attacks and specific features.

4. **UI**
   The UI (User Interface) task includes creating and refining the HUD, the game menu, settings (including resolution, key mapping, and game language), the inventory, and potential post-processing to improve the graphical quality of the game.

5. **Loot / Upgrades**
   Those responsible for the Loot/Upgrades section will handle the loot system (i.e., how enemies or chests drop items), the upgrades provided by items, and the addition of new weapons.

6. **Combat / Player Experience**
   The Combat/Player Experience section covers the movement of various characters, the combat system (specifically how attacks are performed), the enhancement of combat with ranged attacks, the major addition of Spells, and finally, the procedural generation of enemies and chests.

7. **Marketing / Communication**
   The Marketing and Communication team will primarily focus on the game's website to present and explain the game in more detail. Secondly, they will handle potential advertising for the game through social media or other platforms.

1.3 Task planning

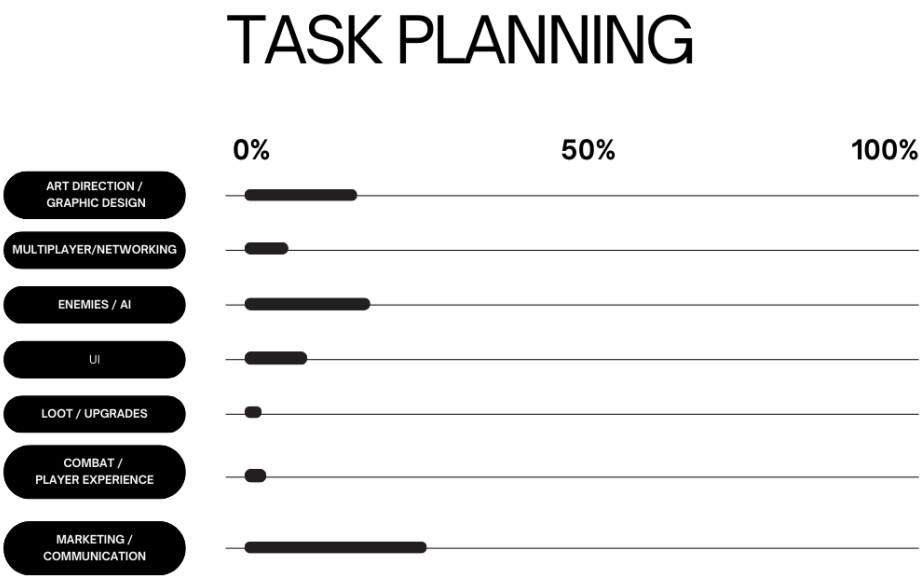

Figure 1.2: Graph of the advancement and the planning of the tasks

Tasks Advancements

2.1 Tilemap and Heraut sprite

In our recent project, we integrated various game development elements to create a cohesive platformer experience. Here's a detailed breakdown of our process:

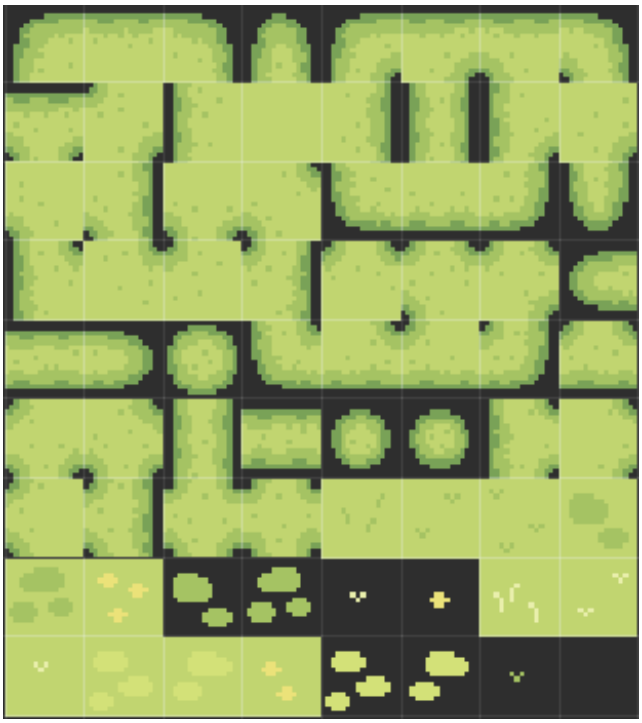

Figure 2.3: TileSet

We sourced a TileSet from itch.io, a popular platform for game assets. This TileSet provided the foundational graphics for our game's environment, enabling efficient level design and consistent aesthetics.Utilizing the imported TileSet, we constructed a simple platform to serve as the player's initial standing point. This involved arranging tiles to form a solid surface, ensuring proper alignment and collision properties for seamless player interaction.

2.2 Movement and Animation

We introduced a node to represent the player character. By assigning an appropriate sprite, we established the visual representation necessary for player identification and engagement.

```
private void AdjustPlayerFacingDirection()
{
    if (movement.sqrMagnitude > 0.01f)
    {
        // Character faces the movement direction
        mySpriteRender.flipX = movement.x < 0;
    }
    else
    {
        // Character looks towards the cursor when not moving
        Vector3 mousePos = Input.mousePosition;
        Vector3 playerScreenPoint = Camera.main.WorldToScreenPoint(transform.position);

        mySpriteRender.flipX = mousePos.x < playerScreenPoint.x;
    }
}
```

Figure 2.4: Player facing direction

To facilitate player movement, we imported the script and implemented basic movement mechanics using delta time. This approach ensures smooth and frame-independent motion, enhancing gameplay responsiveness.

We developed a method to adjust the player's facing direction based on mouse position. When the player is idle, the character orients left or right, corresponding to the cursor's location. This feature adds a layer of interactivity and realism to the character's behavior.

2.3 Enemy Implementation

We created an Enemy class in order to define what attributes all enemies should have. All common methods between the enemies are also defined there.

```
public GameObject player;


protected float speed;
protected float attackSpeed;
protected float hp;
protected float attackDamage;
protected float attackRange;
```

Figure 2.5: Enemy.cs Attributes

For now, only a Chase() method has been implemented. If placed in the Update() function of an enemy, it forces it to chase the player if he enters its range of player detection.

The primary goal for this stage of the project was to create the game's first enemy character.

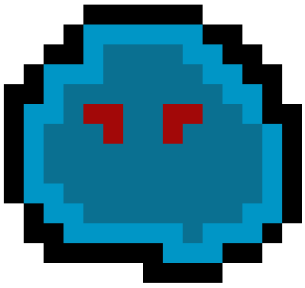

Figure 2.6: Puddle's Sprite

This is why Puddle was created, it is a simple but engaging mob designed to challenge players at the initial stages of the game. The visual assets for Puddle were created using the PiskelApp on-line sprite editor, allowing for the production of custom animations that match the aesthetic of the game.

```
Message Unity | 0 références
void Start()
{
    speed = 2.0f;
    attackSpeed = 1.0f;
    hp = 50.0f;
    attackDamage = 5.0f;
    attackRange = 0.5f;
}
```

Figure 2.7: Puddle's Sprite

Puddle is controlled by the Puddle.cs script. Since Puddle is an enemy, its class inherit from the enemy class.

Puddle's attributes are declared in the Start() function, so it is instantiated as soon as the object is created.

The last important task realized concerning the enemies is the animation.

Figure 2.8: Attack Right Animation

We used the animation creator that is included in Unity to make the animations with the sprites that we had already made.

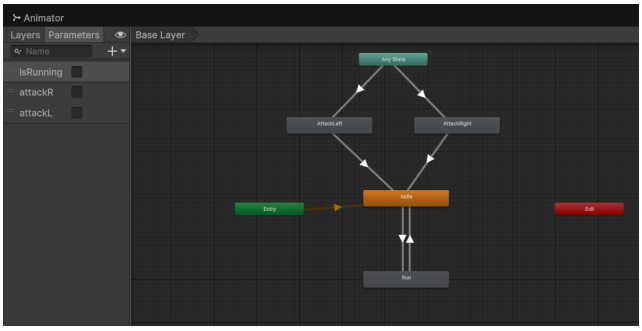

Figure 2.9: Puddle's Animator

Then, we handled the transitions with the animator tool.

2.4 Simple Attacks

We focused on creating a foundational attack system for End the Bully. This system is designed to serve as a flexible framework that can accommodate various attack types. The modular structure allows us to introduce new weapons, animations, and special effects, such as fire, poison, or area-of-effect abilities, without needing significant changes to the core mechanics. By reusing and extending this system for all types of attacks in the game, we ensure development efficiency, maintain consistency, and reduce future workload when adding new gameplay elements.

To improve the feel of combat, we worked on developing smooth and visually appealing sword animations. Our goal was to enhance both the realism and satisfaction of the combat experience. These animations are designed to strike a balance between aesthetics and responsiveness, ensuring that attacks not only look polished but also feel intuitive to the player. By focusing on animation quality, we aim to make combat engaging and immersive for players, contributing to a more enjoyable gameplay experience.

We also implemented a precise hitbox detection system for combat. This system ensures that attacks are registered accurately by defining a hit zone around the player's weapon. When an enemy is within this hit zone during an attack, damage is dealt accordingly. The hitbox system is essential for maintaining fairness and precision in combat, making every attack count and ensuring players are rewarded for skillful timing. This feature adds depth to the combat system and reduces frustration by minimizing inaccurate or missed hits.

2.5 Health System

To complement the combat mechanics, we developed a robust health system.

This system includes several key components:

Damage Handling : Players take damage when hit by enemies, with a flexible damage function that allows us to handle various types of attacks.

Healing Mechanism: Players can regain health through different methods, such as using items or abilities, managed by a centralized health function.

Respawn Logic: When the player's health reaches zero, a respawn mechanism triggers, returning them to a designated checkpoint. This adds a layer of challenge while allowing players to continue progressing without losing significant progress.

This health system creates a well-rounded gameplay loop, balancing challenge with fairness. In future iterations, we plan to expand this system by adding features such as temporary health boosts, armor effects, and status-based modifiers like regeneration or shields.

Figure 2.10: Health

2.6 Death and Respawn

If the player's health bar hits zero, they'll respawn and have the chance to restart the level, keeping the action going. To handle respawning, we created a die function. When the player's health drops to zero, this function resets their position to the coordinates (0, 0) and fully restores their health.

```
private void Die()
{
    Debug.Log("I am dead!");
    transform.position = new Vector3(0, 0, transform.position.z);
    currentHealth = 100;
    healthBar.SetHealth(currentHealth);
}
```

Figure 2.11: Die method

We implemented this functionality so that when the player's health bar hits zero, they're not simply met with a game over screen. Instead, the player will respawn at a designated starting position with their health fully restored, giving them a fresh opportunity to face the challenges ahead.

This system creates a smoother gameplay experience by allowing players to quickly re-enter the action without losing momentum. It also adds an extra layer of strategy and engagement, as players must carefully manage their health during encounters to avoid frequent respawns, which can reset their progress within the level.

By incorporating this mechanic, we strike a balance between difficulty and accessibility. The ability to restart the level after respawning ensures the game remains challenging yet fair, motivating players to refine their tactics and learn from their mistakes. This approach keeps the gameplay dynamic and rewarding, creating an exciting loop where every encounter matters.

2.7 UI

Figure 2.12: MENU

The main menu serves as the player's first interaction with the game, making its design a cornerstone of the overall user experience. I approached this task with the goal of creating a clean, intuitive, and visually appealing interface that reflects the hack 'n' slash genre. The design process involved selecting cohesive colors, fonts, and layouts to align with the game's adventurous and immersive tone. I ensured that the menu was user-friendly by designing it to be simple yet engaging, offering clear navigation to the settings, quit, and play buttons.

From a technical perspective, developing the main menu required careful consideration of functionality. Each button was scripted to perform its intended action reliably and without delay. For example, the play button was designed to transition players into the main game smoothly, avoiding lag or interruptions. I achieved this by scripting the button to preload the assets necessary for the game scene, ensuring that players experience a seamless start to their adventure. Settings Button :

Although the settings button currently serves as a placeholder, its development was guided by the need for scalability and flexibility. I structured the button to allow for future additions, such as sound volume adjustments, graphics quality settings, and gameplay preferences. Implementing this button involved creating a basic script that prepares the framework for these features while maintaining stability and reliability.

Testing was an integral part of the process. I conducted multiple iterations to ensure the button responds accurately to user interactions. Future enhancements will focus on integrating detailed options, such as sliders for audio control or

dropdown menus for resolution settings, to enhance player customization. Quit Button :

The quit button provides a simple yet essential function: allowing players to exit the game. Despite its straightforward purpose, I prioritized making the quit process immediate and intuitive. During development, I accounted for edge cases to ensure that the button operates as expected under all circumstances. This included preventing accidental exits by carefully defining the button's interaction zone and response time. Play Button :

As the main gateway to the core gameplay, the play button was one of the most critical components of the menu. Its implementation required creating a seamless transition from the menu scene to the gameplay scene. To achieve this, I designed a script that connects the button to the main game while preloading assets to reduce loading times.

The button's performance was rigorously tested to ensure reliability. This involved simulating multiple user interactions, such as repeated clicks and rapid navigation between scenes. I also focused on optimizing the transition, ensuring that it aligns with the overall fluidity of the game.

Figure 2.13: Health Bar

The health bar is an integral part of the game's interface, offering players real-time feedback on their character's status. Currently, the health bar's design is complete, visually aligning with the game's aesthetics. The design process involved selecting appropriate colors and styles to ensure the health bar is both functional and visually consistent with the game.

While the health bar doesn't yet dynamically update based on the character's health, I've laid the foundation for this feature. Future steps will involve scripting the health bar to interact with the character's health system, making it responsive to changes in health during gameplay. For example, the bar will gradually deplete when damage is taken and refill when health is restored. This will require careful coordination between the character's stats and the UI, ensuring accuracy and responsiveness.

2.8 Multi-player

Multiplayer functionality is a core aspect of our game. For its integration, we have chosen to utilize the Fusion 2 tool from Photon.

Initially, we explored the possibility of using Unity Cloud for multiplayer implementation. However, this option was deemed unsuitable for our project, prompting us to consider alternative solutions. During a conference, we learned about Photon and its suite of tools, which led us to incorporate it into our project.

Figure 2.14: Photon PUN's Logo

Photon provides a range of tools, each with its own specific features, advantages, and limitations. While following an initial tutorial, I worked with the PUN tool. Unfortunately, I discovered upon completion that PUN is an outdated tool that is not compatible with our version of Unity. Consequently, we transitioned to the Fusion 2 tool, which appears to be a better fit for our project's requirements.

Figure 2.15: Photon Fusion's Logo

Despite its potential, Fusion 2 has limited tutorial resources, and the few available guides have proven incompatible with our game's setup. To address this challenge, I am currently studying the official documentation to deepen my understanding of Fusion 2 and tailor its functionality to our game.

Conclusion

In conclusion, our progress on the game has been both rewarding and insightful. Each team member's contribution has played a crucial role in shaping the foundation of our hack 'n' slash game. Noé's work on the tilemap and animations has brought the game world to life, ensuring an immersive and seamless experience for the player. Louise's implementation of enemy movement, intelligence, and attacks has added depth and challenge to the gameplay, making every encounter engaging and dynamic. Alex's development of the health system, along with simple attacks, death, and respawn mechanics, forms the backbone of the game's combat system, ensuring a balanced and responsive experience for the player. Wachirawit's focus on the game menu and visual effects enhances the overall presentation and accessibility of the game, creating a polished first impression. Meanwhile, Hugo's efforts on multiplayer functionality have opened the door to cooperative and competitive gameplay, adding significant value and replayability to the project.

This phase of the project has also been a valuable learning experience for all of us. We encountered challenges, such as optimizing animations, ensuring compatibility between systems, and streamlining player interactions. Tackling these obstacles together not only improved our technical skills but also helped us grow as problem solvers and collaborators. Weekly meetings and consistent communication allowed us to stay aligned, share ideas, and support each other through difficult tasks.

Looking ahead, we plan to build on our current progress by expanding the settings menu, implementing a save/load system, refining the AI, and conducting extensive testing of multiplayer functionality. Our ultimate goal is to deliver a game that offers an enjoyable, smooth, and visually appealing experience for players.

Every feature we have developed so far has been designed with the player in mind. From intuitive controls to immersive visuals, we aim to create a game that not only challenges but also entertains. Through this project, we've gained a deeper understanding of teamwork, project management, and the intricacies of game design. The experience has been invaluable in preparing us for future endeavors, both individually and as a team.