

4.1/4

Technical Defence Report

Queen Amann

**Clémence GOUTAUDIER - Matteo PARNIERE-BOSQUEZ -
Yasmine DGHOUGHY - Céleste LECAMP - Mary Bruny
NGUEDIA NGUIAZON
14/01/2026**

Table of contents

1. Introduction	2
1.1 Context of the project	2
1.2 Objectives of the first technical defense.....	2
2. Project Overview and Technical Requirements.....	2
2.1 Project concept and game genre.....	2
2.2 Technologies and tools used	3
3. Team Organization and Task Distribution.....	4
3.1 Team presentation	4
3.2 Task distribution and responsibilities	6
4. Ideation and Design Phase	7
4.1 Ideation process (why / why not)	7
4.2 Brainstorming outcomes.....	7
5. Project Advancement and Proof of Concept.....	9
5.1 Global project status	9
5.2 Tutorial implementation (Proof of Concept)	9
6. Technical Implementations	10
6.1 Multiplayer and networking	10
6.2 Artificial Intelligence	12
6.3 Collision and interactions	14
7. Visuals and Audio	14
7.1 Visual inspirations and first drafts	14
7.2 Sound effects and audio integration.....	15
8. Website Development.....	16
8.1 Website objectives	16
8.2 Structure, navigation, and design.....	16
8.3 Current state of the website	17
9. Planning and Future Work.....	17
10. What we learnt.....	19
11. Conclusion.....	20



1. Introduction

1.1 Context of the project

This project is part of a collaborative academic assignment focused on the design and development of a video game. The objective is to apply theoretical and technical knowledge acquired during the curriculum to a concrete, multidisciplinary project involving programming, game design, artificial intelligence, networking, visual design, and project management.

The project is developed as a team, combining research, experimentation, and implementation. Attention is given to collaboration, technical feasibility, and creative originality. In addition to the game itself, the project also includes the development of a website and technical documentation, ensuring a complete presentation of both the creative and technical aspects of the work.

1.2 Objectives of the first technical defence

The first technical defence aims to present the current state of the project and demonstrate the progress achieved during the initial development phase. Its main objectives are to present the overall concept of the game and its design intentions, explain the organization of the team and the distribution of tasks, detail the research and design choices made during the ideation phase to do so we will demonstrate a proof of concept, including implemented gameplay mechanics as well as showcase the first technical implementations, such as multiplayer networking and artificial intelligence, present the project website and its role within the project and outline the planning and objectives for the next development stages.

This defence focuses on validating the technical foundations of the project rather than presenting a finished product. It highlights the learning process, the challenges encountered, and the solutions implemented, while establishing a clear roadmap for future improvements.

2. Project Overview and Technical Requirements

2.1 Project concept and game genre

The project is a 2D cooperative puzzle-platformer and adventure game, inspired by titles such as *Rayman* and *LittleBigPlanet*. While the game includes classic platforming mechanics (movement, jumping, level navigation), its core identity is built around cooperation rather than competition.

Players must work together to progress through the game. Many puzzles and challenges are designed so that the actions of one player directly affect the possibilities of the other, making communication and coordination essential. The game cannot be completed alone, reinforcing its cooperative philosophy.

The overall experience is intended to be relaxing and cozy, with colourful environments and a calm atmosphere. Difficulty increases progressively through levels, not by overwhelming the



player, but by introducing more complex cooperative interactions and environmental challenges. Progression is level-based, with players advancing through different regions of the world while collecting story-related elements tied to the narrative.

2.2 Technologies and tools used

The game was developed using Python as the main programming language, chosen for its readability and flexibility. We were encouraged as well to use the PyGame library for game development, including rendering, input handling, basic physics, and game logic. *Visual Studio Code* and *PyCharm* were used as the main development environments, providing debugging tools and workflow flexibility during implementation. Other libraries were used such as Sockets, Threading, JSON and Sys. To ensure a clean and coherent work we used *GitHub*, which we also used for the host of the website.

Regarding team communication, *WhatsApp* is primarily used for quick discussions and short updates, allowing team members to react efficiently when immediate coordination is required. *Discord* is used for more detailed exchanges, technical discussions, and problem-solving sessions that require longer and more structured conversations.

Notion serves as the central organizational tool for the project. It is used to gather all relevant information, document ideas, track the project's advancement, and maintain a clear overview of tasks, deadlines, and responsibilities. This combination of tools ensures effective communication, organization, and collaboration throughout the development process.

To simplify and optimize level design, Matteo developed a custom level editor specifically for the Queen Amann game. This tool allows the team to create maps efficiently by placing and removing blocks directly within a grid-based environment. Each block can be assigned different textures depending on the desired visual style of the level.

The generated maps are exported in JSON format, enabling easy integration into the game without requiring manual coding of each texture or element. The editor supports both grid-aligned placement on a 32x32 grid and free placement, offering flexibility while maintaining precision.

Additionally, the tool distinguishes collidable elements (such as ground and platforms) and decorative elements (such as houses or background assets), ensuring proper gameplay behaviour without restricting visual creativity. This level editor significantly reduces development time and will be an asset for creating and iterating on multiple levels as the project progresses.





(Personal 4.¼ level editor with a house and ground)

3. Team Organization and Task Distribution

3.1 Team presentation

Our company is called 4.¼ (Quatre-Quarts). This name reflects both our team spirit and our way of working: like a recipe, each member brings an essential part to create a balanced and coherent project.

Quatre Quarts was founded in September 2025 as part of our academic program. We had to create groups to develop a video game, and we naturally chose to work together. We already knew each other, got along well, and were curious to see how far we could go as a team. In October, we welcomed a new member, and today we are five, a number we are very proud of.

The core values of Quatre Quarts are creativity, both artistic and technical, and curiosity. We love understanding how things work, experimenting on our own, and learning by doing.

Even though this is our first video game project, both individually and collectively, we see it as a strength. Everything is new, everything is to be learned, tested, and built together. This project is a true learning adventure for us. It allows us to discover teamwork, Python development, the use of PyGame, and the different stages of a technical project including its successes and challenges. Our goal is not only to create a video game, but to grow through this experience: gaining new skills, strengthening our ability to work together, and developing a clearer understanding of the world of computer science.



Each member of Quatre Quarts has a key role:

Clémence GOUTAUDIER oversees the technical side of the project. She manages server creation and coordinates the code, ensuring that all technical components work smoothly together. She plays a key role in maintaining the stability and reliability of the project and makes sure that technical issues are identified and resolved efficiently.

Matteo PARNIERE-BOSQUEZ oversees the artistic and visual aspects of the game. He works on illustrations, the visual identity of the company, making sure that the artistic direction remains consistent. In addition, he contributes to the sound design and has already composed several original music tracks, which play an important role in creating the game's atmosphere and enhancing player immersion. He also created the player's movement code and a level editor.

Yasmine DGHOUGH supervises the coordination and overall project follow-up. She makes sure the project moves forward smoothly by maintaining good communication within the team and ensuring that everyone stays involved. She also helps keep a global vision of the project, balancing creativity and organization while keeping the team focused on the main objectives. She also created some parts of our AI such as wandering AI.

Céleste LECAMP is responsible for scheduling and the website. As Scheduling Manager, she ensures that deadlines are respected and that the project progresses according to the planned timeline. As Website Director, she designs and manages the website, focusing on the interface to help users discover and understand the game.

Mary Bruny NGUEDIA NGUIAZON brings strong knowledge in video game development to the team. She provides valuable guidance on the technical direction of the project, including gameplay systems and the user interface. She is also eager to learn and enjoys exploring new technical challenges throughout the development process. She worked on the enemy's AI.



(Drawing of the group members by Matteo Parniere-Bosquez)



3.2 Task distribution and responsibilities

Tasks	Manager	Alternate
Game Executable	Céleste	Clémence
User interface	Mary	Matteo
Game Basics	Matteo	Mary
AI	Mary/Yasmine	
Game test	Yasmine	Clémence
Story & Level Designs	Everyone	Everyone
Server & Multiplayer	Clémence	Céleste
Website	Céleste	Matteo
Sound & Graphism	Matteo	Yasmine

(table of the task distribution)

Clémence GOUTAUDIER works on the server and the multiplayer aspect of the game. She will write the CDSF and the Technical Defence Report. She also designs the *level B1* centred around Fire.

Matteo PARNIERE-BOSQUEZ works on the tutorial (storyboard and visual ideation of the tutorial). He also works on the game basics such as player movement and collisions. Ideally he will be able to draw new sceneries and characters as he oversees the artistic and visual aspects of the game.

Yasmine DGHOUGH works on the wandering AI, on the UI (User Interface) and on the *level A2* centred around Air.

Céleste LECAMP works on the website, ensures the project's timeline is respected and creates all the PowerPoints for the different defences. She also designs the *level A1* centred around Earth.

Mary Bruny NGUEDIA NGUIAZON works on the enemy's AI, on the tutorial (storyline and gameplay mechanics) and designs the *level B2* centred around Water

The “level A-B 1-2” is four possible levels of the game Queen Amann that would each have a different mechanism, storyline and design.

We expected for the first defence to have a functional server as well as a website and some basic game mechanics (such as movement and collisions).



4. Ideation and Design Phase

4.1 Ideation process (why / why not)

The ideation phase began with a wide exploration of ideas related to gameplay, narrative, visuals, and overall player experience. The objective was to create a game that would be both original and technically feasible within the given timeframe.

To structure this process, we classified ideas using a priority-based method inspired by urgency and importance. Each concept was evaluated according to several criteria: feasibility, required development time, energy cost, and number of people needed to implement it. If an idea required more than two team members to be developed simultaneously, it was considered too complex for the current scope of the project and postponed.

Another key factor in decision-making was functionality. The team systematically asked whether the game could still function without a given feature. Core mechanics such as player movement, basic interactions, and cooperative gameplay were prioritized, while minor or overly detailed features were deliberately set aside for later development stages.

This approach allowed the team to balance creativity with realism, ensuring steady progress while preserving the project's original vision.

4.2 Brainstorming outcomes

The brainstorming phase generated many ideas across all aspects of the game. These ideas were discussed collectively during team meetings, written down, and reviewed one by one. Each team member had the opportunity to present and defend their ideas, encouraging open discussion and mutual understanding.

Disagreements were resolved through argumentation and listening, ensuring that decisions were made collaboratively rather than imposed. This process helped the team converge toward a coherent concept while maintaining creativity and originality.

The results of the brainstorming phase are detailed below.

4.2.1 Game type and core mechanics

The game is designed as a relaxing, cozy, cooperative puzzle game. It is not centred on competition or high stress, but rather on collaboration and shared problem-solving. The atmosphere is intended to be colourful, immersive, and comforting.

The gameplay forces two players to progress only by working together. The players must coordinate their actions to solve puzzles, activate mechanisms, open doors, and overcome obstacles. For example, one player may need to stand on a button to open a passage for the other, or distract enemies while their partner advances.



The multiplayer experience is strictly cooperative. Progression is impossible without coordination, reinforcing the idea that players depend on one another. Each player's actions have direct consequences on the other's possibilities in future moments of the level.

4.2.2 Story and narrative ideas

The main character of the game is a dandelion, a small and curious creature symbolizing travel, transmission, and connection. The story takes place in a world affected by a mysterious curse that made their Queen disappear.

At the centre of this curse is Queen Amann, a character inspired by the kouign-amann and medieval mythical figures. To restore balance to the world, players must recreate a legendary recipe capable of breaking the curse.

The narrative emerged from a shared personal experience: despite different cultural backgrounds, the team discovered a common love for the same cake. This led to the idea of using food as a narrative bridge between cultures.

Players travel through regions inspired by European medieval settings, such as Mont-Saint-Michel, and Latin American civilizations, with architecture reminiscent of pre-Columbian temples. Each region introduces unique cultural references and ingredients tied to local traditions. As a result, the culinary quest also becomes a cultural journey.

The story is revealed through explicit narration, and progressively evolves through environments, non-playable characters, and level progression.

4.2.3 Visual direction and inspirations

The visual direction is inspired by childhood memories, video games, animated series, and fantasy worlds. The art style aims to be colourful, warm, and expressive, reinforcing the cozy atmosphere of the game.

Each region features a distinct visual identity. For example, European-inspired levels emphasize water, stone, plains, and medieval architecture whereas Latin American-inspired levels focus on vibrant colours, temples, jungles, and warm tones such as red and yellow.

All visual elements are designed to be original and handcrafted, contributing to the uniqueness of the game world. Visual progression also reflects gameplay progression: as levels become more complex, cultural references become richer and more detailed.

4.2.4 Gameplay mechanics ideas

Gameplay mechanics were inspired by analysing how other cooperative and platformer games function and why their mechanics are effective. The team explored different possibilities, such as turn-based versus real-time gameplay, and competitive versus cooperative dynamics.

After comparison, a real-time cooperative model was chosen, inspired by games such as *LittleBigPlanet* and *Rayman*, where players must react simultaneously and help one another in dynamic situations.



Mechanics include environmental puzzles, cooperative challenges, and boss encounters. For example, bosses may summon waves of enemies and become vulnerable only after a certain phase, requiring players to survive and coordinate their actions. Difficulty increases gradually across levels, both mechanically and culturally, allowing players to learn while remaining engaged.

The overall design prioritizes accessibility, cooperation, and emotional connection over difficulty or competition.

5. Project Advancement and Proof of Concept

5.1 Global project status

At the time of the first technical defence, the project has reached an early but functional stage. Core systems such as player movement, gravity, collisions with the environment, and a tutorial level have been implemented, serving as a first proof of concept (PoC).

The current focus of development has been on establishing solid foundations rather than completing all gameplay features. As a result, several systems such as combat, advanced enemy behaviour, and a complete gameplay loop are still under development. This approach allows the team to validate technical choices and ensure the game is playable before expanding its mechanics.

5.2 Tutorial implementation (Proof of Concept)

A tutorial level was implemented first to introduce players to the basic rules and controls of the game. This tutorial is designed to be the player's first contact with the game and to teach essential mechanics within the first few minutes of gameplay.

5.2.1 Player movement and controls

Player movement is fully implemented and functional. The controls are intentionally simple to ensure accessibility:

- Q: Move left
- S: Move right
- SPACE: Jump (up to two consecutive jumps)

Gravity is applied constantly to the player, and jumping temporarily counteracts this force. The double-jump mechanic allows better control during platforming sections and helps the player navigate vertical spaces.



6. Technical Implementations

6.1 Multiplayer and networking

6.1.1 Architecture Choice

The multiplayer system was designed using a client–server architecture. Initially, a peer-to-peer (P2P) approach was considered. However, this model quickly revealed several conceptual and technical limitations. It required one player to implicitly act as a server, which raised questions about connection initialization, synchronization, and scalability. Moreover, even a peer-to-peer system still requires an initial server-like component to establish connections between players. Furthermore, after some thorough investigation, we noticed that this process had many flaws and loopholes especially considering security.

For these reasons, the architecture was redesigned around a dedicated server to which all clients connect. This decision made the system more coherent, easier to debug, and more suitable for understanding how real multiplayer game servers operate. This shift marked a key learning step in understanding networking principles applied to video games.

6.1.2 Technologies and Libraries Used

The multiplayer system was implemented in Python, using the following libraries:

- Sockets: to manage low-level network communication between the server and the clients.
- Threading: to allow concurrent execution of multiple tasks.
- PyGame: for game logic and rendering.
- JSON: to serialize and deserialize data exchanged between the server and clients.
- Sys: for system-level control and execution management.

Sockets are used to establish connections, send data, and receive messages. Threading plays a crucial role, as Python normally executes code sequentially. By using threads, the server can simultaneously listen for incoming messages and send updates to connected clients, which is essential in a real-time video game environment.

6.1.3 Server Initialization and Connection Handling

The server is designed to run independently from the clients. When launched, it:

1. Initializes a host and a port.
2. Starts listening for incoming connections.
3. Accepts new clients as they attempt to connect.

The port can be compared to a “key” that allows clients to enter a specific “room” (the server). All clients use the same port to ensure they connect to the correct server instance. Once



connected, the server maintains a list of active clients and manages communication between them.

Special care was taken to handle client disconnections gracefully. Initially, when a client disconnected, the server would crash. This issue was resolved by adding proper exception handling and cleanup logic, allowing the server to continue running even if one or more players leave.

6.1.4 Data Encoding, Decoding, and Message Handling

All data exchanged between the server and the clients is transmitted in binary format, as required by socket communication. To make this data usable, a custom encoding and decoding system was implemented.

Messages are structured using JSON, allowing complex information to be transmitted, such as the player's position, colour or appearance, the player's actions (movement, attacks) and the connection and disconnection events.

Once decoded, messages are processed through an event-based system. Each message contains an action identifier, which is interpreted by decorators. These decorators act as dispatchers: when a specific action is received (for example, a player movement), the corresponding function is automatically executed in the game logic.

This system allows the game to transform raw network data into concrete in-game actions, ensuring that player behaviour is correctly synchronized across all clients.

6.1.5 Development Process and Testing

To better understand server behaviour, a chat application was developed as an initial prototype. This application allowed multiple clients to connect to a server and exchange text messages. Testing this system locally on two separate computers made it possible to validate message transmission, reception, and broadcasting.

Once the chat system was functional, it was progressively adapted into a game-specific server. Human-readable messages were replaced by structured game data, such as player state and actions. One of the main challenges encountered was correctly processing received data and displaying remote players on screen. Initially, data was received correctly but not rendered, which led to the false assumption that communication was failing. This issue was resolved by properly integrating received data into the game loop.

Extensive local testing was conducted to ensure stability. The server was kept running for long periods, multiple clients were connected simultaneously, and edge cases such as abrupt disconnections were tested. The current multiplayer implementation is functional and stable.



6.2 Artificial Intelligence

Artificial intelligence was implemented to control enemy behaviour to create a more dynamic and engaging gameplay experience. The objective is to give enemies simple but believable behaviours, making them appear active within the environment while remaining predictable and balanced for the player.

The long-term goal of the AI system is to allow enemies to wander around their initial position, detect the player when they enter a specific range, chase them, and eventually attack if the player gets close enough. If the player manages to escape, the enemy should stop the pursuit and return to its initial passive behaviour. This state-based logic ensures a clear and understandable enemy reaction system for the player.

To achieve this behaviour, the artificial intelligence is divided into two main components:

6.2.1 Enemy wandering behaviour

Each enemy is initialized with a predefined size, position, and movement speed. The speed of the enemy is intentionally set lower than the player's speed to maintain balanced gameplay.

When the player is outside the detection range, the enemy remains in an idle (passive) state. During this state, the enemy does not move, attack, or react. At each frame, the system compares the horizontal position of the player with the position of the enemy and calculates the absolute difference between these two values. This calculation represents the distance separating the enemy from the player along the x-axis.

If this distance is greater than the defined detection threshold, no behavioural condition is triggered. As a result, the enemy remains inactive and does not modify its position or state. This passive behaviour allows enemies to appear present in the environment without constantly interacting with the player.

At the current stage of development, detection thresholds and transition smoothing are basic. More advanced detection ranges and progressive triggers are planned for future iterations.

6.2.2 Danger detection and attack behaviour

When the distance between the player and the enemy becomes smaller than the detection range, the AI transitions into an active state. The enemy's behaviour then depends on how close the player is.

If the player is within a medium distance, the enemy initiates a pursuit behaviour. The system compares the horizontal position of the player with that of the enemy to determine whether the player is located to the left or to the right. Based on this comparison, the enemy moves in the corresponding direction to reduce the distance separating them. The pursuit speed remains lower than the player's speed to prevent unfair difficulty and allow the player to escape.



When the player becomes very close, the enemy switches from pursuit to attack mode. In the current implementation, the attack behaviour is represented by a jump. The jump state is activated by assigning an upward vertical velocity to the enemy. Gravity is then applied at each frame, pulling the enemy back toward the ground. The vertical position is updated continuously, producing a natural jump-and-fall motion.

The attack state remains active until the enemy lands on the ground. Once contact with the ground is detected, the vertical movement stops, the attack state is cleared, and the AI evaluates the player's position again during subsequent frames.

If the player moves away while the enemy is active, the AI exits the attack state and returns to pursuit mode. If the player exceeds the detection distance, the enemy exits the active state entirely and returns to its passive idle behaviour.

6.2.3 Enemy Variations, Obstacles, and AI Structuring

To test and validate enemy logic, a separate main testing environment was developed. This allowed enemies and obstacles to appear sequentially, ensuring that their behaviours functioned correctly before full integration into the game. This testing phase was essential to verify movement logic, detection behaviour, and interaction timing.

In addition to the core enemy behaviours, several enemy types and obstacles were designed to test different mechanics and gradually introduce difficulty in our work. Two main enemy categories were implemented: a small enemy and a large enemy. Both enemies share the same general AI structure but differ in their characteristics and impact on the player.

The small enemy is designed to detect the player from a short distance and causes limited damage. It patrols or remains idle when the player is not nearby, allowing the player to familiarize themselves with basic enemy interactions. In contrast, the large enemy has a wider detection range and inflicts greater damage on the player's health, increasing the level of threat and encouraging more cautious gameplay.

In addition to enemies, two types of obstacles were implemented: a falling obstacle and a projectile-shooting obstacle. These obstacles are currently not fully integrated into the player's progression but serve as a foundation for future level design and gameplay challenges. Their purpose is to introduce environmental hazards that interact with the player independently of enemy AI.

During development, the AI logic was structured using a divide-and-conquer approach, where enemy behaviour was split into multiple smaller components. Each component was implemented and tested independently before being combined into a complete system. This method made the AI logic easier to debug, understand, and extend.

One of the main challenges encountered during this process was designing a coherent progression system. Initially, it was difficult to define how enemies and obstacles should appear in a logical order to support player learning. To overcome this, inspiration was taken from various 2D platformer games, which helped shape a more structured gameplay flow.



Through this work, valuable experience was gained in using PyGame, structuring AI logic, and managing incremental development. The process highlighted the importance of perseverance, experimentation, and iterative improvement when building complex systems such as artificial intelligence.

6.3 Collision and interactions

6.3.1 Collision with environment

Environmental collisions are fully functional and form a core part of the platforming mechanics. Gravity is constantly applied to all entities in the game. Terrain blocks act as floors, walls, and ceilings, preventing entities from passing through them. The player can jump up to two times to temporarily counter gravity. Visual assets and background textures are purely decorative and do not have collision properties. This separation between collision objects and visual elements ensures clarity in level design and simplifies future modifications.

7. Visuals and Audio

Visual and audio integration are key components of an immersive and engaging game therefore having an overview early on was necessary to understand and determine the direction the project would take. It was decided that all the artistic elements would be handmade and the use of generative artificial intelligence nor the use of ready-made image and audio banks would be considered. Some close friends and relatives even kindly offered to take part to help us with sound effects and a fully animated cinematic.

7.1 Visual inspirations and first drafts

The visual direction of the game draws inspiration from several well-known titles while maintaining an original artistic identity. *Rayman (1995)* inspired the terrain and environmental elements, particularly grassy platforms whereas *Hollow Knight (2017)* influenced the design of the dandelion character.

The initial character concepts were created as post-it doodles, allowing rapid experimentation with shapes and ideas. These sketches were then refined and redesigned multiple times. The characters were ultimately drawn and animated using Procreate, with sprites created at a resolution of 17×35 pixels.

An early concept based on elemental characters (earth, fire, water) was explored but later abandoned in favour of designs inspired by symbolic and cultural representations:

The dandelion symbolizes travel and adaptability, as it grows everywhere. The skeleton character draws inspiration from Latin American culture, including traditional clothing elements such as a poncho.





(Drafts of main characters)



(draft of a possible character, the design was later abandoned)

This iterative process allowed the visuals to evolve organically until a convincing and coherent artistic direction was achieved.

7.2 Sound effects and audio integration

The game's audio direction aims to create a soothing and immersive atmosphere. The first musical track is composed in D minor.

As for the main inspiration, it comes from the Minecraft soundtrack, particularly its calm and ambient tone. Instruments such as violins were chosen to evoke a medieval atmosphere. Both the music and visual assets were created during personal work time outside of scheduled development sessions. While the audio system is still minimal, it proves a strong foundation for future sound design and musical expansion.

The music was the first element developed by the team. Even before the ideation phase was fully completed, a first musical proposition was created. This early work on sound design played a key role in defining the overall ambiance of the game. The calm and soothing atmosphere set up by the music helped guide later artistic and design decisions, ensuring coherence between gameplay, visuals, and audio direction.



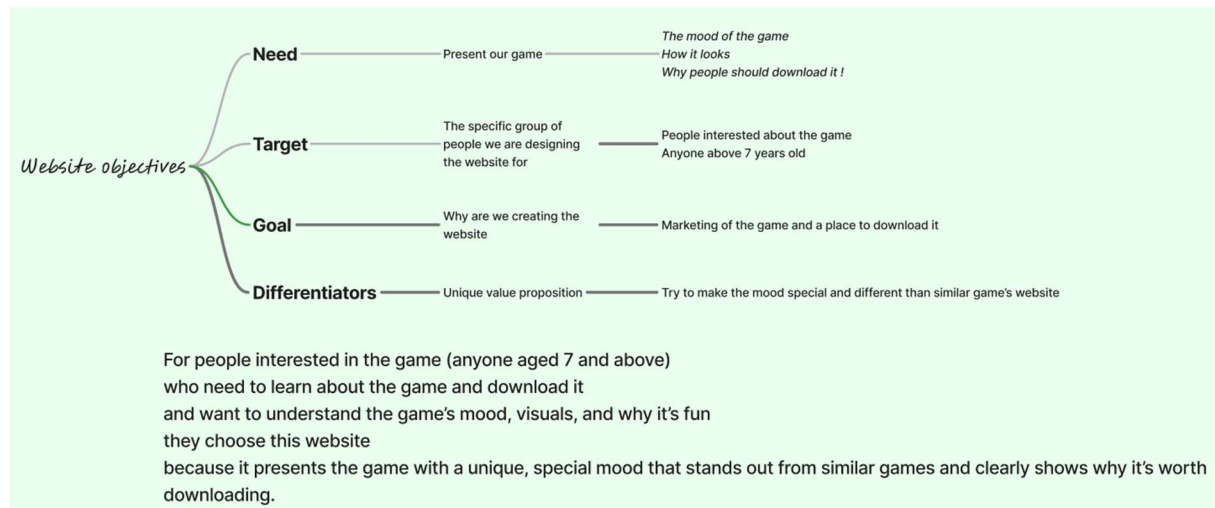
8. Website Development

8.1 Website objectives

The main objective of the website is to present the game project in a clear, attractive, and accessible way. It serves as a communication and promotional tool, allowing visitors to understand the game's concept, mood, and visual identity.

The website is designed for anyone aged 7 and above who is interested in discovering the game, learning about its story and mechanics, or downloading the project. It targets players, teachers, and potential users who want a concise overview of the game without requiring prior technical knowledge.

Another important objective of the website is marketing and visibility. It highlights what makes the game unique compared to similar titles, especially its cooperative gameplay, cozy atmosphere, and strong artistic direction. The website also acts as a central hub where users can access project documents, visual content, and contact information.



(Figma scheme of the website objectives)

8.2 Structure, navigation, and design

The website is organized into four main sections to clearly separate content and improve readability:

- Home
- Documents
- Gameplay and Graphics
- About

Us



Each page follows a consistent layout to ensure a smooth and intuitive user experience. A fixed navigation bar is present on all pages, allowing users to move easily between sections without needing to scroll back to the top. This design choice improves accessibility and usability across the entire website.

The Home page features a prominent hero section that immediately introduces the project, presents its atmosphere, and captures the visitor's attention. Below this section, each page contains a main content area where information is presented in a clear and structured manner. A contact section is also included, providing users with a simple way to reach the team.

To enhance navigation on longer pages, a “back to top” feature is implemented, allowing users to quickly return to the top of the page.

The visual design of the website was created using Figma, ensuring consistency in typography, colours, and layout. Both desktop and mobile versions were designed to guarantee responsive behaviour across different screen sizes. These mock-ups define the overall visual identity of the website and reflect the artistic direction of the game.

8.3 Current state of the website

The website is currently online and hosted on GitHub Pages. It is fully functional and accessible to users.

At its current stage, the website provides:

- Information about the project and team members
- A presentation of the game's story and narrative universe
- Access to downloadable documents such as:
 - Technical Specifications Document
 - Functional Specifications Document
 - Technical report 1
 - Technical Presentation Plan 1
- A contact email, allowing users or visitors to get in touch with the team

9. Planning and Future Work

As this is our first technical defence report many things are to be done in the next few months. This is what the team would like to achieve and finish, as well as possible implementations or improvements.



9.1 Features to implement next

At this stage of development, combat mechanics are not yet implemented. The tutorial focuses solely on movement and navigation to avoid overwhelming the player. Combat systems, including attacks and health management, are planned for future iterations.

Enemy collisions are not implemented, as enemy entities are still under development. This system will be added once enemy behaviours and combat mechanics are integrated into the game.

Future work includes fully integrating enemy AI with player mechanics, implementing health bars and user interface elements, and refining the balance between enemy difficulty and player progression.

Detection ranges, difficulty levels, and behavioural transitions are fixed and cannot be changed by the player. Enemy physics are also simple, relying only on gravity and jump mechanics. Environmental factors such as variable gravity or terrain effects are not yet implemented.

It is envisioned that the attack system will be implemented. In addition to entity health mechanics

9.2 Planned improvements (levels, multiplayer, content)

A complete and finalized gameplay loop has not yet been fully implemented. However, a preliminary concept has been defined:

- Discover a new level and learn new mechanics
- Explore the environment and collect ingredients
- Face a boss encounter
- Exit the level and progress to the next area

This structure will serve as the foundation for future gameplay development and level design.

At its current state, the multiplayer system allows multiple players to connect to the same server, players to choose display names and real-time synchronization of player presence and actions.

Possible improvements include:

- A room-based system allowing players to join specific game sessions
- A database to store player accounts, names, and potentially scores
- Authentication using passwords
- Improved scalability and data management



While some content is still being refined, the website already fulfils its primary goals of presentation, documentation, and communication, and serves as a solid foundation for future updates and improvements such as Visual content, including images of the game

Each task was done individually meaning we will have to compile the codes and assemble each work so that it becomes one game. This will be a challenging but necessary step.

10. What we learnt

Throughout the project, the team gained valuable insights. We learned that asking for help is essential. Matteo and Clémence, for example, received help from a relative when tackling complex mechanics. This highlighted the importance of seeking help and using external knowledge. We discovered that there are often multiple ways to achieve the same goal in programming. This flexibility allowed us to explore different approaches and find the most effective solutions. With PyGame, we learned the importance of specifying every action. Since PyGame is not a full game engine, it requires more manual implementation of mechanics, making mindfulness crucial.

One of the biggest challenges was understanding how to design the game world with an effective camera system. Some of us addressed this by consulting with experienced PyGame developers and studying common platformer camera implementations. This research led to a more intuitive camera system that enhances player experience. Setting up the multiplayer server posed a significant challenge due to the complexity of networking protocols. The team navigated through extensive documentation and explored various approaches (such as TCP and UDP) before settling on the most suitable solution. The team's persistence and collaboration ultimately led to a stable and functional server system. One of the key lessons was the importance of motivation and effective communication within the team. Initially, the team faced challenges due to a lack of experience and direction. However, through continuous research, clear communication, and setting realistic deadlines, the team overcame these hurdles. This experience taught us the value of patience, organization, and perseverance.

Throughout the development process, we had the opportunity to engage with professionals in the field. For instance, Céleste consulted with experienced developers to understand technical aspects of programming and problem-solving strategies. This helped her gain insights into approaching challenges with an engineer's mindset. Similarly, Clémence learned valuable lessons in team management, including how to foster effective communication and ensure that every team member felt respected and heard.

These experiences emphasized the importance of collaboration and patience. We also recognized that motivation plays a crucial role, especially when facing challenges and starting from scratch. Ultimately, these lessons not only helped us overcome technical hurdles but also strengthened our teamwork and our project.



11. Conclusion

Queen Amann is an ambitious project our team, *Quatre Quarts*, has embarked on. Although none of the members had prior experience in such programming or communication skills, every single member has done their best to successfully complete each of their tasks accordingly, and while respecting the time constraints, to serve a delicious taste of a captivating and immersive game, despite it not being completely operational yet.

We may have met issues regarding communication and organisation; however, we have managed to go through them and solve every problem we had by changing our manner of working together, favouring an active and collaborative communication platform. We have also had to change our organisation to fit our standards for an efficient and satisfying progression on the game. This project has also allowed us to study the aspects we should consider when working on a big scale project, as we should find when working in a company.

To sum up, our group has worked on the basic mechanism of the players' movement and collision with their surroundings. AI wandering and attack work have been done and need to be refined. The server is functional and waits for improvement allowing two players to connect on the same game. The website is fulfilling its requirement and is published. The visuals and sound effects are to be enriched. Most importantly all these tasks still need to be compiled into one single code.

Queen Amann is our first project, of course, but it testifies of our will to work together, as well as of artistic creativity. It is a promise to keep trying our best to bring this game to life. We are very proud of our accomplishments so far, and we congratulate everyone who is working or had to work on this project.

