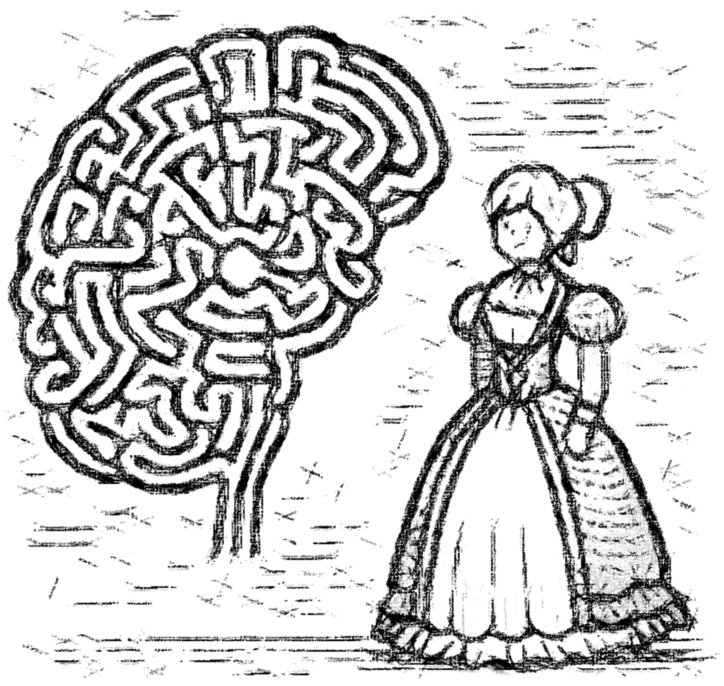
MMI716 – Game Programming Patterns

Project Phase II Report



Burcu Alakuş
2459410

List of Figures

Table of Contents

1. Introduction

In this report, I will provide an overview of the second phase of my game programming project. I will begin by presenting a summary of the concept and objectives for Phase I of the project, offering insight into the foundational ideas guiding its development. Following this, I will examine the specific game components utilised, including aspects such as physics, audio, and artificial intelligence, and how they contribute to the overall gameplay experience. Additionally, I will discuss the design patterns employed in the project, detailing their applications and benefits in various tasks within the game's architecture. The report will also include an evaluation of the progress made thus far, quantifying the completion percentage, and outlining the remaining tasks. Finally, I will outline the projected timeline for the subsequent phases, ensuring that all goals are met within the designated time frame. This document aims to not only track the current progress but also to organise the efficient advancement of the project moving forward.

2. Project Summary

Palette of Memories: The Artist's Path to Remembrance is a first-person exploration game about a female painter who loses her memory. Players navigate through rooms, capturing unique aspects of her artwork and memories. The goal is to find items that activate her memories, combining puzzle-solving and storytelling for an emotional exploration. It is a first-person exploration game about a female painter who awakens with no memory of her past. Set in a strange ever-changing location, the game takes players on a quite emotional journey through the protagonist's mindset and artwork. Players will move through a sequence of rooms, each capturing a unique aspect of the artist's lost memory and including the colours and themes from her earlier artworks. The goal is to find specific items within these artistic locations that activate pieces of her memories, helping her piece together her identity and background. The game combines puzzle-solving and storytelling, inviting players to solve puzzles about the painter's life and the importance of her art. This game is a one-of-a-kind examination of memory, identity, and creation that features deep gameplay and emotional depth.

3. Game Components

In the discussion of my game components, I will go into the interactive elements that constitute my game’s environment, such as the house generation system, which leverages a binary space partitioning algorithm to intricately layout houses with rooms and corridors, and the integration of physics engines for realistic object interactions, as well as assets, ghosts, environmental components that breathe life into the game.

3.1. House Creator

In my game, I’ve implemented a procedural generation technique using a house generator based on the binary space partitioning (BSP) algorithm. This method is a systematic approach to dynamically create a complex layout of a house complete with rooms and corridors. BSP is a recursive division of space. Imagine it as continually splitting a space into subsections using imaginary, straight lines (or planes in 3D). In the context of my game, this space represents the interior of a house. The initial space is the entire area of the house, which the BSP algorithm divides into smaller sections, each potentially becoming a room or a corridor. One of the outputs of this script is as in Figure 1, I have chosen this layout as my house plan. The parameters for creating a house can be seen in Figure 2.

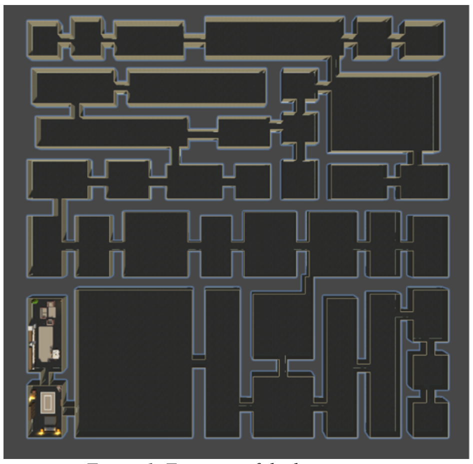
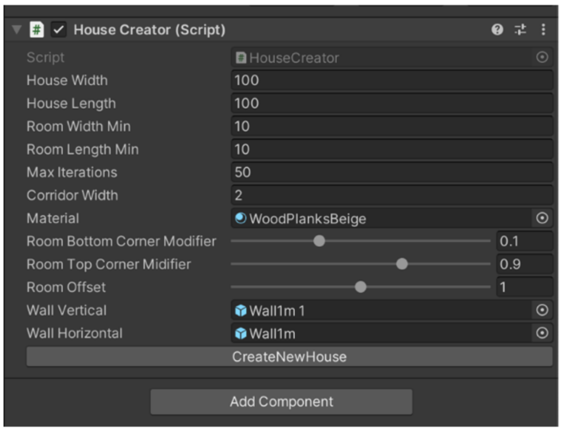


Figure 1. Top view of the house.



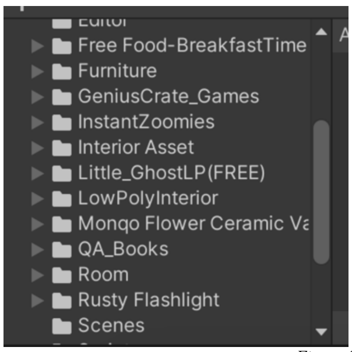
Script	HouseCreator
House Width	100
House Length	100
Room Width Min	10
Room Length Min	10
Max Iterations	50
Corridor Width	2
Material	WoodPlanksBeige
Room Bottom Corner Modifier	0.1
Room Top Corner Midifier	0.9
Room Offset	1
Wall Vertical	Wall1m 1
Wall Horizontal	Wall1m

Figure 2. HouseCreator script parameter view in Unity UI.

By setting parameters such as the minimum and maximum size of a room, I can control the outcome of the generation to ensure that the spaces created are usable within the game’s context — not too small to move around in and not so large that they don’t resemble realistic rooms. Once the algorithm has partitioned the space into a satisfactory layout of rooms and corridors, the maze generator comes into play. It creates pathways between the rooms, ensuring there are no inaccessible areas. These pathways are the corridors of the house, and the logic of the maze ensures that they interconnect the rooms in a complex, yet navigable, manner.

3.2. Assets

In crafting the environmental and aesthetic dimensions of my game, I've incorporated a suite of free assets that greatly enhance the visual richness and atmosphere. These assets, see Figure 3, collected from Unity’s AssetStore, encompass textures and room designs that contribute distinct atmospheres to various game settings. I've employed available 3D models to furnish the game with interactive elements and structures, while using tile sets for creating diverse terrains, thus eliminating the need for extensive custom graphics design. Particle effects from these free resources have been used to add environmental dynamics, such as the play of light and shadow or the animation of natural elements. The utilisation of these assets has been executed with attention to the licensing conditions, ensuring that the integration within my game is both ethical and legal. This approach not only saves development resources but also allows for a more efficient design process while maintaining academic and professional standards.



- Free Food-BreakfastTime
- Furniture
- GeniusCrate_Games
- InstantZoomies
- Interior Asset
- Little_GhostLP(FREE)
- LowPolyInterior
- Monqo Flower Ceramic V
- QA_Books
- Room
- Rusty Flashlight
- Scenes

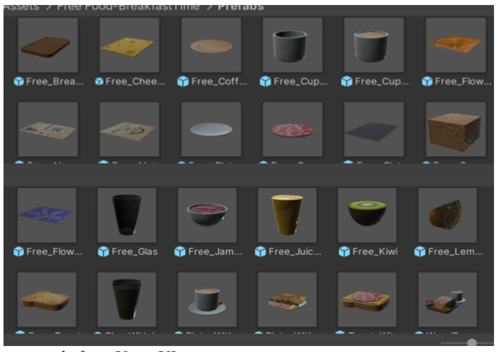
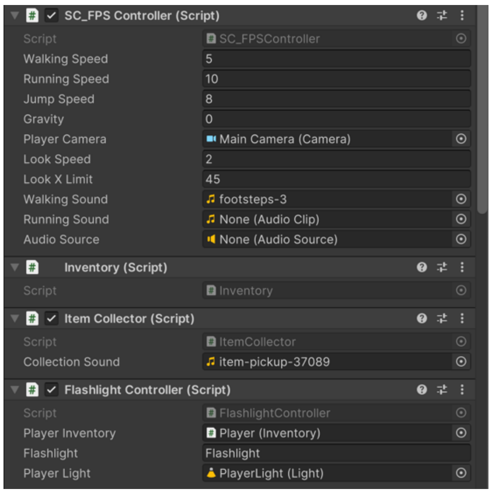


Figure 3. *Assets sample from Unity UI*

3.3. Player

The player component in the game is equipped with a collection of scripts that manage various aspects of gameplay, as in Figure 4, script UI. The FPS Controller script adjusts the player's movement and camera control, setting how it walks, runs, jumps, and looks around within the game's environment. There's also an inventory script that handles the items the player collects, storing them and enabling their use at key moments. An item collector script allows for interaction with collectable objects, a sample is given in Figure 4, adding them to the player's inventory. Lastly, a flashlight controller script manages a flashlight's functionality, critical for navigation and interaction within darker areas of the game. Together, these scripts define the player's interaction with the virtual world, making for a cohesive and engaging gameplay experience.



SC_FPS Controller (Script)

Script	SC_FPSController
Walking Speed	5
Running Speed	10
Jump Speed	8
Gravity	0
Player Camera	Main Camera (Camera)
Look Speed	2
Look X Limit	45
Walking Sound	footsteps-3
Running Sound	None (Audio Clip)
Audio Source	None (Audio Source)

Inventory (Script)

Script	Inventory

Item Collector (Script)

Script	ItemCollector
Collection Sound	item-pickup-37089

Flashlight Controller (Script)

Script	FlashlightController
Player Inventory	Player (Inventory)
Flashlight	Flashlight
Player Light	PlayerLight (Light)

Figure 4. *Player scripts and some items with "Collectable" tag*

3.4. Ghosts

In the game, an engaging component of the environment is the presence of ghosts, which are programmed to enhance the thrill and challenge of gameplay. These spectral entities are designed to materialize when the player acquires certain items, triggering their appearance as if awakened by these objects. Once active, they pursue the player, adding a layer of urgency and danger to the navigation of the game's spaces. The ghosts also emit creepy sounds that contribute to the haunting atmosphere, heightening the sense of suspense and immersion. However, their presence isn't permanent; the ghosts are coded to vanish when the player has either put enough distance between themselves and these apparitions or after a set duration, ensuring that the game remains balanced between challenge and playability. This mechanic not only adds depth to the gameplay but also serves as a motivator for players to continue exploring and interacting with the game world, pushing forward to uncover more of the game's mysteries. The ghost types can be seen in Figure 5.

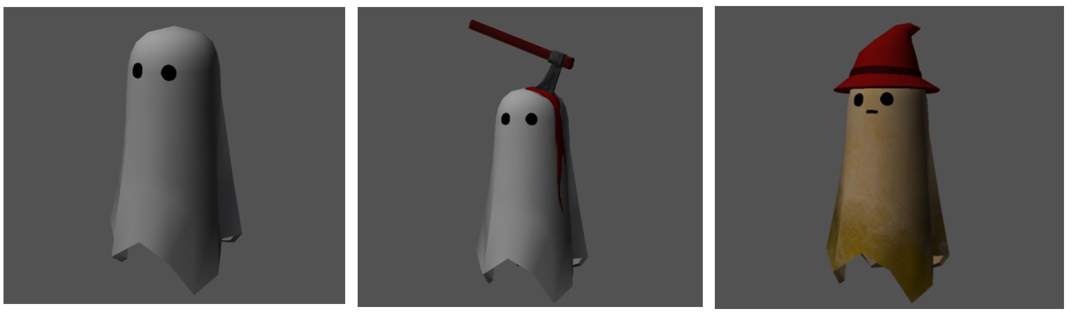
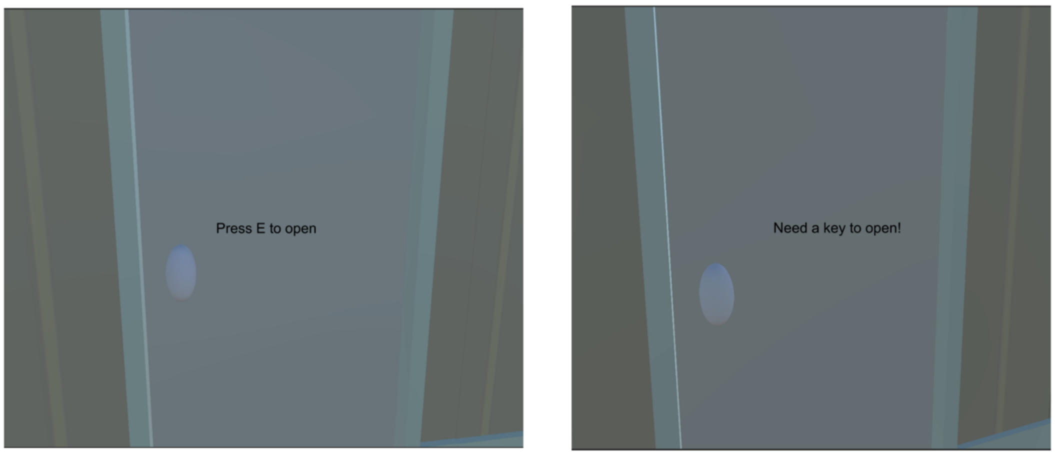


Figure 5. Ghost types

3.5. Doors & Keys

Within the game's design, a classic yet effective gameplay mechanic is the incorporation of locked doors and the requisite keys needed for their opening, as seen in Figure 6. This component plays a strategic role in guiding the progression through the game. Some rooms within the game environment are inaccessible from the outset, locked, prompting a search for the keys. This search encourages exploration and interaction with the game's


Press E to open

Need a key to open!

Figure 6. Openable and locked doors

environment, as the player must navigate through the rooms and corridors, all while potentially avoiding threats such as ghosts. This mechanic not only serves to structure the player's journey through the game but also adds layers of engagement and problem-solving, as players must remember which doors they have encountered and return once they have found the correct keys. It's a balance between exploration and reward, providing a satisfying sense of accomplishment as previously locked areas become accessible and reveal their sense of accomplishment as previously locked areas become accessible and reveal their secrets.

3.6. Environment Lightning

To craft a distinctive atmosphere within my game, I've chosen to avoid the use of a traditional skybox material. Instead, I've embraced the use of pervasive fog throughout the game environment. This design decision increases the mysterious and possibly claustrophobic atmosphere, directing the player's attention to their immediate surroundings and generating a sensation of solitude. The absence of a skybox means that there are no far views to provide navigation or distraction, which increases the player's immersion in the game world, with and without views of a skybox can be seen in Figures 7 & 8. The fog not only hides distant objects, providing mystery and suspense, but it also helps to improve game performance by reducing the rendering burden of the distant landscape can be seen in Figures 9 & 10. This combines both aesthetic intent and technical practicality, ensuring that the player remains grounded in the game's immediate narrative and visual experience.

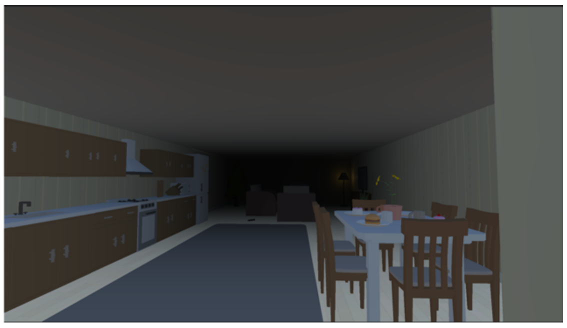


Figure 7. Without skybox view.

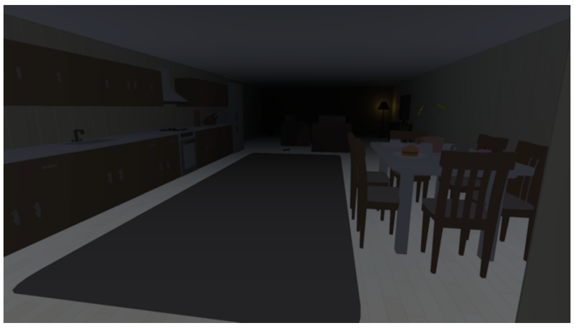


Figure 8. With skybox view.

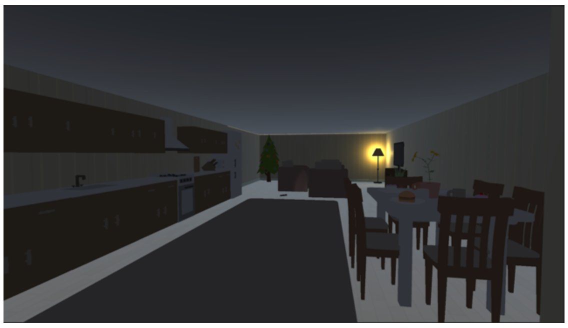


Figure 9. Without fog view.

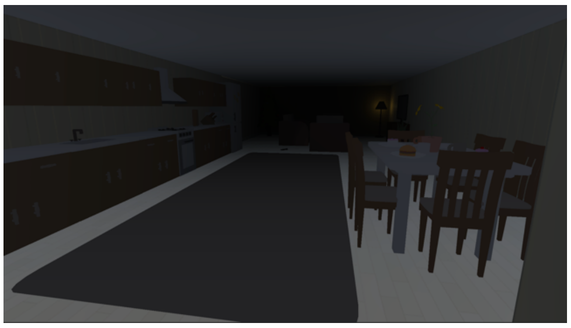


Figure 10. With fog view.

3.7. Audio

The audio dimension of my game is thoughtfully designed with a selection of sounds that are integral to the gaming experience. The opening of doors is accompanied by different sound cues that not only indicate a transition from one area to another but also add to the environment's realness. Creepy sounds attributed to ghosts create a disturbing atmosphere, adding to the tension and suspense that are key to the game's horror concept. Collecting items causes a distinctive sound, which provides the player with quick auditory feedback, reinforces the activity, and contributes to an enjoyable experience. In addition, I intend to include a variety of walking and running noises that change depending on the sort of floor the player is on. This feature not only involves the player more in the game world by reflecting the physical properties of the environment, but it also improves the overall ambience by providing a more dynamic and responsive audio system. The Figure 11 shows the audio files used in the game.

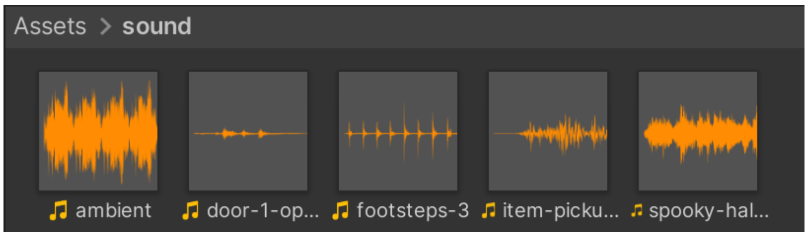


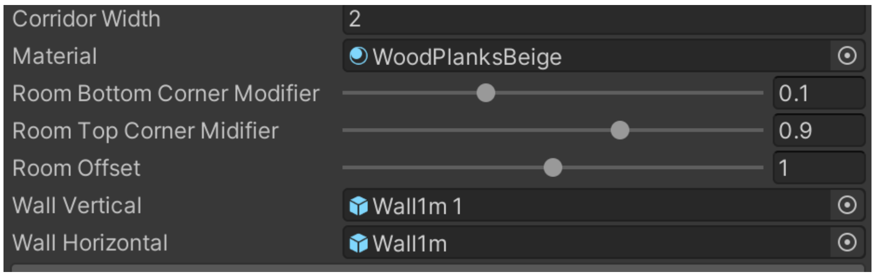
Figure 11. Audio files used in the game.

4. Design Patterns

In my game development, I've carefully used design patterns to simplify the production and maintenance of game components. The Flyweight and Prototype patterns are critical for quickly handling in-game items and characters, whilst the Command pattern (not yet implemented) ensures a seamless user experience with responsive controls. These patterns form the basis of my game's architecture, assuring performance and versatility.

4.1. Flyweight

In the architectural design of my game, I used the Flyweight pattern in the creation of the house interiors, such as walls and floors. This design pattern is crucial for optimising memory usage and performance, particularly when dealing with many similar objects within the game's world. By using BSP, I can efficiently generate a complex layout of rooms and corridors. This approach is further refined by cloning a single prefab—an archetype of an object in the game—for various instances of walls and floors throughout the house, as seen in Figure 12. While each instance is a clone of the original prefab, its position is uniquely determined by the binary space partitioning process, allowing for a variety of room shapes and corridor connections while maintaining a consistent look and feel. This not only saves resources but also allows for a diverse and engaging game environment to emerge from a relatively limited set of base components. Through this method, I'm able to create an intricate and memory-efficient house layout that is both functional for gameplay and varied enough to maintain the player's interest.



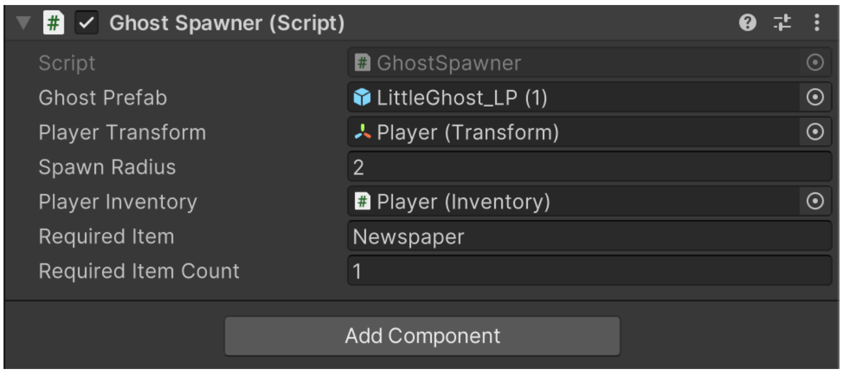
Corridor Width	2
Material	WoodPlanksBeige
Room Bottom Corner Modifier	0.1
Room Top Corner Midifier	0.9
Room Offset	1
Wall Vertical	Wall1m 1
Wall Horizontal	Wall1m

Figure 12. HouseCreator uses prefabs to clone them.

4.2. Prototype

For the dynamic appearance of opponents within my game, I've implemented the Prototype design pattern through the 'Ghost Spawner' system. This system utilizes a ghost prefab, a template for ghost entities, which can be instantiated multiple times within the game world. The Ghost Spawner is configured to track the player's position and inventory, spawning ghosts within a certain radius once the player has collected specific items.

The spawn conditions, managed by the Ghost Spawner script, seen in Figure 13, ensure that ghosts only materialize under specific circumstances, adding an element of strategy to the gameplay. Players must consider the implications of collecting items that might trigger the appearance of these ghost enemies. The Prototype pattern allows for the efficient creation of each ghost instance with unique parameters while avoiding the overhead of creating each from scratch. This method not only provides a reliable way to expand the challenges faced by the player but also streamlines the game development process.



Ghost Spawner (Script)

Script	GhostSpawner
Ghost Prefab	LittleGhost_LP (1)
Player Transform	Player (Transform)
Spawn Radius	2
Player Inventory	Player (Inventory)
Required Item	Newspaper
Required Item Count	1

Figure 13. GhostSpawner System

5. Progress and Completion Evaluation

The progress section of my report focuses on improvements made in several aspects of game development. I've finished the technical components, which comprise the core coding structures and menus, and I intend to implement the command pattern for efficient in-game command processing. During the layout and design process, the room interiors were methodically organised, and the necessary assets for these spaces were incorporated.

I'm currently working on the storyline and collectable items, both of which are important to the game's plot. Once completed, these will be linked into the rooms, giving the player's adventure more depth and meaning. Gameplay instructions are also being developed to guide players from the beginning and perhaps throughout the game, ensuring a clear knowledge of game mechanics and objectives.

In the final stages, all these individual parts will be combined to create the entire game. This integration is critical because it allows the narrative, gameplay mechanics, and visuals to work in together, presenting the player with a seamless and appealing experience. Overall, I believe I have accomplished 35-40% of the tasks; I am hoping that by completing the harder pieces, I will be able to progress more quickly in the remaining parts.

6. Project Timeline

I continue my game development journey with a clear plan, setting the stage with the story and items by April 28th, laying down the narrative foundation that would drive the entire game. By May 5th, I will bring the rooms of the game to life, crafting the environments that players would explore. The next step is to integrate the coding with the story; by May 12th, the two are seamlessly tied together, making the game's heartbeat with every line of code. I then will focus on the auditory experience and the user interfaces, polishing the sounds and menus by May 15th to ensure an immersive experience. Testing is critical, and from May 15th to the 19th, I will iron out any issues, ensuring that everything works as intended. This difficult process is all in preparation for the final milestone: the game release will be on May 23rd, as all shown in Figure 14.

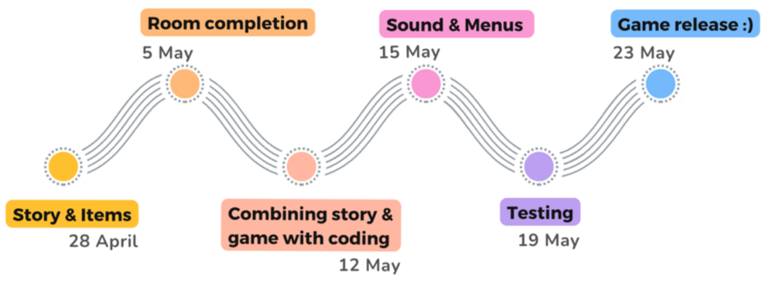


Figure 14. Timeline for the rest of the game development