Epoka University

Faculty of Engineering and Architecture

Department of Computer Engineering

CEN 302 – Software Engineering

Worlds Around™
Requirements Specification

M01, 2023

Worlds Around™

Third Draft

May 05, 2023

Prepared by:

Arbin Bici

Elvjo Fejzo

Emis Reka

Enri Shtjefni

Eralba Korbi

Eva Veli

Marvin Hoxha

Terens Tare

Received by: M.Sc. Ari Gjerazi

Table of Contents

1. Executive Summary

1.1 Project Overview

In our ever-evolving world, computer games serve as the main form of entertainment available. Their popularity has increased ever since the release of the original Pong game. This increase in popularity ignited a rivalry in the game development industry, which forced developers to improve their games on every possible aspect to increase the sales rate. Hardware limits were often passed as if they didn't exist in the first place, with the focus of these optimizations being the visuals offered and making the game playable on limited (usually last-generation or older) hardware.

Nowadays, the hardware available can handle a wide range of games, even on low-end devices, thus easing the process of developing games that do not contain a huge number of objects in them. AI and physics-related code are an exception to this statement, where research is still undergoing on implementing more efficient algorithms or improving existing ones. There are still other things that need to be taken into consideration when developing a game. On this document, we present the process of developing a game, describing the requirements of said game, the problems and shortcomings encountered, as well as the solutions proposed to address them.

The game that will be described is a platformer-style game, a choice made based on two main factors: first, the genre is popular on both AAA studios and indie developers, thus allowing for both points of inspiration for game mechanics and UI design, as well as for comparison of these aspects as well, using existing games as a "standard". Second, the game incorporates many aspects of game development, such as UI design (different in-game menus), animation (player, objects in the world, etc.), physics (jumping, colliding with objects, etc.), AI (for enemies), serialization (saving/loading data), thus making it suitable to showcase different aspects of game development. We will further expand on each of these aspects, and others such as player controls and interaction with the surroundings, describing approaches we have considered, with the benefits and shortcomings of each. Along with it, we will provide sketches of the game's design on different stages of the game to make visualization easier.

The game's main objective will be to provide the player with a memorable experience. One of our goals as developers is to build the gameplay such that it allows for simple, yet configurable controls. To further help players, the game will demonstrate new mechanics and concepts to players as needed, by effectively providing built-in tutorials for such mechanics.

1.2 Purpose and Scope of this Specification

The purpose of this specification is to define the requirements for a platformer-style game that provides players with an engaging and entertaining experience. This document outlines the game's objectives, mechanics, and features, as well as the technical specifications required for its development. The specification serves as a guide for the game development team to ensure that the game meets the desired quality standards, is delivered within the specified budget, and is completed within the agreed timeline.

The scope of this specification is to define the requirements for developing a platformer-style game that includes multiple levels, characters, enemies, and obstacles. The game is designed to provide a fun and challenging experience for players, with various game mechanics and features that keep the players engaged. This specification covers the game's objectives, mechanics, and features, including player controls, user interface, graphics, sound, music, and game progression. The specification will also define the technical specifications, such as the platforms, programming languages, software tools, and hardware requirements necessary for game development. However, this specification does not cover the marketing, distribution, or monetization of the game.

2. Product/Service Description

The platformer-style game we are developing is an action-packed, adventure game that challenges players to overcome obstacles, defeat enemies, and explore new worlds. The game is designed to provide players with an immersive experience that keeps them engaged and entertained for hours.

The game will include multiple levels, each with its own unique challenges, enemies, and obstacles. The player will control a character who has the ability to run, jump, and attack. The player will face various challenges such as jumping across platforms, dodging obstacles, and defeating enemies. The game will also include power-ups and collectibles that will help the player progress through the game.

In addition to the core gameplay, the game will feature an engaging storyline that takes the player on an adventure through various worlds. The player will encounter a variety of characters, each with their own unique story and personality. The game will feature high-quality graphics and sound effects to enhance the player's experience. The user interface will be intuitive and easy to use, allowing players to quickly navigate through the game and access different features.

The platformer-style game we are developing is a thrilling adventure that combines challenging gameplay with an engaging storyline, high-quality graphics, and an intuitive user interface. It is designed to provide players with a fun and entertaining experience that they will want to come back to again and again.

2.1 Product Context

Market and Industry Landscape: Platformer games are a popular genre, with many successful titles available on various platforms, including consoles, PCs, and mobile devices. Our game must compete with existing titles and meet or exceed their quality and entertainment value to attract players.

Technological Advancements: The gaming industry is continually advancing, and our game must leverage the latest technologies to create a visually stunning and immersive experience for players. This includes the use of game engines, graphics rendering, animation, physics simulations, and other tools to create a polished and engaging game.

Target Audience Expectations: We will conduct market research to gain insight into the preferences and expectations of platformer game players. This research will inform our decisions on game mechanics, art styles, level design, and other features to ensure our game meets the needs of our target audience.

Cross-Platform Compatibility: To reach the broadest audience possible, our game will be designed to run on multiple platforms, including PC and Mac. This will require careful consideration of the technical requirements and limitations of each platform.

Cultural and Social Context: Our game will be released in a global market, and we must consider the cultural and social context in which it will be played. This includes language localization, cultural sensitivities, and accessibility features to ensure all players can enjoy the game regardless of their background or abilities.

2.2 User Characteristics

There is only one user that will interact with the game:

Player:

- ***Can move:*** it has the ability to control the character's movement using the keyboard arrows or some other particular keys which are displayed at the menu page. They are able to move the character left and right, jump and progress through the game.
- ***Can use superpowers:*** The player can use one of the two options to give advantage in game. The first superpower makes the user run faster, while the second one gives the player the possibility to become invisible.
- ***Can customize the settings:*** The player can select one of the powers. It can also customize the background music of tha game and also the volume of the character movement sounds.
- ***Can create and delete profiles:*** The player has the possibility to create a profile in the game. It also has the option to delete a particular profile and add another new profile.
- ***Can pause the game:*** The player has the ability to pause the game at any time, providing them the control and flexibility during gameplay. This feature allows the player to take a break, and then resume the gameplay at their convenience without losing the progress.
- ***Can quit the game:*** The player basically is available to quit the game anytime providing them the option to exit the game in a controlled and safe manner. This feature enables the user to end their gameplay session in a particular level.
- ***Can restart the game:*** This option provides the option to again begin their gameplay session.This feature allows the user to reset their progress and start over, either to improve their previous performance or to experience the game again from the beginning.
- ***User can receive accomplishment:*** The ability to play the game can also allow the user to experience the achievement as they overcome obstacles and reach new milestones as they progress through

different levels. By taking reward the user can better indicate their progress and performance encouraging them to continue playing.

2.3 Assumptions

User:

- It is assumed that the game is a 2D platformer with a main character that can jump,run, interact with the environment.

- It is assumed that the game is designed for a computer platform.

- It is assumed that the user has access to a computer or device that meets the minimum system requirements for running the game.

- It is assumend that the game is free of bugs that could hinder the user's ability to play or complete the game.

- It is assumed that the game will have sound effects and background music to enhance the gaming experience.

- It is assumend that the game will be played using the keyboard inputs for controlling the main character , with arrow keys for movements, or some other characters visible at the main page and the spacebar for jumping.

- It is assumed that each level of the game becomes progressively more challenging, with a gradual increase in difficulty by adding more difficult obstacles.

- It is assumed that the game has different levels with varying degerees of difficulty.

- It is assumed that the game has a scoring system that rewards the player for completing levels and achieving specific objectives.

- It is assumed that the game includes obstacles that the main character needs to overcome to progress through the game.

- It is assummed that the game has two power-ups that can improve the main character's abilities and enhance the gaming experience.

- It is assumed that the game has a well-designed user interface, including the main page, pause and the settings.

- It is assumed that the game engine is optmized for performance to ensure smooth gameplay on the target hardware and software configuration.

- It is assumed that the game's controls and mechanisms are intuitive and easy to understand, allowing the user to quickly learn how to play the game.

2.4 Constraints

- The game should be designed to run on any set of hardware and software configurations, such the operation system, processor and graphics card.

- The game development process should be completed within a specific resource allocation, including hardware, software and human resources.

- The game development process should be completed within a specific timeframe to meet the project deadlines.

- The game should be designed to meet specific design requirements and constraints, such as the game's theme, styling, colors and other design specifications.

- The user interface need to be designed to meet specific design requirements , such as the user interface's appearance, layout and functionality.

- The sound effects should match with the movements of the main character and together with the background music should have the right quality for a better gameplay.

- The game is designed to fit within a certain screen size or aspect ratio, limiting the amount of visual information that can be presented at once.

- The game may be limited by the audio capabilities of the device, which could affect the quality and complexity of the sound effects and music.

2.5. Dependencies

- Pygame library: Pygame is a set of Python modules that enable developers to create games and multimedia applications. It provides access to a range of functionality for creating 2D games, including graphics, sound, input handling, and event handling.

- Python programming language: Python is a high-level, interpreted programming language that is widely used for developing games and other applications. It provides a simple syntax and a wide range of libraries and tools that make it easy to build complex applications.

- Documentation: Documentation is used to generate documentation for the game platformer. It mostly includes description of different classes responsibilities, the methods and attributes it provides, making it easier for developers and users to understand the code and its functionality.

- Graphics and sound assets: The game platformer will require graphics and sound assets, such as images, animations, and sound effects, to create a visually appealing and engaging game.

- Development environment: The game platformer will require a development environment for coding and testing. The development environment chosein is Visual Studio Code.

- Version control system: A version control system is a software tool that tracks changes to code over time, allowing developers to collaborate on a project and keep track of different versions of the code. The version control system choose is Git.

3. Requirements

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date	Reviewed/ Approved
FR_01	The game should have different views for different menus.	The view for main menu, settings and pause menuwill be different.	2	19/04/2023	Emis Reka/ Eva Veli
FR_02	The game should be able to take user input from mouse and keyboard.	To be able to move the character the game should be able to take inputs.	1	19/04/2023	Emis Reka/ Eva Veli
FR_03	The user should be able to explore the in-game world.	The user will be able to move around the world freely.	2	19/04/2023	Emis Reka/ Eva Veli
FR_04	The user should be able to interact to object in the environment.	The game should have interactive objects that the player can use to solve puzzles or progress through the levels.	1	19/04/2023	Emis Reka/ Eva Veli

FR_05	The user should be able interact with the enemies.	The game should include different types of enemies that the player must avoid or defeat.	2	19/04/2023	Emis Reka/ Eva Veli
FR_06	The game should have appropriate sound effects and background music that enhance the player's experience.	The user should be able to control the music volume.	3	19/04/2023	Emis Reka/ Eva Veli
FR_07	The user should be able to pause the game and restart it.	The user will be able to pause the game or restart it in every moment.	2	19/04/2023	Emis Reka/ Eva Veli
FR_08	The game should be able to save the player's progress at specific points in the game.	The will be able to save the game in any point and the game will autosave when level is finished.	1	19/04/2023	Emis Reka/ Eva Veli
FR_09	The user should be able to exit the game in every moment.	The game can be exited without loosing the progress of the player.	2	19/04/2023	Emis Reka/ Eva Veli
FR_10	The user should be able to change the profile name.	Allowing users to change their profile name in a game refers to the ability for them to modify the name displayed in their gaming account or profile.	3	19/04/2023	Emis Reka/ Eva Veli
FR_11	The game should have various power-ups that can be collected by the player.	These power-ups should provide the player with temporary abilities such as invincibility, speed boost, or extra lives.	1	19/04/2023	Emis Reka/ Eva Veli
FR_12	The game should have a scoring system that tracks the player's progress and performance.	The score should be based on the number of enemies defeated, and time taken to complete each level.	1	19/04/2023	Emis Reka/ Eva Veli

FR_13	The game should have different difficulty levels.	The player can choose from various difficulty levels such as easy, medium, and hard.	2	19/04/2023	Emis Reka/ Eva Veli
FR_14	The game should have collectibles items through the levels	The player can find and collect items throughout the levels, such as power-ups and points.	2	19/04/2023	Emis Reka/ Eva Veli
FR_15	The game should have a time limit for each level.	The time limit will be to add an element of urgency and challenge to the gameplay.	3	19/04/2023	Emis Reka/ Eva Veli
FR_16	The game should include accessibility options for players with disabilities.	Every user can change things to be able to play the game, such as adjustable difficulty or alternative control schemes.	3	19/04/2023	Emis Reka/ Eva Veli
FR_17	The game should include a tutorial mode.	The tutorial mode should be able to teach the player how to play the game and use its various mechanics.	3	19/04/2023	Emis Reka/ Eva Veli
FR_18	The game should allow the player to control the camera.	The user can change the camera to get a better view of the level or environment.	3	19/04/2023	Emis Reka/ Eva Veli
FR_19	The game should have achievements or trophies.	The player can get achievements that he can earn by completing specific challenges or objectives.	3	19/04/2023	Emis Reka/ Eva Veli

FR_20	The game will have a lives system.	This will allow the player to earn extra lives by collecting coins or defeating enemies.	2	19/04/2023	Emis Reka/ Eva Veli
FR_21	The game should have platforming challenges.	This will require the player to use their jumping and timing skills to navigate obstacles.	3	19/04/2023	Emis Reka/ Eva Veli
FR_22	The game should have customizable video settings.	Customizable video settings in a game allow players to adjust visual parameters such as resolution, texture quality, anti-aliasing, shadows, and lighting to match their preferences and hardware capabilities.	3	19/04/2023	Emis Reka/ Eva Veli
FR_23	The game should include multiple levels with increasing difficulty.	Each level should be designed to challenge the player's skills and offer a unique experience.	2	19/04/2023	Emis Reka/ Eva Veli
FR_24	The game should be able to allow the user to take a screenshot.	Refers to the ability of the user to capture an image of the current game screen and save it as a picture file for later viewing or sharing	3	19/04/2023	Emis Reka/ Eva Veli
FR_25	The game should allow the user to share the status on Discord.	Display their current gaming activity to their friends, allowing them to see what game they are playing and join in on the action.	3	19/04/2023	Emis Reka/ Eva Veli

3.2 *Non-Functional Requirements*

General Non-Functional Requirements :

Req#	Requirement	Comments	Priority	Date	Reviewed/ Approved
NFR_01	Performance.	The game should be responsive and fast, with no noticeable lag or delay in gameplay.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_02	Scalability.	The game should be scalable to different hardware platforms, screen sizes, and resolutions.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_03	Compatibility.	The game should be compatible with different operating systems, browsers, and devices.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_04	Reliability.	The game should be reliable and stable, with minimal crashes or bugs.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_05	Usability.	The game should be easy to use and navigate, with clear and concise instructions.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_06	Maintainability.	The game should be easy to maintain and update, with clear and organized code structure and documentation.	1	19/04/2023	Emis Reka/ Eva Veli
NFR_07	Compliance.	The game should comply with relevant laws, regulations, and industry standards, such as copyright or data protection laws.	1	19/04/2023	Emis Reka/ Eva Veli

3.2.1 Product Requirements

3.2.1.1 User Interface Requirements

The user interface for the game is a critical aspect that can make or break the player's experience. To ensure the best possible experience, the interface should be easy to use and navigate, providing players with an intuitive way to interact with the game. Furthermore, it should be compatible with different devices and operating systems, allowing players to enjoy the game regardless of the platform they are using. These factors can greatly enhance the overall enjoyment of the game, creating a positive user experience that keeps players engaged and coming back for more.

In addition to outlining the necessary functions, it is important to describe the characteristics of each interface supported by the game's sketches. This provides valuable insight into how the user interacts with the game and how the game responds to user input. By analyzing these characteristics, we can identify areas for improvement and make adjustments to create a more seamless and enjoyable user experience. The main interfaces are:

Main Menu

The Main Menu interface of World Around will be the first screen that the player sees upon launching the game. The screen will have a dark-colored background image that evokes the adventurous theme of the game. At the top of the screen, the game name, "Worlds Around," will be displayed in large and bold white letters using a clear and easily readable font, such as Arcade Classic. The game logo, which may feature elements such as the world map or an adventure-themed icon, will be placed next to the game name on the left-hand side of the screen.

Below the game name and logo, three buttons will be displayed in a row - Play, Settings, and Quit. These buttons will be rectangular in shape with rounded corners and have a slightly raised appearance. The text on each button will be written in a clear and legible font, such as Arcade Classic, and will be presented in a large size to make it easy for players to read. The Play button will have a green color scheme with white text, the Settings button will have a gray color scheme with white text, and the Quit button will have a red color scheme with white text.

Finally, on the far right of the screen, a small square button with an info logo will be displayed, which players can click to access information about the game, such as the developer, game version and the game about. This button will have a light blue color scheme with white text and a transparent background.

After clicking the Play button on the Main Menu interface, the player will be taken to a new screen that displays the profiles available in the game. This screen will feature a white background with a simple and minimalist design that emphasizes the profiles.

The profiles screen will have three rectangular boxes aligned horizontally, each representing a different profile. The profile number (either one, two, or three) will be displayed prominently on the top of each box using a clear and legible font, such as Arcade Classic.

On the bottom of each profile box, there will be a Delete button with a red background, which players can click to delete the corresponding profile. When the Delete button is clicked, a warning message will appear, asking the player to confirm if they really want to delete the profile. The warning message will be displayed in a prominent red color to draw the player's attention and ensure that they do not accidentally delete the profile. There will be two buttons here, yes or no, to confirm it or not.

When a player selects a profile, they will be taken to the last point where they left the game previously or they will continue the game from the beginning if this profile has never been used.

Pause

The pause tab is essential that it is visually appealing, intuitive, and consistent with the main menu and should have a similar background image as the main menu and buttons with similar characteristics. The buttons should be clearly labeled with text and have a border to distinguish them from the background. The font style should be chosen to complement the overall design, and it should be easy to read. A sans-serif font like Arcade Classic is a good choice, as it is simple and legible. The font size should be large enough to be easily readable.

The resume, restart, settings, and quit buttons should be aligned in a logical order and be visually distinguishable. When the player clicks the resume button, the game should immediately resume from where it was paused. Clicking the restart button should reset the game to where he started. The settings button should take the player to the settings tab, and clicking the quit button should bring the player back to the main menu, and the game should be saved where it was left in the corresponding profile.

The color scheme for the pause menu should be consistent with the game's overall aesthetic. The buttons should be a different color from the background to make them stand out, but the contrast shouldn't be too strong to avoid eye strain. A light-colored button with a dark-colored text can work well. The text on the buttons should be descriptive and easy to understand, and the buttons should be large enough to be easy to click.

Finally, when the player clicks on a button, there should be a clear visual indication that the action has been registered. This could be a brief animation, a sound effect, or a color change of the button.

Settings

The Settings tab will have a similar background image to the Main Menu and Pause tabs, providing a consistent visual theme throughout the game. In the middle of the tab, there will be a large text header that reads "Settings" in a clear, easy-to-read font, with a font size that is large enough to be easily visible.

At the top left corner, there will be a back button with an arrow pointing to the left and the text "Back" below it. The button should have a different color from the other buttons on the tab to make it stand out and should be easy to click. When you click it, it will take you to the main menu if you clicked the settings button from the main menu tab or the pause tab if you clicked the settings button from the pause tab.

Below the Settings header, there will be the control buttons for the game. There will be four texts on the left side, one below the other, which read "Enter Door", "Left", "Down", and "Right". On the right side of each of these four texts, there will be four buttons that are "W", "A", "S" and "D" respectively, corresponding to the four texts on their left.

At the right side of these four buttons, there will be three texts one below the other, that are "Jump", "Power1", and "Power2". On the right of each of these three texts, there will be three buttons that are "Space", "1", and "2" respectively, corresponding to the three texts on their left. The buttons should have a clear, easy-to-read font and a font size that is large enough to be easily visible.

When the user clicks on these buttons, they can change the controls to their preferences and the new control scheme will be saved. Below these texts and buttons in the middle, there will be a reset button that the user can click if they want to reset the controls to the default.

At the end of the tab, there will be two small sound images, one for the background music and the other for the game sound. These images will be one below each other and will have two straight lines and a point for each line that will be on these lines. This will show how much the volume is for the background music and the game sound. If it is at the beginning of the line, it tells that there will be no sound or music for the game, and there will be a red cross on the image to show that this sound is muted.

The colors of the buttons and text should be chosen to provide good contrast against the background image and should be visually appealing. The buttons should be large enough to be easily clicked and the text should be easy to read.

3.2.1.2 Learnability

If you are working on a platformer game, here are some learnability requirements to consider:

- **User Interface Design:** The user interface of the game should be designed to be intuitive and easy to navigate. All the game mechanics, menus, and settings should be accessible through the interface.
- **Consistency:** Consistency is key when it comes to learnability. The game mechanics, controls, and visual feedback should be consistent throughout the game. The player should be able to predict the outcome of their actions based on their experience from previous levels.
- **Onboarding:** The game should have a clear and concise onboarding process that introduces the player to the game mechanics and controls. It is best to introduce new mechanics one at a time, rather than overwhelming the player with too much information at once.
- **Feedback:** The game should provide clear and immediate feedback to the player. This can be done through visual or audio cues. The player should know what action they need to take and understand the consequences of their actions.
- **Difficulty Curve:** The game should have a gradual difficulty curve that increases as the player progresses through the levels. It is important to balance the challenge with the player's skill level to keep them engaged and motivated.
- **Playtesting:** Playtesting is an essential part of the game development process. You should regularly test the game with real users to get feedback on the game mechanics, controls, and difficulty level. This feedback can be used to make adjustments to the game to improve its learnability.

3.2.1.3 Accessibility

When designing a platformer game , it is essential to consider accessibility requirements to ensure that the game is playable by a wider audience. Here are some accessibility requirements to consider:

- **Input Options:** Provide different input options for players, such as keyboard, mouse and gamepad. Ensure that the game can be played with a single hand, as some players may have a disability that affects one hand.
- **Customizable Controls:** Allow players to customize the game controls to their liking. This feature is essential for players with disabilities that may not be able to use the default controls.
- **Text Size and Font:** Provide an option to adjust the text size and font of the game. This feature is important for players with visual impairments.
- **Colorblind Mode**: Provide a colorblind mode that adjusts the colors of the game to make it easier for colorblind players to distinguish between different objects.
- **Sound Options:** Provide different sound options, such as subtitles or closed captions for players who are deaf or hard of hearing.
- **Difficulty Options:** Provide different difficulty options, such as easy or hard mode, to accommodate players with varying skill levels.
- **Clear Visual Feedback:** Ensure that all the game mechanics, controls, and visual feedback are easy to understand and distinguish.

- **Avoid Time-Based Challenges:** Avoid time-based challenges that may be difficult for players with disabilities that affect their reaction times.

3.2.1.4 Efficiency

Efficiency requirements for a platformer game are related to the performance of the game, its load times, and its ability to run smoothly on different hardware configurations. Here are some efficiency requirements to consider:

- **Optimization:** Optimize the game code and assets to reduce load times and improve performance. Use efficient algorithms, data structures, and rendering techniques to minimize the CPU and GPU usage.
- **Level Design:** Design the levels with efficiency in mind, avoiding large areas or overly complex structures that may affect the game's performance. Use level streaming or loading techniques to keep the memory usage under control.
- **Asset Management:** Manage the game assets efficiently, using compressed formats and reducing the number of textures or models to improve load times and reduce memory usage.
- **Platform Support:** Ensure that the game can run smoothly on different hardware configurations, including low-end and high-end systems. Optimize the game for different platforms, such as PC, console, or mobile, to ensure good performance on each platform.
- **Memory Management:** Implement efficient memory management techniques, such as object pooling or garbage collection, to reduce memory fragmentation and improve performance.
- **Testing:** Test the game on different hardware configurations and use profiling tools to identify performance bottlenecks. Address any performance issues identified during testing to improve the game's efficiency.

3.2.1.5 Memorability

Memorability requirements for a platformer game are related to the ability of the player to remember how to play the game after a break or when returning to the game after some time. Here are some memorability requirements to consider:

- **Tutorial and Onboarding:** Include a tutorial and onboarding process that is easy to understand and remember. This will help the player remember the game mechanics and controls when returning to the game after a break.
- **Consistency:** Ensure that the game mechanics, controls, and visual feedback are consistent throughout the game. This will make it easier for the player to remember how to play the game when returning to it.
- **Clear Objective:** Provide a clear objective or goal for the player to remember. This will help the player understand what they need to do when returning to the game after a break.
- **Save Progress:** Include a save progress feature that allows the player to pick up where they left off. This will help the player remember where they were in the game and what they need to do next.
- **Reminders:** Include reminders or hints that help the player remember how to play the game. This could be in the form of pop-ups or visual cues that highlight important game mechanics or controls.
- **Feedback:** Provide clear and immediate feedback to the player. This will help the player remember the consequences of their actions and what they need to do next.

3.2.1.6 Errors

⍰ The error rate is lower than the current error rate.

⍰ If an error occurs it can be edited and corrected immediately.

3.2.1.7 Satisfaction

Satisfaction requirements for a platformer game are related to the player's emotional response to the game and their overall satisfaction with the game experience. Here are some satisfaction requirements to consider:

- **Engaging Gameplay:** Provide engaging gameplay that is challenging but not frustrating. This will keep the player motivated and interested in playing the game.
- **Interesting Storyline:** Include an interesting storyline or narrative that captures the player's attention and provides motivation to progress through the game.
- **Reward System:** Implement a reward system that incentivizes the player to complete objectives and progress through the game. This could include unlocking new levels, power-ups, or cosmetic items.
- **Replayability:** Provide replayability features, such as different difficulty modes or randomized levels, to encourage the player to play the game again and again.
- **Visual Appeal:** Create visually appealing graphics and animations that enhance the player's immersion in the game world.
- **Audio Design:** Create an immersive audio design that includes sound effects and music that enhances the player's emotional response to the game.
- **Accessibility:** Ensure that the game is accessible to a wide range of players, including those with disabilities, to provide a more inclusive and satisfying gaming experience.

3.2.1.8 Capacity

Capacity requirements for a platformer game are related to the storage and processing resources required to develop and run the game. Here are some capacity requirements to consider:

- **Storage Capacity:** Estimate the amount of storage capacity required for game assets such as levels, characters, textures, sound effects, and music. Ensure that there is sufficient storage capacity available for the development and distribution of the game.
- **Processing Power**: Estimate the amount of processing power required to render the game graphics and animations, calculate physics, and handle game logic. Ensure that there is sufficient processing power available for the game to run smoothly and efficiently.
- **Cloud Infrastructure:** Consider using cloud infrastructure services such as Amazon Web Services or Microsoft Azure to scale up the capacity of the game. This can help reduce costs and improve scalability.
- **Hardware Compatibility:** Ensure that the game is compatible with a wide range of hardware configurations, including low-end and high-end systems. Optimize the game for different platforms, such as PC, console to ensure good performance on each platform.

3.2.2 Organizational Requirements

3.2.2.1 Availability

Availability requirements for a platformer game are related to ensuring that the game is available and accessible to players at all times. Here are some availability requirements to consider**:**

- **Uptime:** Ensure that the game is available to players for the majority of the time. This means minimizing server downtime and addressing any technical issues that could impact the game's availability.
- **Accessibility:** Make the game accessible to a wide range of players, including those with disabilities. Ensure that the game is compatible with assistive technologies and that it meets accessibility guidelines.
- **Responsiveness:** Ensure that the game responds quickly to player input and provides a smooth and lag-free gameplay experience. This means optimizing game code, minimizing network latency, and using efficient server infrastructure.
- **Scalability:** Design the game architecture to be scalable, so that it can handle increases in player traffic and usage without impacting availability or performance.
- **Disaster Recovery:** Have a disaster recovery plan in place in case of unexpected events that could impact the availability of the game, such as server crashes or natural disasters.

3.2.2.2 Latency

Latency requirements for a platformer game are related to ensuring that the game responds quickly and smoothly to player inputs, providing a seamless and enjoyable gameplay experience. Here are some latency requirements to consider:

- **Input Responsiveness:** Ensure that the game responds quickly and accurately to the player's input, with minimal delay or lag. This is particularly important for a platformer game where precise timing and movement are crucial.
- **Game Engine Optimization:** Optimize the game engine to reduce processing time and minimize input lag. This can be achieved by using efficient algorithms, reducing unnecessary computations, and optimizing the game engine's architecture.
- **Hardware Optimization:** Optimize the game for different hardware configurations to minimize input lag. This can be achieved by optimizing the game engine for different CPU and GPU architectures, and testing the game on a wide range of hardware configurations.
- **Player Feedback:** Provide immediate and clear feedback to the player when they perform an action. This can include visual and audio cues, such as animation or sound effects, to indicate that the game has registered the player's input.

3.2.2.3 Monitoring

Monitoring requirements for a platformer game project are related to the measurement and management of various aspects of the game during development and after release. Here are some monitoring requirements to consider:

- **Performance Monitoring:** Monitor the performance of the game, including frame rate, loading times, and resource utilization, to identify and diagnose performance issues.
- **Bug Monitoring:** Implement a bug tracking system to monitor and track reported bugs and issues. This

can help prioritize and manage bug fixes.

- **User Behavior Monitoring:** Monitor user behavior, including playtime, progression, and retention, to understand how users interact with the game and identify areas for improvement.
- **Server Monitoring:** Monitor game servers to ensure that they are running smoothly and efficiently, and to identify and diagnose issues that may impact gameplay.
- **Security Monitoring:** Implement security monitoring to detect and prevent cheating, hacking, and other security issues that may impact the integrity of the game.
- **Analytics:** Implement analytics tools to measure and analyze user behavior, engagement, and other metrics that can inform game development and optimization.

3.2.2.4 Maintenance

Game maintenance is an important aspect of game development that ensures the game remains functional and enjoyable for players after its release. It involves ongoing updates and fixes to address bugs, glitches, and other issues that may arise.

For this platformer-style game, maintenance will be crucial in ensuring the game remains playable and enjoyable for players. This includes:

- **Bug fixes:** Game developers must monitor the game closely to identify and fix any bugs that may arise. These can include issues with player movement, collisions, enemy behavior, and other gameplay elements.
- **Balancing updates:** As players become more experienced with the game, they may identify areas where the difficulty needs to be adjusted or certain elements need to be rebalanced. Developers will need to monitor player feedback and make necessary adjustments to ensure the game is challenging, but not frustrating.
- **Content updates:** To keep players engaged and interested in the game, new content such as levels, enemies, and power-ups can be added over time. This can help to keep the game fresh and encourage players to continue playing.
- **Performance optimization:** As new hardware and operating systems are released, game developers must ensure that the game remains compatible and runs smoothly on these new platforms. This may require updates to the game's code or optimizations to improve performance.

3.2.2.5 Operations

- **Network and server infrastructure:** Games that involve multiplayer gameplay typically require a robust network and server infrastructure to handle the traffic and data generated by users. Developers need to ensure that the game's backend systems are designed to be scalable, redundant, and secure. This means that they need to consider things like load balancing, failover mechanisms, network topology, and security protocols when designing the infrastructure. They also need to ensure that the infrastructure can handle high volumes of traffic and that it is able to provide a consistent user experience across different devices and platforms.
- **Game deployment and management:** Developers need to ensure that the game is deployed and managed in a way that ensures maximum uptime and availability. This means that they need to consider things like continuous integration and deployment, monitoring and alerting, and incident response procedures. They also need to ensure that they have appropriate backup and disaster recovery plans in place to minimize the impact of any disruptions or outages.
- **Player support and engagement:** To ensure a positive player experience, developers need to

provide appropriate support and engagement mechanisms. This includes things like providing clear and comprehensive documentation, offering responsive customer support, and engaging with players through social media and other channels. They also need to ensure that they have appropriate mechanisms in place to address player feedback and concerns.

3.2.2.6 Standards Compliance

- **Payment processing:** Many games rely on in-app purchases and microtransactions to generate revenue. To ensure compliance with relevant regulations and standards, developers must implement secure payment processing systems that protect user data and adhere to industry standards such as the Payment Card Industry Data Security Standard (PCI DSS). This includes things like encrypting sensitive data, using secure payment gateways, and implementing fraud detection mechanisms.
- **Data privacy:** As games collect and process user data, developers must ensure that they comply with relevant data privacy regulations such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA). This means that they need to implement appropriate data protection measures, such as encryption, access controls, and data retention policies. They also need to provide users with clear and transparent information about what data they collect and how they use it, and obtain user consent where necessary.
- **Accessibility:** Games should be designed to be accessible to all users, including those with disabilities. Developers need to ensure that they comply with accessibility standards such as the Web Content Accessibility Guidelines (WCAG) and the Americans with Disabilities Act (ADA). This means that they need to consider things like color contrast, font size, and support for assistive technologies such as screen readers and keyboard navigation.
- **Content moderation:** As games allow users to interact with each other, developers must ensure that they comply with relevant content moderation regulations and standards. This includes things like implementing reporting mechanisms, establishing community guidelines, and enforcing appropriate sanctions for violations of those guidelines. They also need to ensure that they comply with relevant laws and regulations regarding hate speech, harassment, and other harmful behaviors.

3.2.2.7 Portability

Game portability refers to the ability of the game to be played on different platforms and operating systems without any significant changes to the codebase. This is an important aspect of game development, as it allows the game to reach a wider audience and ensures that players can enjoy the game regardless of the device or system they are using.

For this platformer-style game, we aim to ensure high portability across multiple platforms, including desktops, laptops, and mobile devices running on Windows, macOS, and Linux operating systems.

To achieve this, we will use game development frameworks and engines such as Unity or Unreal Engine, which provide cross-platform support and allow developers to write code once and deploy to multiple platforms with minimal effort. We will also ensure that the game's assets (such as images, sound effects, and music) are optimized for different resolutions and aspect ratios to ensure the game looks and feels consistent across different platforms.

Moreover, we will make sure that the game's user interface is designed with portability in mind. This means that the game's menus, buttons, and other user interface elements will be designed to work well on different screen sizes and resolutions. Additionally, we will ensure that the game's control scheme is designed to work well with different input devices, such as keyboards, mice, gamepads, and touchscreens. By prioritizing portability, we aim to make the game accessible to as many players as possible, regardless of their device or operating system.

3.2.3 External Requirements

3.2.3.1 Security

Security is an important aspect of game development, and it is essential to ensure that the game is secure and does not pose any risks to the players. The following measures will be taken to ensure the security of the game:

- **Data encryption:** All sensitive data such as user credentials, payment information, and other personal information will be encrypted to prevent unauthorized access.
- **Access controls:** Access to the game servers and databases will be restricted only to authorized personnel to prevent data breaches and unauthorized access.
- **Anti-cheat mechanisms:** The game will include anti-cheat mechanisms to prevent cheating, hacking, and other malicious activities that may compromise the security of the game.

3.2.3.2 Protection

- **Code obfuscation:** The game's code will be obfuscated to make it harder for hackers to reverse engineer and steal intellectual property or manipulate game data.
- **Encryption:** Game data, including user data and game files, will be encrypted to prevent unauthorized access.
- **Anti-cheat measures:** The game will implement anti-cheat measures to detect and prevent cheating and unfair gameplay, such as using third-party software or exploiting bugs.
- **User account security:** User accounts will be secured with strong passwords, two-factor authentication, and other security measures to prevent unauthorized access and protect user data.

3.2.3.3 Authorization and Authentication

Authentication is the process of verifying a user's identity. In games that involve user accounts, developers need to implement secure authentication mechanisms to prevent unauthorized access to user accounts and protect sensitive user data. This includes things like implementing strong password requirements, two-factor authentication, and session management to prevent session hijacking.

Authorization is the process of granting or denying access to specific resources or functionality based on a user's identity and permissions. In games that involve online multiplayer functionality or social features, developers need to implement authorization mechanisms to control what actions users can perform in the game, what data they can access, and who they can interact with. This includes things like implementing role-based access controls, establishing user permissions, and implementing appropriate validation and error handling mechanisms to prevent unauthorized actions.

Social authentication allows users to log in to the game using their social media accounts, such as Facebook or Google. Developers need to ensure that their social authentication mechanisms are secure and comply with relevant regulations and standards, such as OAuth and OpenID Connect. This includes things like implementing secure data exchange between the game and the social media platform, handling user consent and permissions, and implementing appropriate error handling and logging mechanisms.

3.3 Domain requirements

Domain requirements for a platformer game refer to the specific features, mechanics, and content that are required to create a game that fits within the platformer genre. Here are some domain requirements to consider:

- **Character Movement:** Platformer games require precise and responsive character movement mechanics, such as jumping, running, and sliding, to navigate through levels and avoid obstacles.
- **Level Design:** Platformer games require well-designed levels that provide interesting challenges for players to overcome, such as jumping puzzles, enemy encounters, and environmental hazards.
- **Collectibles:** Platformer games often feature collectibles, such as coins, power-ups, or other rewards, which incentivize exploration and mastery of levels.
- **Enemies and Bosses:** Platformer games often include enemies and bosses that players must defeat to progress through levels and the game.
- **Platform Types:** Platformer games feature different types of platforms that the player must navigate, such as moving platforms, disappearing platforms, and platforms that crumble or break.
- **Power-Ups:** Platformer games often feature power-ups that grant the player temporary abilities or enhancements, such as invincibility or increased speed.
- **Music and Sound Effects:** Platformer games often feature catchy and upbeat music, as well as sound effects that enhance the player's experience and create a sense of immersion.

4 Software Design / Diagrams

4.1 Requirements Analysis

4.1.1 User Scenarios

4.1.1.1 User Scenarios List

Nr	Name	Description
US_01	User opens the game	User presses the game button and the game opens at the Main Menu.
US_02	User presses the play button	When the play button is pressed, the profile view is shown.
US_03	User presses the settings button	When the settings button is pressed, the game shows some settings that can be configured.
US_04	User presses the exit button	When the quit button is pressed, the game closes.
US_05	Create a profile	The user can create a profile and start playing the game from there.
US_06	Choose a profile	The user can choose an existing profile and continue the game where they left off.
US_07	Delete a profile	The user can choose to delete an existing profile and is asked to confirm it when the delete button is pressed.
US_08	Back button	When the back button is pressed, the user is returned to the previous view.
US_09	Enter door key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_10	Left key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_11	Down key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_12	Right key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_13	Jump key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_14	Power1 key clicked	When the key is pressed, the user can choose to remap this action to the key they press.

US_15	Power2 key clicked	When the key is pressed, the user can choose to remap this action to the key they press.
US_16	User presses the reset button	When the reset button is pressed, the controls are set to their default values.
US_17	Game music slider is moved	When the user moves the music slider, the volume of the game's music changes accordingly.
US_18	Background music slider is moved	When the user moves the background slider, the volume of the background's music changes accordingly.
US_19	User presses the resume button	When the resume button is pressed, the game continues where it was paused.
US_20	User presses the restart button	When the restart button is pressed, the game starts from the beginning of the level.
US_21	User presses the quit button at pause menu	When the quit button is pressed, the user is returned to the main menu.
US_22	Enter door key pressed	When the key is pressed and the user is on a door, the user the door and moves to the other side.
US_23	Left key pressed	When the key is pressed, the player moves to the left.
US_24	Down key pressed	When the key is pressed, the player crouches.
US_25	Right key pressed	When the key is pressed, the player moves to the right.
US_26	Jump key pressed	When the key is pressed, the player jumps.
US_27	Power1 key pressed	When the key is pressed, the player uses the first power up.
US_28	Power2 key pressed	When the key is pressed, the player uses the second power up.
US_29	Level finished	When the level is finished, the player continues the next level.

4.1.1.2 User Scenarios Extended

US1: User opens the game

a) User launches the game application on their device.
b) The game software initializes and displays the Main Menu screen.
c) The Main Menu screen showcases the game's title, logo, and captivating artwork.
d) The Main Menu provides an overview of the available game features, options, and modes.
e) The user interacts with the Main Menu using their input device.
f) The user selects the option to start the game.
g) The game progresses to the next stage, which may involve loading assets or presenting profile selection options.
h) The necessary preparations are completed, and the user enters the game world or a specific gameplay mode.
i) The user can now engage in the interactive experience the game offers, such as exploring levels, challenges, or storylines.

US2: User presses the play button

a) User is at the Main Menu screen of the game.
b) The Main Menu displays various options, including the play button.
c) User visually locates the play button on the screen.
d) User uses their input device to interact with the play button.
e) User clicks or taps the play button.
f) The game processes the user's input and initiates the gameplay sequence.
g) The game transitions to the appropriate gameplay mode or level.
h) User is now actively engaged in playing the game.

US3: User presses the settings button

a) User is at the Main Menu or Pause screen of the game.
b) The screen displays various options, including the settings button.
c) User visually locates the settings button on the screen.
d) User uses their input device to interact with the settings button.
e) User clicks or taps the settings button.
f) The game processes the user's input and opens the settings menu.
g) The settings menu displays different options for configuring game settings.
h) User can adjust various settings such as sound and controls preferences.
i) User modifies the desired settings by interacting with the corresponding UI elements.

j) The game saves the changes made by the user.
k) User can navigate within the settings menu and make further modifications if needed.
l) User can access additional submenus or screens within the settings menu to adjust more specific settings.
m) User can return to the previous screen (Main Menu or Pause) by pressing the back button or a designated UI element.
n) The game processes the user's input and returns to the previous screen, with the updated settings in effect.

US4: User presses the exit button

a) User is currently at the Main Menu screen.
b) The screen displays various options, including the exit button.
c) User visually locates the exit button on the screen.
d) User uses their input device to interact with the exit button.
e) User clicks or taps the exit button.
f) The game processes the user's input and initiates the exit process.
g) The game saves any necessary data or progress.
h) The game closes gracefully, returning the user to the operating system or a designated exit screen.

US5: Create a profile

a) User is at the profile selection screen of the game.
b) The screen provides an option to create a new profile.
c) User visually locates the "Create Profile" button or a similar option.
d) User uses their input device to interact with the "Create Profile" button.
e) The game prompts the user to enter a name or identifier for the new profile.
f) User enters the desired name.
g) The game validates the entered name to ensure it meets any required criteria (e.g., length, uniqueness).
h) If the name is valid, the game creates a new profile with the entered name.
i) The game saves the newly created profile and associated data.
j) The game confirms the successful creation of the profile to the user.
k) The user is returned to the profile selection screen, with the new profile now available for selection.

US6: Choose a profile

a) User is at the profile selection screen of the game.

b) The screen displays a list of existing profiles.
c) User visually scans the list to locate the desired profile.
d) User selects the desired profile by clicking on it or using a navigation input to highlight and confirm the selection.
e) The game loads the selected profile and associated data.
f) The user is directed to the game interface where they can continue playing from where they left off in the selected profile.

US7: Delete a profile

a) User is at the profile selection screen the game.
b) The screen displays a list of existing profiles.
c) User visually scans the list to locate the profile they want to delete.
d) User selects the profile they want to delete by clicking on it or using a navigation to highlight it.
e) User clicks on the "Delete" button or selects a delete option from a context menu.
f) The game prompts a confirmation message asking the user if they are sure they want to delete the selected profile.
g) User confirms the deletion by clicking on the "Yes" option.
h) The game removes the selected profile and associated data from the system.
i) The profile list is updated, and the deleted profile is no longer displayed.

US8: Back button

a) User is in a specific view or menu within the game.
b) User wants to navigate back to the previous view or menu.
c) User locates and clicks on the "Back" button, typically displayed in the top-left or bottom-left corner of the screen.
d) The game processes the user input and performs the appropriate action.
e) The game transitions the user back to the previous view or menu, restoring the previous state.

US9: Enter door key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Enter Door" action.
c) User locates and clicks on the "Enter Door" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Enter Door" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Enter Door" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US10: Left key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Left" action.
c) User locates and clicks on the "Left" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Left" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Left" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US11: Down key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Down" action.
c) User locates and clicks on the "Down" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Down" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Down" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US12: Right key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Right" action.
c) User locates and clicks on the "Right" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Right" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Right" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US13: Jump key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Jump" action.
c) User locates and clicks on the "Jump" key mapping option.

d) The game interface prompts the user to input a new key or combination for the "Jump" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Jump" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US14: Power1 key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Power1" action.
c) User locates and clicks on the "Power1" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Power1" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Power1" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US15: Power2 key clicked

a) User is in the Settings menu of the game, specifically in the controls section.
b) User wants to customize the key mapping for the "Power2" action.
c) User locates and clicks on the "Power2" key mapping option.
d) The game interface prompts the user to input a new key or combination for the "Power2" action.
e) User presses the desired key on the keyboard or selects a key from a dropdown menu.
f) The game registers the user's input and updates the key mapping for the "Power2" action.
g) The new key mapping is saved in the game's settings and will be used for future gameplay.
h) User can navigate back to the main Settings menu or continue customizing other key mappings.

US16: User presses the reset button

a) User is in the Settings menu of the game.
b) User wants to reset the settings to their default values.
c) User locates and clicks on the "Reset" button.
d) The game resets all settings to their default values, including controls, sound settings, and any other configurable options.
e) The game interface updates to reflect the default settings.
f) User can navigate back to the main Settings menu or continue adjusting other settings.

US17: Game music slider is moved

a) User is in the Settings menu of the game.
b) User wants to adjust the volume of the game's music.
c) User locates the "Game Music" slider control.
d) User clicks and drags the slider control to the desired position, indicating the desired volume level for the game's music.
e) The game interface dynamically adjusts the volume of the game's music based on the position of the slider.
f) User can hear the changes in the music volume in real-time.
g) The position of the slider visually represents the current volume level.
h) User can navigate back to the main Settings menu or continue adjusting other settings.

US18: Background music slider is moved

a) User is in the Settings menu of the game.
b) User wants to adjust the volume of the background's music.
c) User locates the "Background Music" slider control.
d) User clicks and drags the slider control to the desired position, indicating the desired volume level for the game's music.
e) The game interface dynamically adjusts the volume of the game's music based on the position of the slider.
f) User can hear the changes in the music volume in real-time.
g) The position of the slider visually represents the current volume level.
h) User can navigate back to the main Settings menu or continue adjusting other settings.

US19: User presses the resume button

a) User is playing the game and decides to pause the gameplay.
b) User clicks on the pause button, and the game transitions to the pause menu.
c) User sees various options in the pause menu, including the "Resume" button.
d) User selects the "Resume" button by clicking on it.
e) The game resumes from the exact point where the user paused, with all progress and game state intact.
f) User continues playing the game from where they left off.
g) The pause menu disappears, and the game interface is restored to its regular gameplay view.
h) User can pause the game again in the future if needed or complete the level/game session.

US20: User presses the restart button

a) User is playing the game and wants to restart the current level or the entire game.
b) User clicks on the pause button, and the game transitions to the pause menu.
c) User sees various options in the pause menu, including the "Restart" button.
d) User selects the "Restart" button by clicking on it.
e) The current level or the entire game is reset to its initial state.
f) User is brought back to the starting point of the level or the main menu of the game.
g) User can start playing the level from the beginning or select a different level to play.

US21: User presses the quit button at pause menu

a) User is playing the game and wants to quit the current session and return to the main menu.
b) User clicks on the pause button, and the game transitions to the pause menu.
c) User sees various options in the pause menu, including the "Quit" button.
d) User selects the "Quit" button by clicking on it.
e) The current session is ended, and the game returns to the main menu.
f) User can choose to start a new game, load a different profile, or exit the game.

US22: Enter door key pressed

a) User is playing the game and controlling the character within the game environment.
b) User's character approaches a door or an entrance point within the game.
c) User wants to enter the door or proceed through the entrance.
d) User locates and clicks the designated "Enter Door" key, which triggers the action.
e) The game recognizes the user input and performs the corresponding action.
f) The character's animation changes to indicate the door interaction, such as opening or transitioning to a new area.
g) The game transitions the user to the new game area associated with the door.
h) The user continues playing the game from the new game area.

US23: Left key pressed

a) User is actively playing the game and wants to move the character to the left.
b) User presses the "Left" key on the keyboard.
c) The game detects the input and initiates the left movement action for the character.
d) The character starts moving towards the left direction.
e) The game environment updates accordingly to reflect the character's movement.
f) If there are obstacles or platforms in the character's path, the character interacts with them as intended.
g) The character continues moving to the left until the user releases the "Left" key or takes a different action.

h) The game responds to any subsequent input or events based on the new character position and state.

US24: Down key pressed

a) User is actively playing the game and wants the character to perform a downward action or interact with an object below.
b) User presses the "Down" key on the keyboard.
c) The game detects the input and triggers the corresponding action associated with the "Down" key.
d) Depending on the game mechanics, the character performs an action such as crouching.
e) The game environment updates to reflect the character's action, such as the character's sprite changing to a crouching position.
f) The character remains in the downward action or interaction until the user releases the "Down" key or performs a different action.
g) The game responds to any subsequent input or events based on the current character position and state.

US25: Right key pressed

a) User is actively playing the game and wants to move the character to the left.
b) User presses the "Right" key on the keyboard.
c) The game detects the input and initiates the left movement action for the character.
d) The character starts moving towards the left direction.
e) The game environment updates accordingly to reflect the character's movement.
f) If there are obstacles or platforms in the character's path, the character interacts with them as intended.
g) The character continues moving to the right until the user releases the "Right" key or takes a different action.
h) The game responds to any subsequent input or events based on the new character position and state.

US26: Jump key pressed

a) User is actively playing the game and wants the character to perform a jump.
b) User presses the "Jump" key on the keyboard.
c) The game detects the input and triggers the corresponding action associated with the "Jump" key.
d) The character performs a jump animation and moves vertically upwards.
e) The game environment updates to reflect the character's jump, such as adjusting the character's position, applying gravity, and animating the jump sequence.
f) If there are platforms or obstacles in the character's path, the game may handle collisions or interactions accordingly.
g) The character reaches the peak of the jump and starts descending back down.

h) The character lands on the ground or any platform, and the game adjusts the character's position accordingly.
i) The character continues with the previous action or performs other actions based on the user's subsequent input.

US27: Power1 key pressed

a) User is actively playing the game and encounters a situation where using Power1 is possible.
b) User decides to activate Power1 by pressing the designated key.
c) The game detects the key press event for Power1.
d) The game checks if the user has sufficient resources or requirements to activate Power1.
e) If the user meets the requirements, the game triggers the Power1 effect.
f) The Power1 effect is applied, which can vary depending on the game mechanics. It is a special ability that affects gameplay.
g) The game updates the visual representation and feedback to reflect the activation of Power1.
h) The game has a cooldown period associated with using Power1, depending on the game's design.
i) The user continues playing the game with the effects of Power1.
j) After a certain duration or when the effect is exhausted, the Power1 effect ends.
k) The game reverts to the normal gameplay state without Power1's influence.

US28: Power2 key pressed

a) User is actively playing the game and encounters a situation where using Power1 is possible.
b) User decides to activate Power2 by pressing the designated key.
c) The game detects the key press event for Power2.
d) The game checks if the user has sufficient resources or requirements to activate Power2.
e) If the user meets the requirements, the game triggers the Power2 effect.
f) The Power2 effect is applied, which can vary depending on the game mechanics. It is a special ability that affects gameplay.
g) The game updates the visual representation and feedback to reflect the activation of Power2.
h) The game has a cooldown period associated with using Power2, depending on the game's design.
i) The user continues playing the game with the effects of Power2.
j) After a certain duration or when the effect is exhausted, the Power2 effect ends.
k) The game reverts to the normal gameplay state without Power2's influence.

US29: Level finished

a) User is actively playing a level in the game.
b) User successfully completes all objectives or reaches the end point of the level.
c) The game detects that the level has been finished.

d) The game displays a level completion screen or animation to provide feedback and celebrate the achievement.
e) The user may be awarded points, stars, or other rewards based on their performance during the level.
f) The game transitions to a level summary screen or progresses to the next level.
g) The user may be provided with a summary of their performance, such as the time taken, collected items, or any optional objectives completed.
h) The user is given the option to proceed to the next level, replay the current level, or return to the main menu.
i) If the user chooses to proceed to the next level, the game loads the next level and the user can continue playing.
j) If the user chooses to replay the current level, the game restarts the level from the beginning, allowing the user to improve their performance.
k) If the user chooses to return to the main menu, the game navigates back to the main menu screen.

4.1.2 User Cases

Name	User opens the game
Summary	User click on the game icon to open the game.
Actor	User
Description	After clicking the icon , the main menu of the game will be displayed to the user.
Precondition	-The game has been installed on the user's device -The device is turned on and has power -The user has navigated to the game's icon and clicked on it
Alternatives	-User Cancels Game Launch -User Adjusts Game Settings -User Views Game Instructions -User Chooses Different Level -User Selects Different Game Mode
Post Condition	-The user is successfully playing the game. -The user has access to game controls, settings, and instructions as needed.

UC_01 – US_01 -User Opens the Game

Name	User presses the play button
Summary	User can start playing the game .
Actor	User
Description	The user selects a level, clicks on the "Play" button, and begins playing the selected level.

Precondition	-The game has been opened and is in the main menu screen -The user has selected a level to play from the available options -The user has clicked on the "Play" button
Alternatives	-User Adjusts Game Settings -User Views Level Instructions -User Adds a New Profile -User Deletes a Current Profile -User Pauses or Quits Level
Post Condition	-The user is successfully playing the selected level in the game. -The user has access to game controls and settings as needed.

UC_02 – US_02 -User Presses the Play Button

Name	User presses the settings button
Summary	User can open the game settings menu where they can adjust various game options.
Actor	User
Description	Enable users to customize their gameplay experience by accessing and adjusting game settings as desired, ultimately enhancing user satisfaction and engagement.
Precondition	-The game is running and accessible. -The user is on the game's main menu or pause menu.
Alternatives	- User Cancels Settings Changes -User Resets Default Settings -User Views Game Controls -User Accesses Help or Support -User Exits the Game
Post Condition	- The user has successfully accessed and adjusted the game's settings as desired.

UC_03 – US_03 -User Presses the Settings Button

Name	User presses the exit button

Summary	User can exit the game.
Actor	User
Description	Allows the user to exit the game and return to the device's home screen or another application.
Precondition	-The game is running and accessible. -The user is on the game's main menu or pause menu.
Alternatives	-User Cancels Game Exit -User Switches to Another Application -User Quits Without Confirmation
Post Condition	- The user has successfully exited the game and returned to the device's home screen or another application.

UC_04 – US_04 -User Presses the Exit Button

Name	Creates a new profile.
Summary	User can create up to three new profiles.
Actor	User
Description	Allows the user to create a new player profile to store and track their progress and achievements within the game.
Precondition	-The game is running and accessible. -The user is on the game's main menu or pause menu.
Alternatives	-User Cancels Profile Creation -User Edits Profile Name -User Deletes Profile -User Selects Existing Profile
Post Condition	- The user has successfully created a new player profile and can begin playing the game with the new profile.

UC_05 – US_05 -Create a New Profile

Name	Choose profile.
Summary	User can choose one out of three profiles.
Actor	User
Description	Allows the user to select an existing player profile and resume playing the game where they left off.
Precondition	-The game is running and accessible. -The user is on the game's main menu. -The user has created at least one player profile.
Alternatives	-User Cancels Profile Selection -User Edits Profile Name -User Deletes Profile -User Creates New Profile
Post Condition	- The user has successfully chosen a player profile and can resume playing the game where they left off with the selected profile.

UC_06 – US_06 -Choose a Profile

Name	Delete a profile
Summary	User can delete one or all its profiles.
Actor	User
Description	Allows the user to delete an existing player profile.
Precondition	-The game is running and accessible. -The user is on the game's main menu. -The user has created at least one player profile.
Alternatives	-User Cancels Profile Deletion - If the user deletes their only player profile, the game should automatically create a new empty profile for them to use.
Post Condition	- The user has successfully deleted the selected player profile.

UC_07– US_07 - Delete a Profile

Name	Back Button

Summary	User can press the back button.
Actor	User
Description	Allows the user to go back to the previous screen or menu.
Precondition	-The game is running and accessible. -The user is on a screen or menu that has a back button.
Alternatives	-If the user is on the first screen or menu of the game, the back button may not be available or may be grayed out. -If the user is navigating through a multi-level menu, each click of the back button should take the user back one level at a time until they reach the first screen or menu of the game.
Post Condition	- The user has successfully gone back to the previous screen or menu.

UC_08– US_08 – Back Button

Name	Enter door key clicked

Summary	User can change the key to enter the door by clicking it in the settings menu.
Actor	User
Description	Allows the user to change the key that is required to enter a specific door.
Precondition	-The game is running and accessible. -The user has reached a door that requires a key to unlock. -The user has the administrative rights to change the key.
Alternatives	- If the user attempts to change the key to a key that is already in use for another door, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user does not have administrative rights to change the key, the game should display an error message or sound effect indicating that the user is unauthorized to make changes to the game settings.
Post Condition	- The user has successfully changed the key required to enter the specific door.

UC_09– US_09 – Enter Door Key Clicked

Name	Left Key Clicked

Summary	User can change the left key by clicking it in the settings menu.
Actor	User
Description	Allows the user to change the key that is used to move the player character to the left.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to move the player character to the left.

UC_10– US_10 – Left Key Clicked

Name	Down Key Clicked

Summary	User can change the down key by clicking it in the settings menu.
Actor	User
Description	Allows the user to change the key that is used to make the player character crouch or move downwards.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to make the player character crouch or move downwards.

UC_11– US_11 – Down Key Clicked

Name	Right Key Clicked

Summary	User can change the right key by clicking it in the settings menu.
Actor	User
Description	Allows the user to change the key that is used to move the player character to the right.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to move the player character to the right.

UC_12– US_12 – Right Key Clicked

Name	Jump Key Clicked

Summary	User can change the jump key by clicking it in the settings menu.
Actor	User
Description	Allows the user to change the key that is used to make the player character jump.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to make the player character jump.

UC_13– US_13 – Jump Key Clicked

Name	Power1 Key Clicked
Summary	User can change the power1 key by clicking it in the

	settings menu.
Actor	User
Description	Allows the user to change the key that is used to make the player uses superpowers.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to make the player uses superpowers.

UC_14– US_14 – Power1 Key Clicked

Name	Power2 Key Clicked
Summary	User can change the power2 key by clicking it in the

	settings menu.
Actor	User
Description	Allows the user to change the key that is used to make the player uses superpowers.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user attempts to change the key to a key that is already assigned to another action, the game should display an error message or sound effect indicating that the key is already in use and prompt the user to select a different key. -If the user changes the left key but does not see a change in the player character's movement, the game should prompt the user to try again or provide an option to reset the game settings to default.
Post Condition	- The user has successfully changed the key used to make the player uses superpowers.

UC_15– US_15 – Power2 Key Clicked

Name	User Presses The Reset Button
Summary	User can press the reset button to reset the settings.

Actor	User
Description	Allows the user to reset the game settings to their default values.
Precondition	-The game is running and accessible. -The user has access to the game settings
Alternatives	-If the user cancels the reset, the game should not make any changes to the game settings. -If the game encounters an error while resetting the game settings, the game should display an error message or sound effect indicating that the reset was not completed and provide an option to try again or contact technical support.
Post Condition	- The user has successfully reset the game settings to their default values.

UC_16– US_16 – User Presses The Reset Button.

Name	Game music slider is moved
Summary	User can move the game music slider.
Actor	User
Description	Allows the user to adjust the volume of the game's background music.
Precondition	-The game is running and accessible. -The user has access to the game settings -The game has a background music slider in the settings menu.
Alternatives	-If the game does not have background music or the music is turned off, moving the slider should have no effect in the game.
Post Condition	- The user has successfully reset the game settings to their default values.

UC_17– US_17 – Game Music Slider Is Moved.

Name	Background music slider is moved

Summary	User can move the background music slider.
Actor	User
Description	Allows the user to adjust the volume of the game's background music.
Precondition	-The game is running and accessible. -The user has access to the game settings -The game has a background music slider in the settings menu.
Alternatives	- If the game does not have background music or the music is turned off, moving the slider should have no effect on the game.
Post Condition	-The user has successfully adjusted the volume of the game's background music.

UC_18– US_18 – Background Music Slider Is Moved.

Name	User presses the resume button.
Summary	User can press the resume button .
Actor	User
Description	Allows allow the user to continue playing the game from the point they left off.
Precondition	-The game is running and accessible. -The user has previously started playing the game and paused or stopped the game. -The game has a "Resume" button or similar option available.
Alternatives	- If the user has not yet played the game or has no previous progress saved, the "Resume" button may not be available or may not have any effect. - If the game has lost or corrupted the user's save data, the "Resume" button may not be available or may start the game from a default point or the beginning of the game.
Post Condition	- The user has successfully resumed the game from their previous progress.

UC_19– US_19 – User presses the resume button.

Name	User presses the restart button.
Summary	User can press the restart button to start the game from the beginning.
Actor	User
Description	Allows the user to restart the current level or the entire game from the beginning.
Precondition	- The game is running and accessible. -The user is currently playing a level or has completed a level and is presented with the option to continue or restart.
Alternatives	- If the user presses the restart button by mistake or changes their mind, they should be presented with an option to cancel the restart and return to the game without any changes. - If the user restarts the current level, any progress made in the level up to that point will be lost.
Post Condition	- The user has successfully restarted the current level.

UC_20– US_20 – User Presses The Restart Button.

Name	User presses the quit button at pause menu.
Summary	User can press the quit button at pause menu.
Actor	User
Description	Allows the user to quit the game from the pause menu.
Precondition	-The game is running and accessible. -The user has paused the game and opened the pause menu.
Alternatives	- If the user selects the "Quit" option but then decides to cancel the quit operation, the game should resume where it left off in the paused state.
Post Condition	-The user has successfully quit the game.

UC_21– US_21 – User Presses The Quit Button At Pause Menu

Name	Enter door key pressed
Summary	User can press a key to open the door.
Actor	User
Description	Allows the user to interact with a door in the game by pressing a key to enter it.
Precondition	-The game is running and accessible. -The user has reached a door in the game.
Alternatives	- If the user presses the wrong key to enter the door, the game should display an error message or sound effect indicating that the wrong key has been pressed and prompt the user to try again.
Post Condition	- The user has successfully entered the door and is now in a new room or level of the game.

UC_22– US_22 – Enter Door Key Pressed

Name	Left key pressed
Summary	User can move to the left when user presses that command.
Actor	User
Description	Allows the user to move to the left when left arrow is pressed.
Precondition	- The game is running and accessible. -The player character is currently able to move left.
Alternatives	- If the player presses a key that is not the left arrow key, the game should not register the input and the player character should not move left. - If the player character is in a state where they cannot move, such as being stunned or knocked down, pressing the left arrow key should have no effect and the player character should not move left.
Post Condition	- The player character has moved to the left and the game state has been updated accordingly.

UC_23– US_23 – Left Key Pressed

Name	Down key pressed
Summary	User can crouch when user presses that command.
Actor	User
Description	Allows the user to make the player character crouch when the down key is pressed.
Precondition	- The game is running and accessible. -The player has control of the player character.
Alternatives	- If the player presses a key that is not the down key, the game should not register the input and the player character should not move left.
Post Condition	-The player character has either crouched based on the down key press.

UC_24– US_24 – Down Key Pressed

Name	Right key pressed
Summary	User can move to the right when user presses that command.
Actor	User
Description	Allows the user to move to the right when right arrow is pressed.
Precondition	- The game is running and accessible. -The player character is currently able to move right.
Alternatives	- If the player presses a key that is not the right arrow key, the game should not register the input and the player character should not move right. - If the player character is in a state where they cannot move, such as being stunned or knocked down, pressing the right arrow key should have no effect and the player character should not move right.
Post Condition	- The player character has moved to the right and the game state has been updated accordingly.

UC_25– US_25 – Right Key Pressed

Name	Jump key pressed
Summary	User can move to the right when user presses that command.
Actor	User
Description	Allows the user to make the player character jump by pressing the jump key.
Precondition	-The game is running and accessible. -The player has control of the player character.
Alternatives	- If the player attempts to press the jump key while the player character is unable to jump (e.g. if they are already mid-jump or if they are currently falling), the game should ignore the input
Post Condition	-The player character has successfully jumped in response to the player's input.

UC_26– US_26 – Jump Key Pressed

Name	Power1 key pressed.
Summary	User can use superpowers by pressing this key.
Actor	User
Description	Allows the user to activate a superpower by pressing a designated "Power1" key.
Precondition	-The game is running and accessible. -The user has access to the game controls.
Alternatives	- If the user presses the "Power1" key ,the game does not recognize the input,
Post Condition	- The user has successfully activated superpower pressing the "Power1" key.

UC_27– US_27 – Power1 Key Pressed

Name	Power2 key pressed.
Summary	User can use superpowers by pressing this key.
Actor	User
Description	Allows the user to activate a superpower by pressing a designated "Power2" key.
Precondition	-The game is running and accessible. -The user has access to the game controls.
Alternatives	- If the user presses the "Power2" key ,the game does not recognize the input,
Post Condition	- The user has successfully activated superpower pressing the "Power2" key.

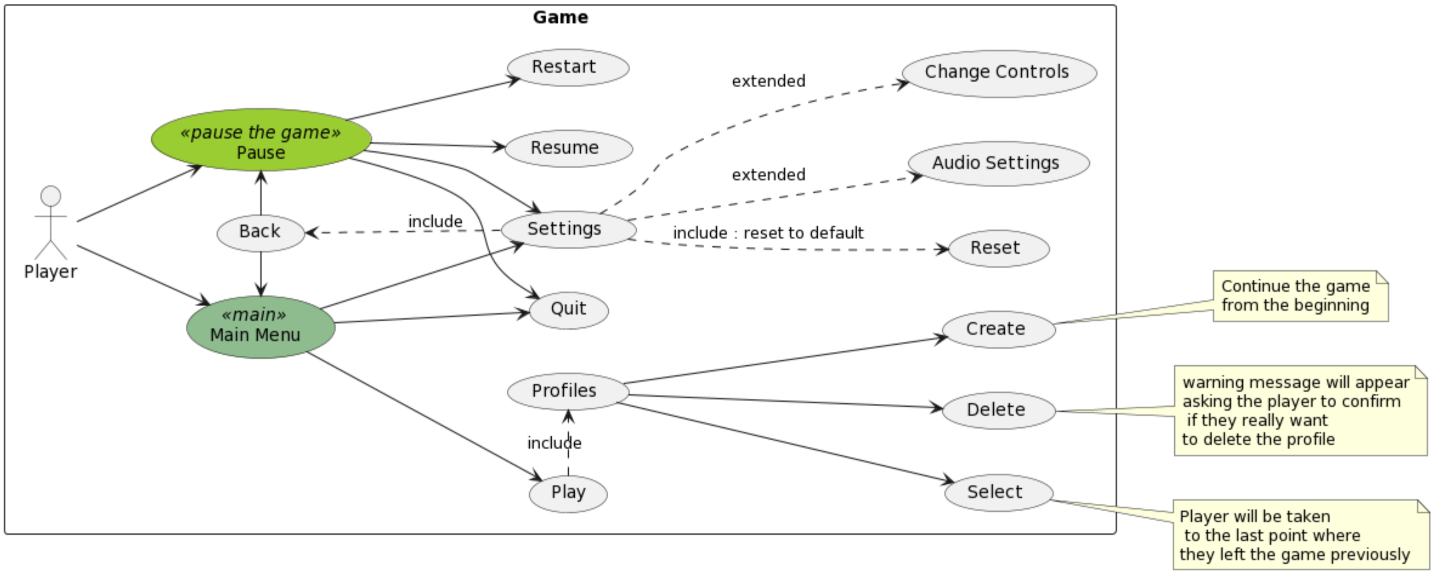
UC_28– US_28– Power2 Key Pressed

Name	Level finished
Summary	User can progress to another level after completing the current one.
Actor	User
Description	Allows the user to progress to the next level after completing the current level.
Precondition	- The game is running and accessible. -The user is currently playing a level.
Alternatives	- If the player does not reach the end of the level, the game should not trigger the level completion event and should allow the player to continue playing the level. - If the player chooses to replay the completed level, the game should reset the level to its initial state and allow the player to replay it.
Post Condition	- The user has successfully completed the level and has the option to progress to the next level or replay the completed level.

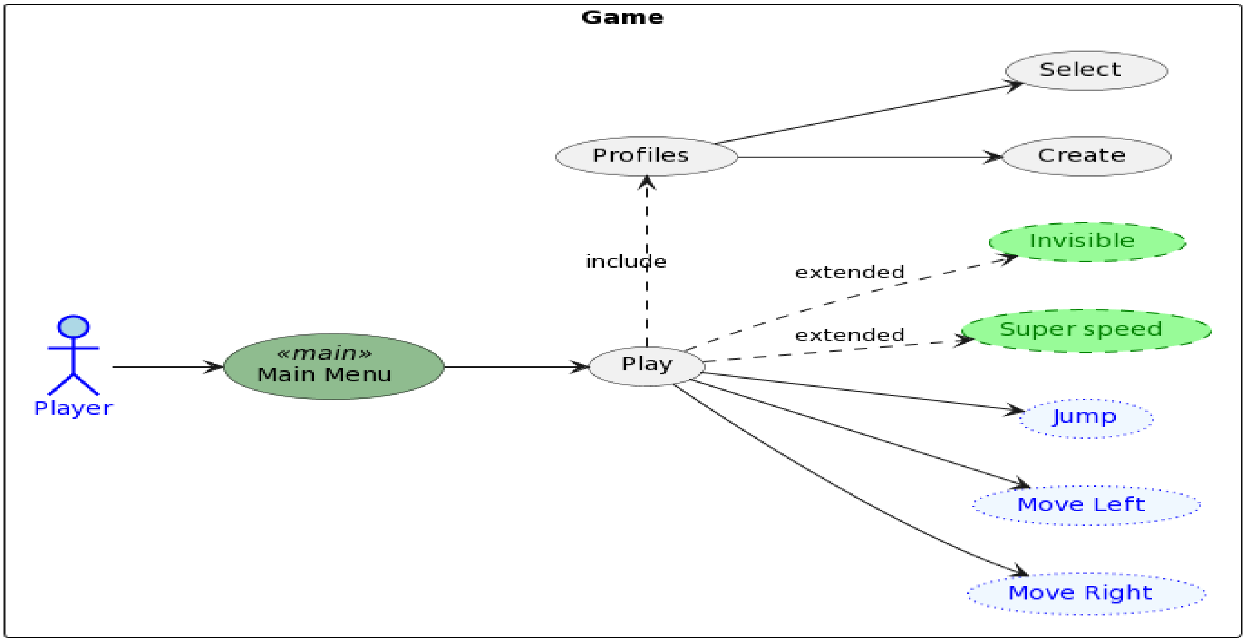
UC_29– US_29 – Level Finished

4.2 Behavioral Diagrams

4.2.1 Use Case Diagrams

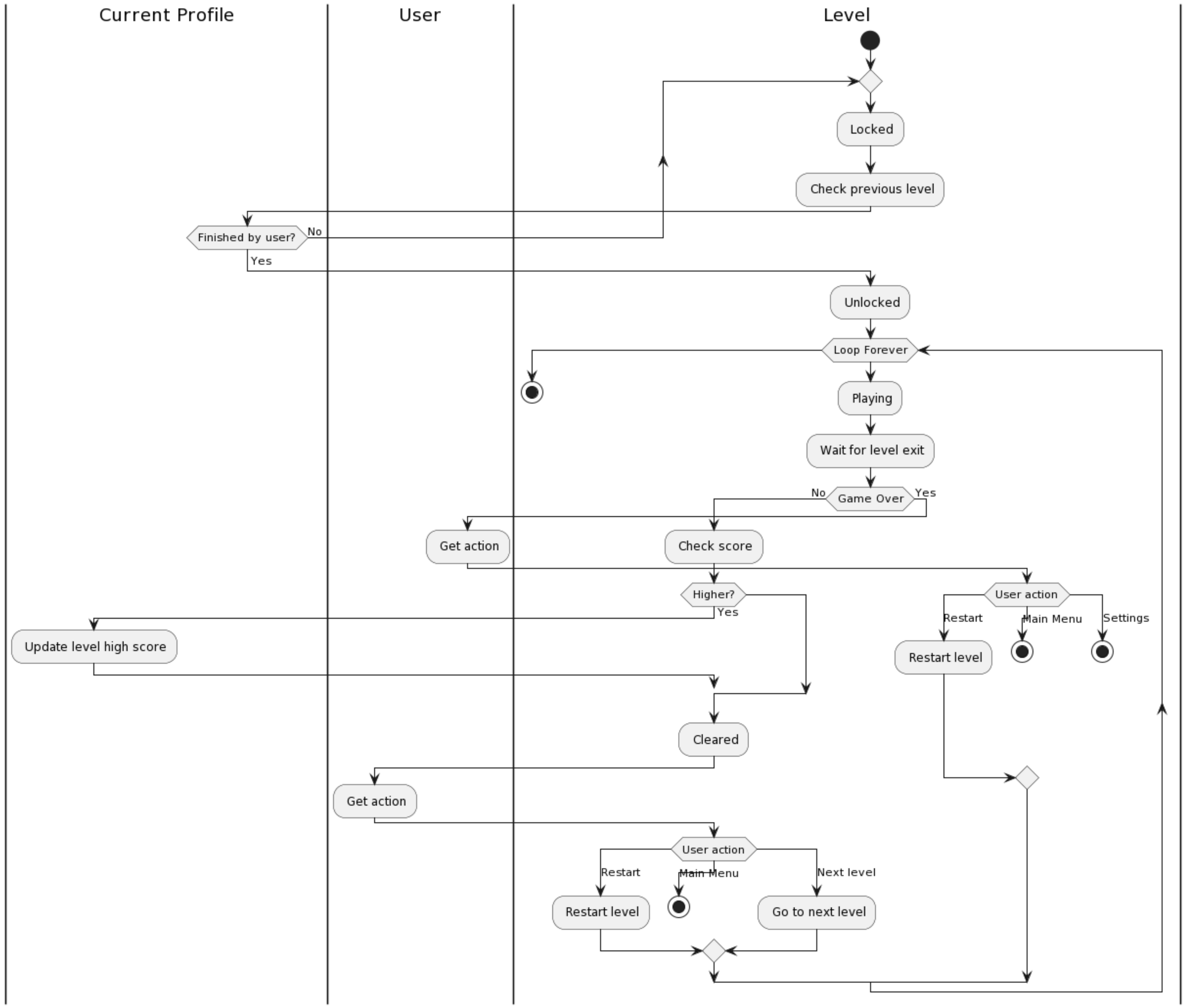


CD_01 - Basic Operations

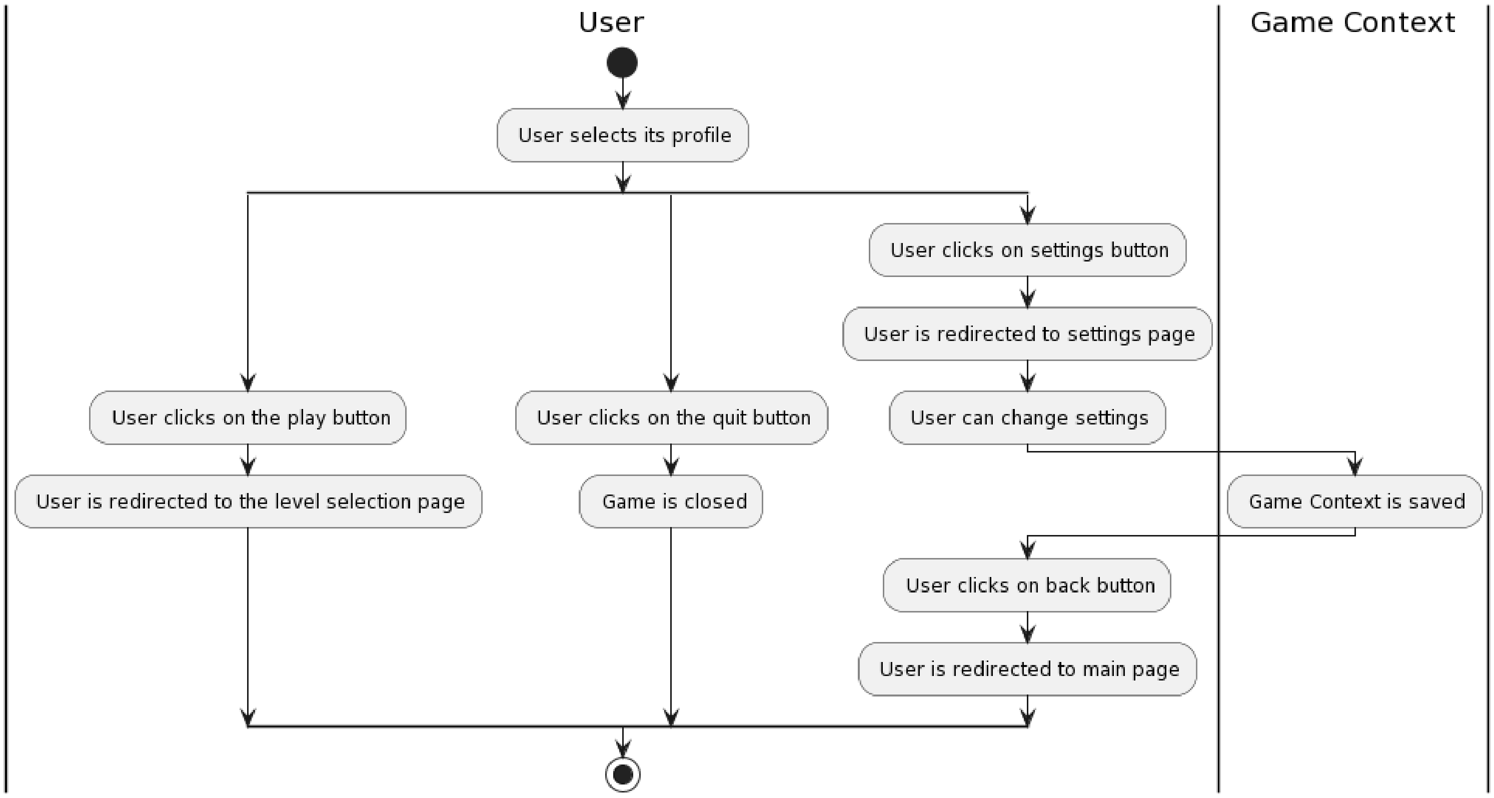


CD_02 - Game Operations

4.2.2 Activity Diagrams



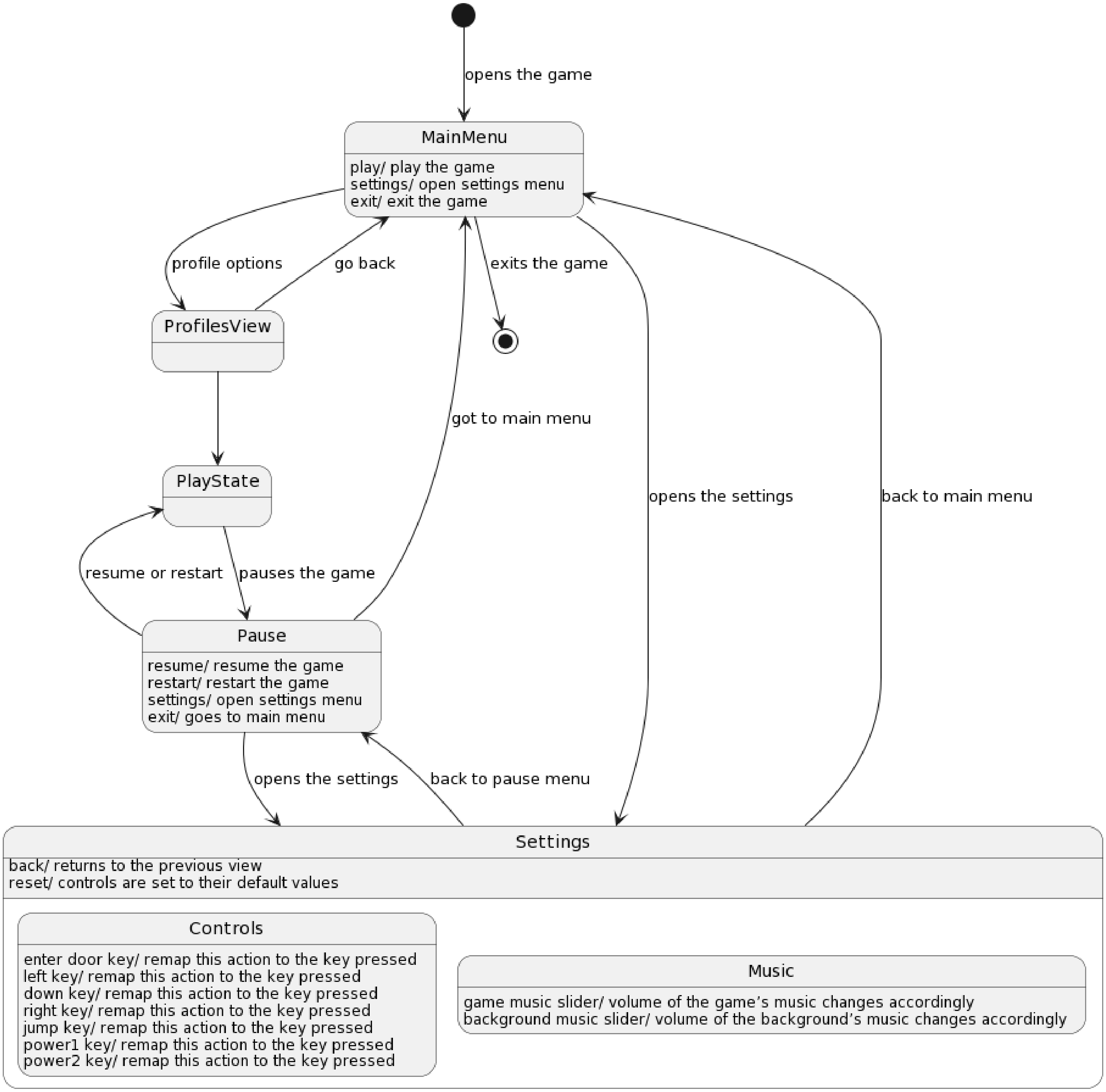
AD_01 - Level


User
Game Context
User selects its profile
User clicks on settings button
User is redirected to settings page
User clicks on the play button
User clicks on the quit button
User can change settings
User is redirected to the level selection page
Game is closed
Game Context is saved
User clicks on back button
User is redirected to main page

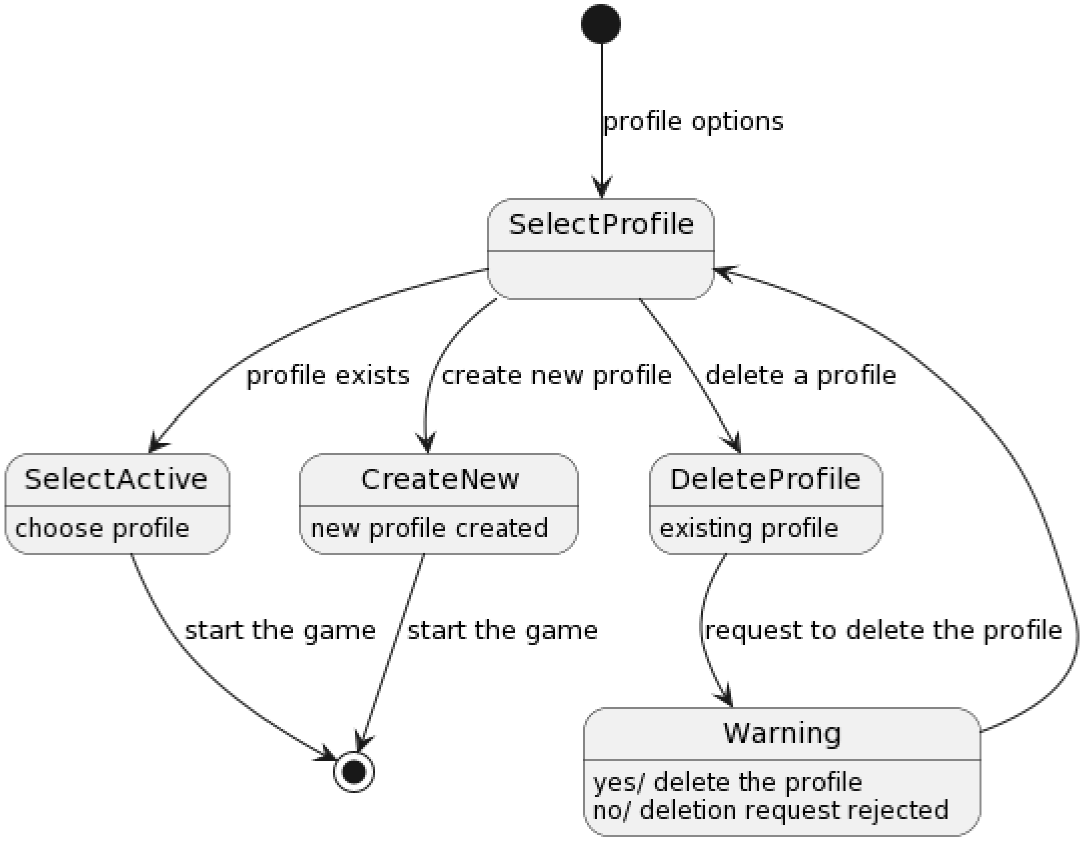
AD_02 – Game Scenes

4.2.3 State Diagrams

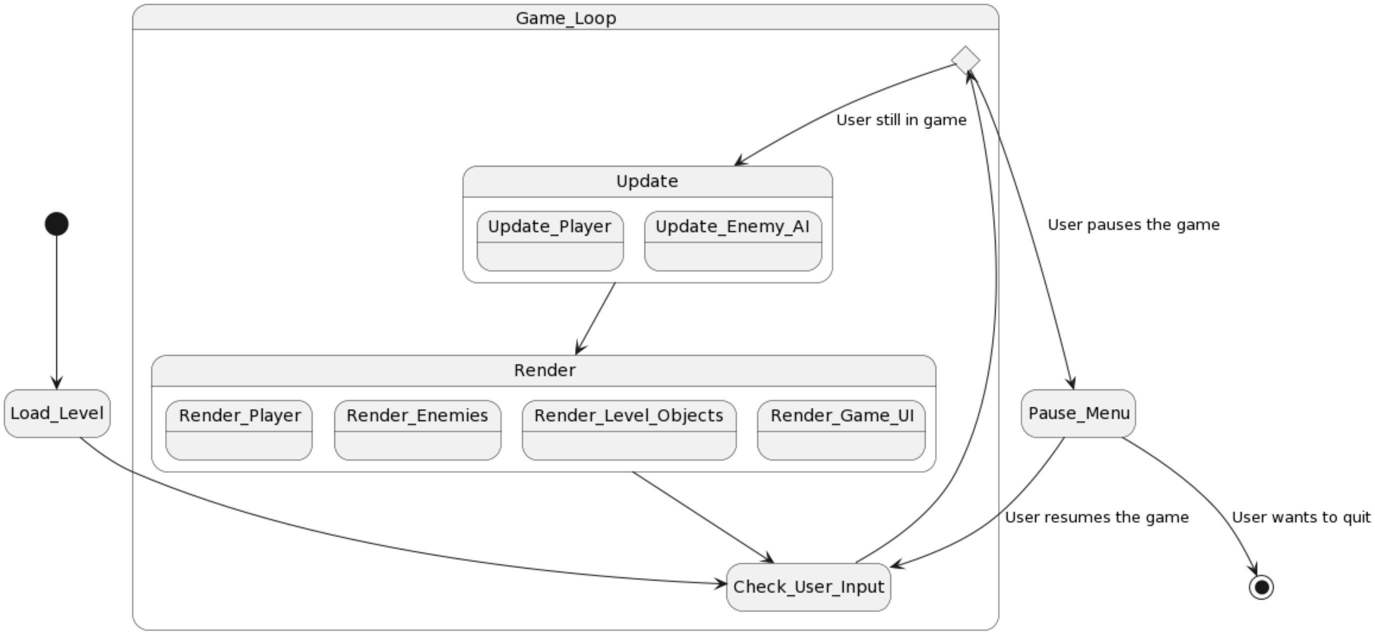
AD_02 – Game Scenes


opens the game
MainMenu
play/ play the game
settings/ open settings menu
exit/ exit the game
profile options
go back
exits the game
ProfilesView
got to main menu
opens the settings
back to main menu
PlayState
resume or restart
pauses the game
Pause
resume/ resume the game
restart/ restart the game
settings/ open settings menu
exit/ goes to main menu
opens the settings
back to pause menu
Settings
back/ returns to the previous view
reset/ controls are set to their default values
Controls
enter door key/ remap this action to the key pressed
left key/ remap this action to the key pressed
down key/ remap this action to the key pressed
right key/ remap this action to the key pressed
jump key/ remap this action to the key pressed
power1 key/ remap this action to the key pressed
power2 key/ remap this action to the key pressed
Music
game music slider/ volume of the game's music changes accordingly
background music slider/ volume of the background's music changes accordingly

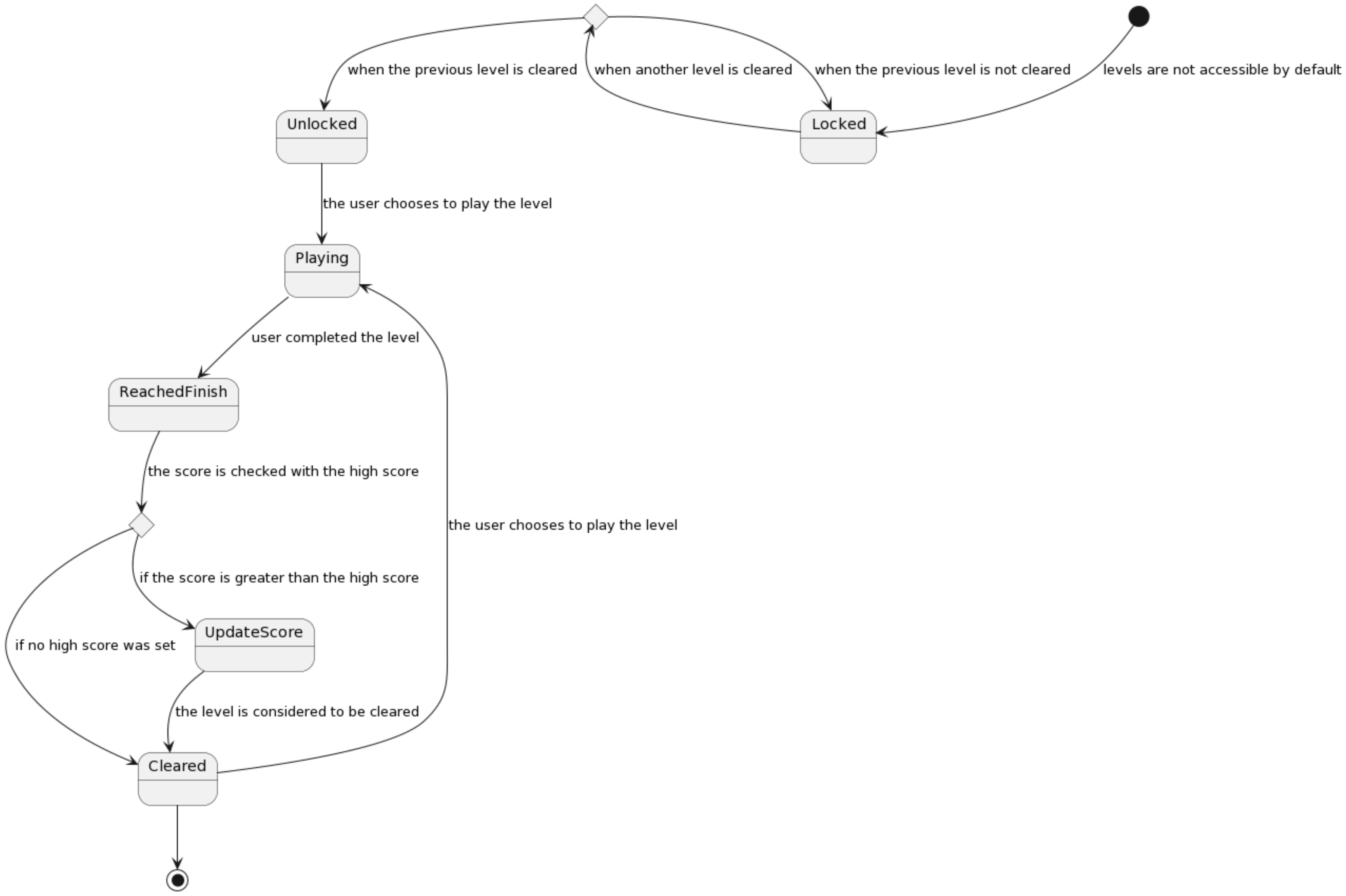
SD_01 – Game Structure



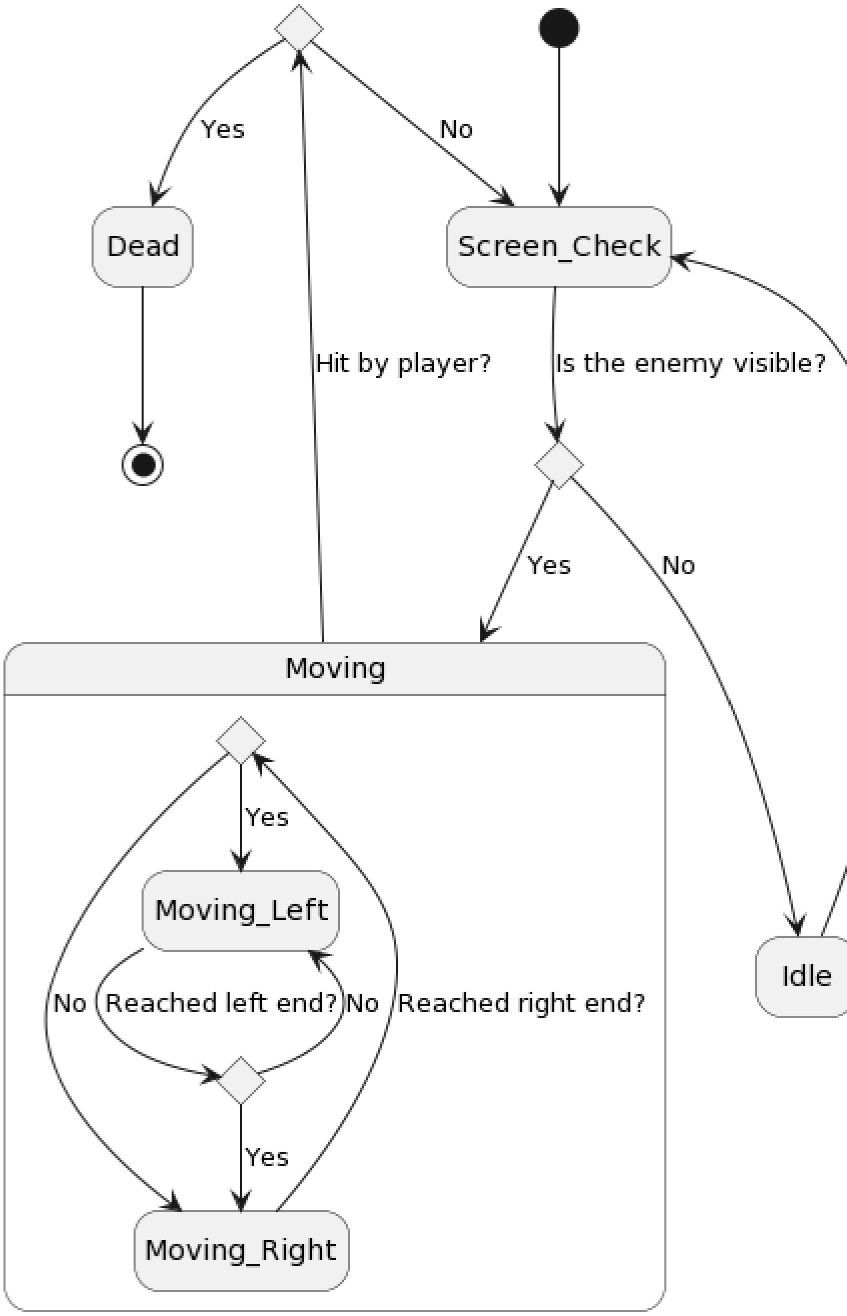
SD_02 – Profile View



SD_03 – Game Loop

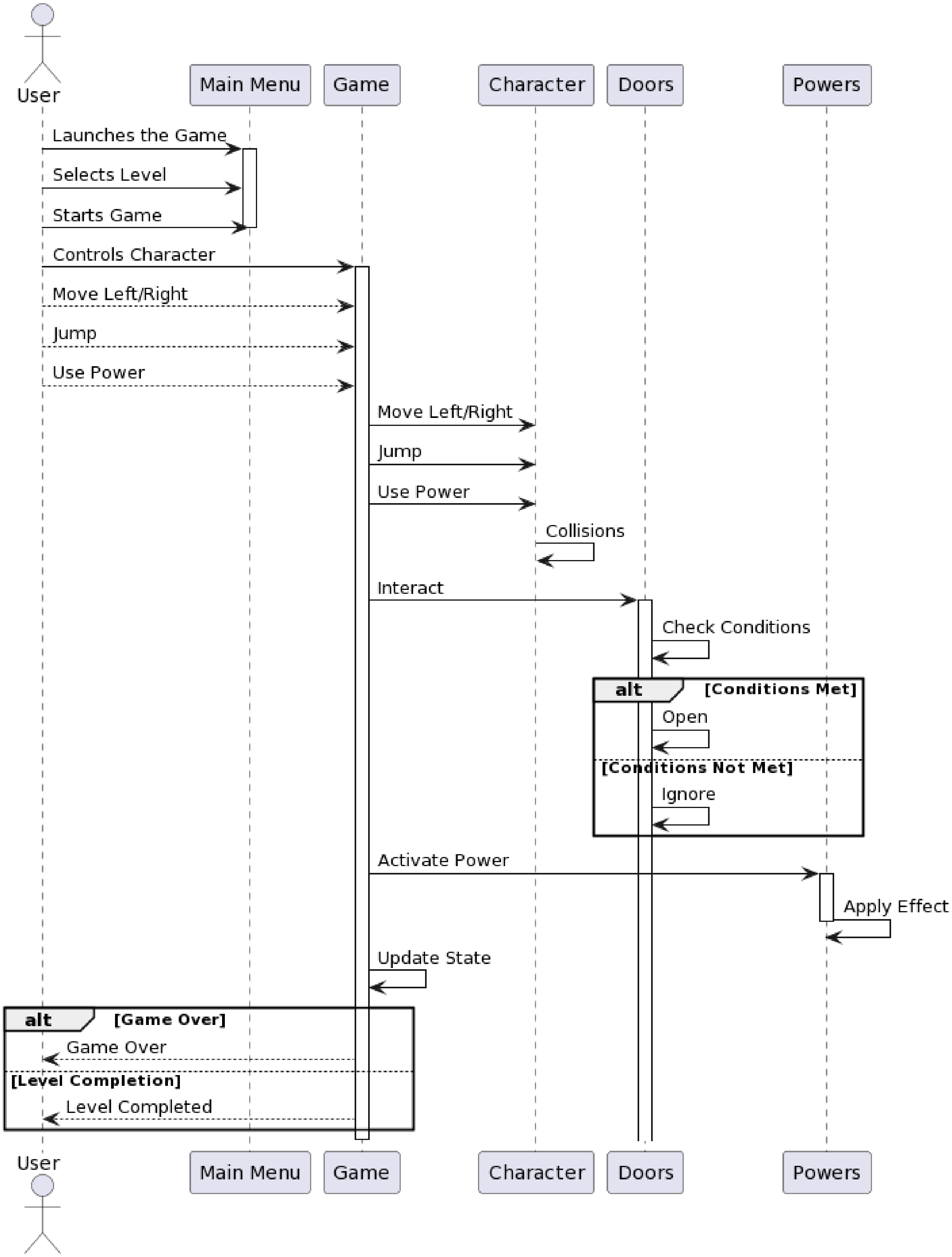

when the previous level is cleared
when another level is cleared
when the previous level is not cleared
levels are not accessible by default
Unlocked
Locked
the user chooses to play the level
Playing
user completed the level
ReachedFinish
the score is checked with the high score
the user chooses to play the level
if the score is greater than the high score
if no high score was set
UpdateScore
the level is considered to be cleared
Cleared

SD_04 – Levels


Yes
No
Dead
Screen_Check
Hit by player?
Is the enemy visible?
Yes
No
Moving
Yes
Moving_Left
No
Reached left end?
No
Reached right end?
Yes
Moving_Right
Idle

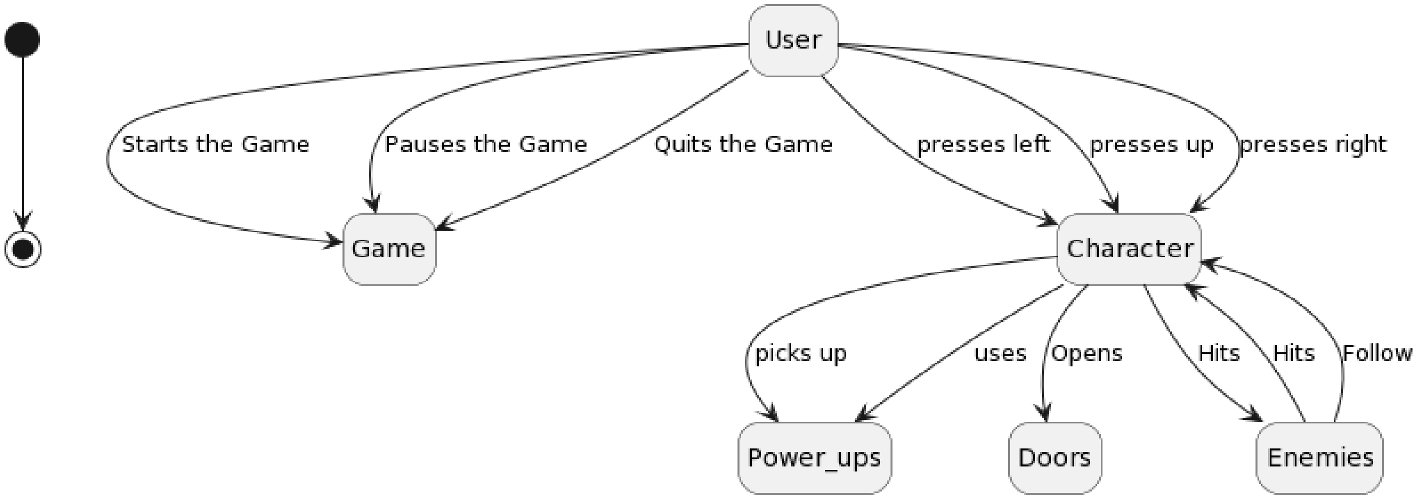
SD_05 – Enemy

4.2.4 Sequence Diagrams



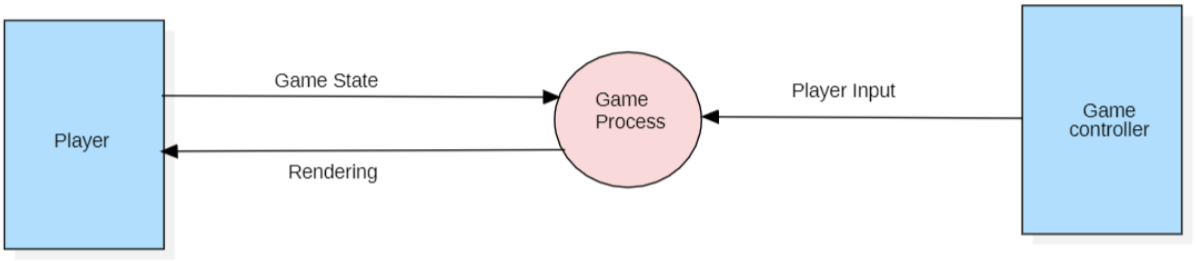
SeqD_01 – Game Organization

4.2.5 Collaboration Diagrams

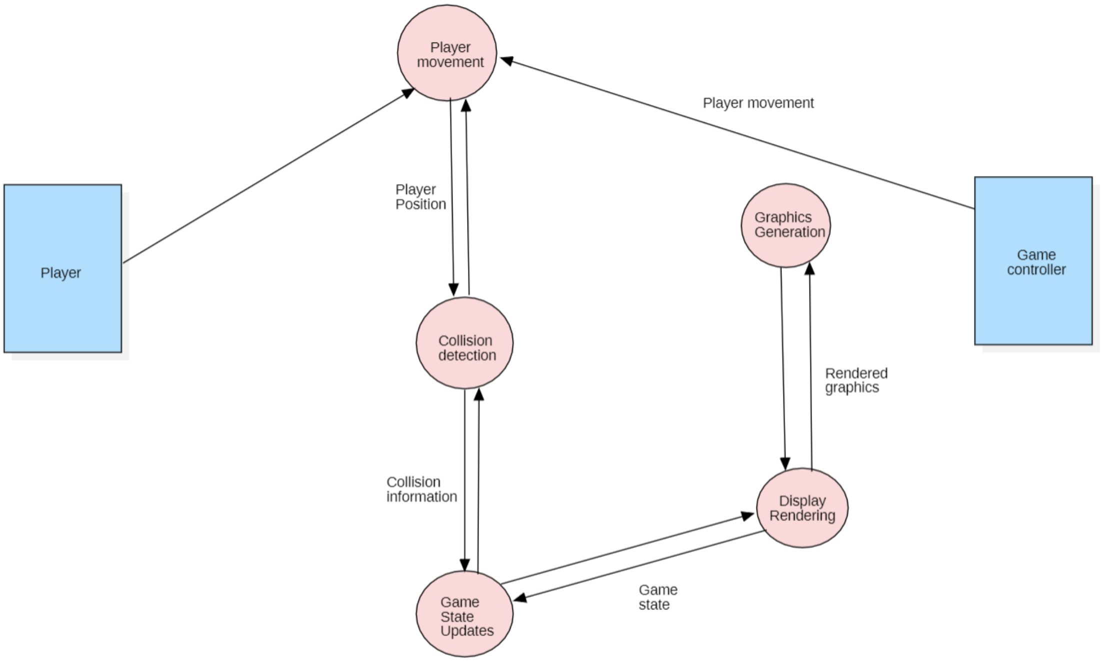


ColD_01

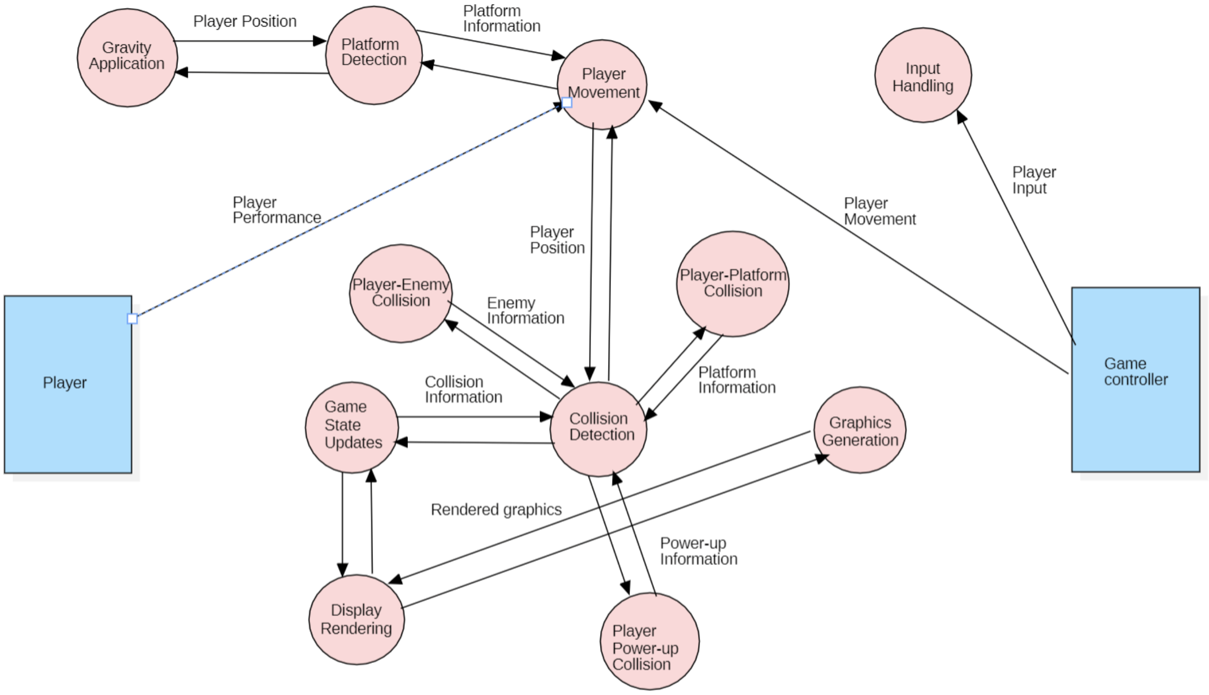
4.3 Data Flow Diagrams



DFD LEVEL 0



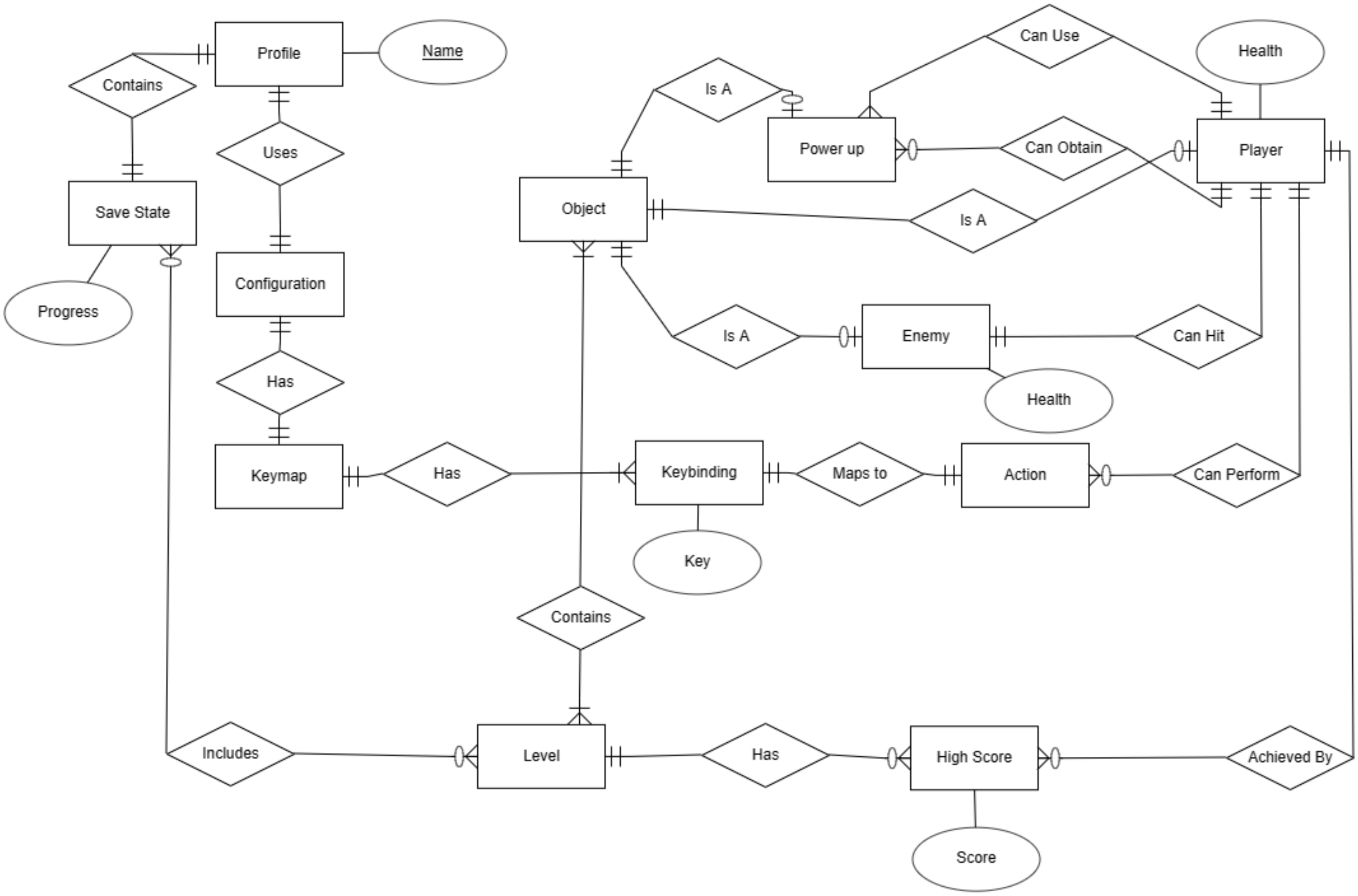
DFD LEVEL 1



DFD LEVEL 2

4.4 Entry Relation

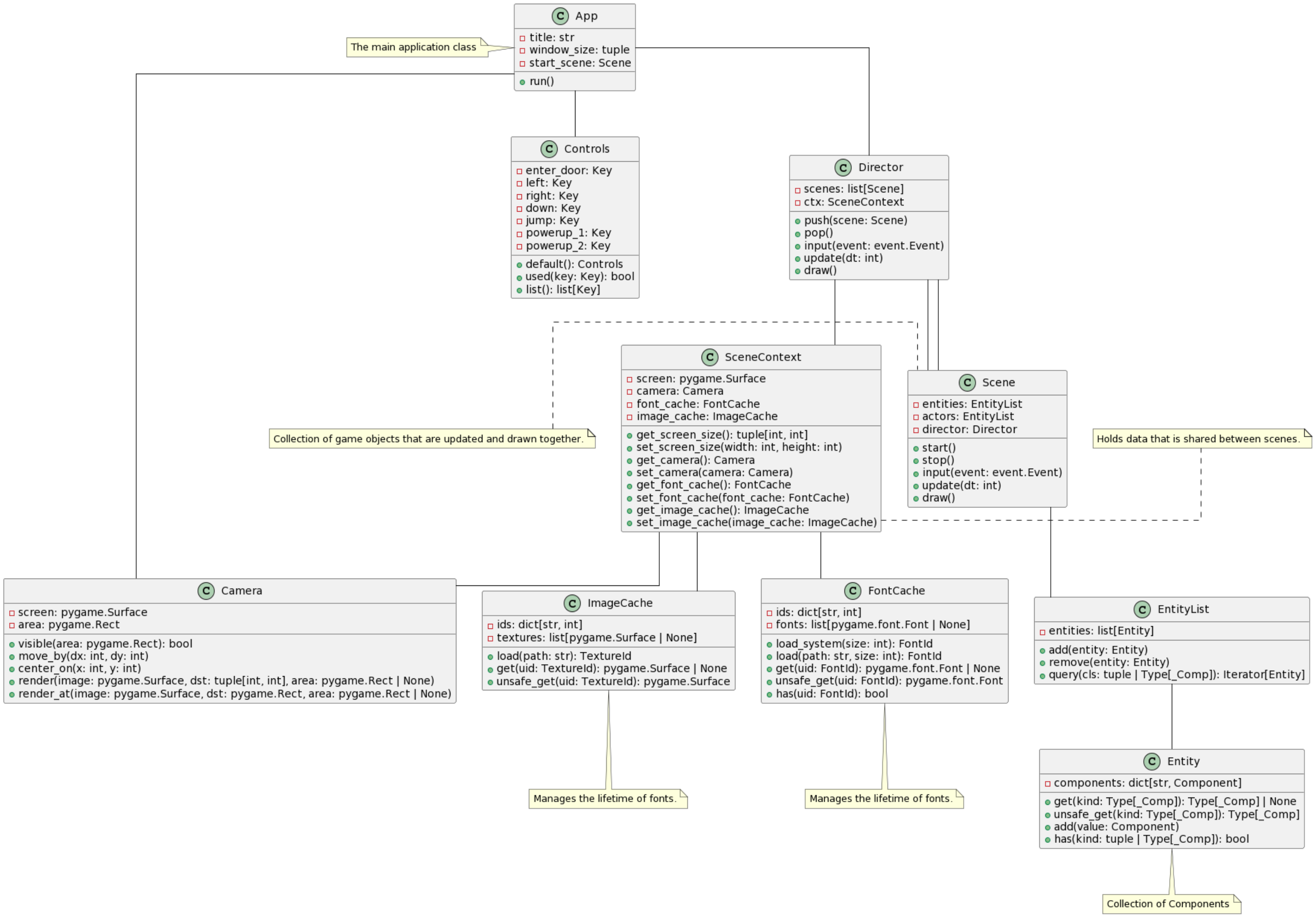
4.4.1 Entity Relation Diagram



ERD_01

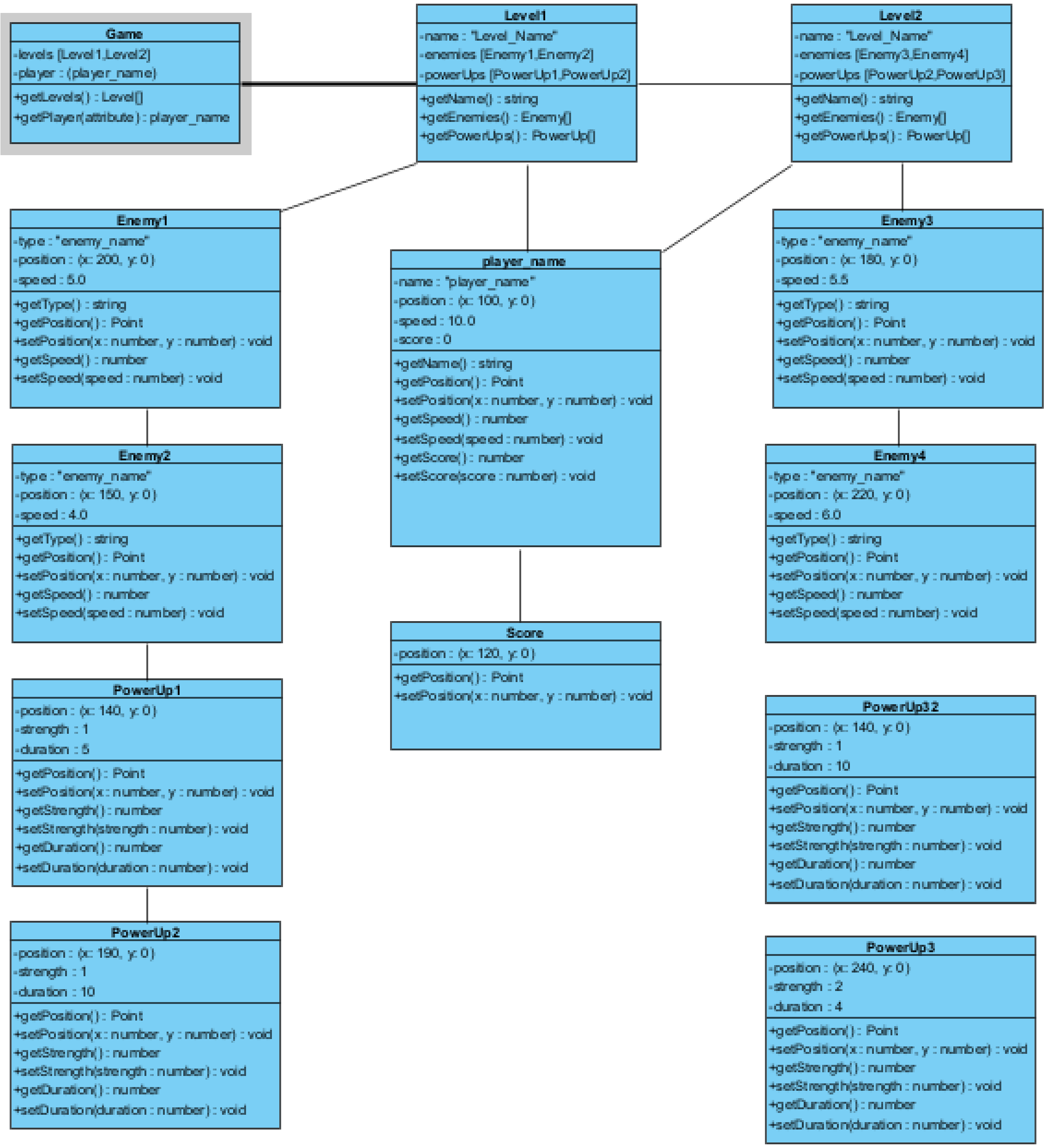
4.5 Structural Diagrams

4.5.1 Class Diagram


The main application class
App
title: str
window_size: tuple
start_scene: Scene
run()
Controls
enter_door: Key
left: Key
right: Key
down: Key
jump: Key
powerup_1: Key
powerup_2: Key
default(): Controls
used(key: Key): bool
list(): list[Key]
Director
scenes: list[Scene]
ctx: SceneContext
push(scene: Scene)
pop()
input(event: event.Event)
update(dt: int)
draw()
SceneContext
screen: pygame.Surface
camera: Camera
font_cache: FontCache
image_cache: ImageCache
get_screen_size(): tuple[int, int]
set_screen_size(width: int, height: int)
get_camera(): Camera
set_camera(camera: Camera)
get_font_cache(): FontCache
set_font_cache(font_cache: FontCache)
get_image_cache(): ImageCache
set_image_cache(image_cache: ImageCache)
Collection of game objects that are updated and drawn together.
Scene
entities: EntityList
actors: EntityList
director: Director
start()
stop()
input(event: event.Event)
update(dt: int)
draw()
Holds data that is shared between scenes.
Camera
screen: pygame.Surface
area: pygame.Rect
visible(area: pygame.Rect): bool
move_by(dx: int, dy: int)
center_on(x: int, y: int)
render(image: pygame.Surface, dst: tuple[int, int], area: pygame.Rect | None)
render_at(image: pygame.Surface, dst: pygame.Rect, area: pygame.Rect | None)
ImageCache
ids: dict[str, int]
textures: list[pygame.Surface | None]
load(path: str): TextureId
get(uid: TextureId): pygame.Surface | None
unsafe_get(uid: TextureId): pygame.Surface
FontCache
ids: dict[str, int]
fonts: list[pygame.font.Font | None]
load_system(size: int): FontId
load(path: str, size: int): FontId
get(uid: FontId): pygame.font.Font | None
unsafe_get(uid: FontId): pygame.font.Font
has(uid: FontId): bool
EntityList
entities: list[Entity]
add(entity: Entity)
remove(entity: Entity)
query(cls: tuple | Type[_Comp]): Iterator[Entity]
Manages the lifetime of fonts.
Manages the lifetime of fonts.
Entity
components: dict[str, Component]
get(kind: Type[_Comp]): Type[_Comp] | None
unsafe_get(kind: Type[_Comp]): Type[_Comp]
add(value: Component)
has(kind: tuple | Type[_Comp]): bool
Collection of Components

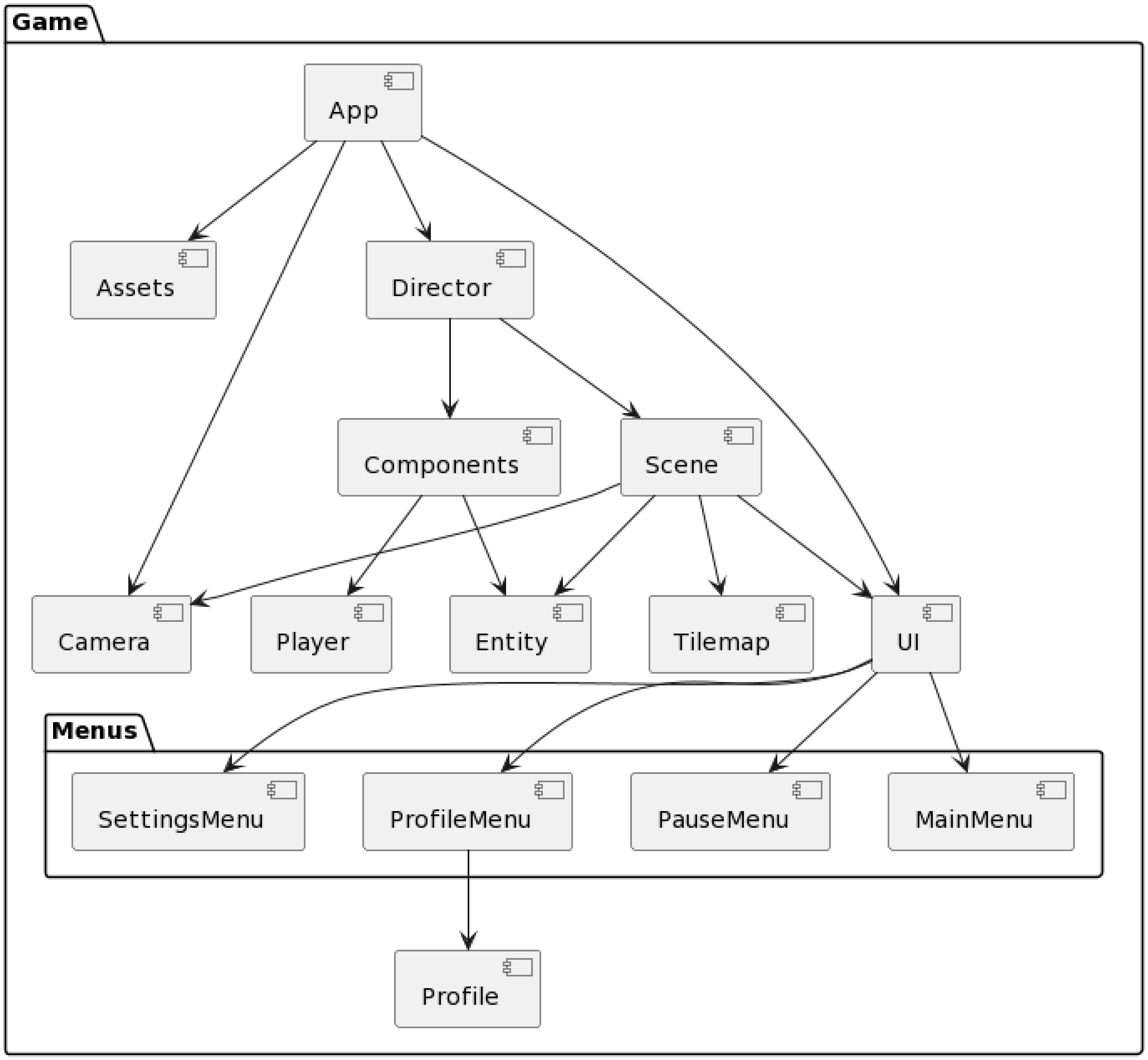
CD_01

4.5.2 Object Diagrams


Game
-levels [Level1,Level2]
-player : (player_name)
+getLevels() : Level[]
+getPlayer(attribute) : player_name
Level1
-name : "Level_Name"
-enemies [Enemy1,Enemy2]
-powerUps [PowerUp1,PowerUp2]
+getName() : string
+getEnemies() : Enemy[]
+getPowerUps() : PowerUp[]
Level2
-name : "Level_Name"
-enemies [Enemy3,Enemy4]
-powerUps [PowerUp2,PowerUp3]
+getName() : string
+getEnemies() : Enemy[]
+getPowerUps() : PowerUp[]
Enemy1
-type : "enemy_name"
-position : (x: 200, y: 0)
-speed : 5.0
+getType() : string
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getSpeed() : number
+setSpeed(speed : number) : void
player_name
-name : "player_name"
-position : (x: 100, y: 0)
-speed : 10.0
-score : 0
+getName() : string
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getSpeed() : number
+setSpeed(speed : number) : void
+getScore() : number
+setScore(score : number) : void
Enemy3
-type : "enemy_name"
-position : (x: 180, y: 0)
-speed : 5.5
+getType() : string
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getSpeed() : number
+setSpeed(speed : number) : void
Enemy2
-type : "enemy_name"
-position : (x: 150, y: 0)
-speed : 4.0
+getType() : string
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getSpeed() : number
+setSpeed(speed : number) : void
Enemy4
-type : "enemy_name"
-position : (x: 220, y: 0)
-speed : 6.0
+getType() : string
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getSpeed() : number
+setSpeed(speed : number) : void
Score
-position : (x: 120, y: 0)
+getPosition() : Point
+setPosition(x : number, y : number) : void
PowerUp1
-position : (x: 140, y: 0)
-strength : 1
-duration : 5
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getStrength() : number
+setStrength(strength : number) : void
+getDuration() : number
+setDuration(duration : number) : void
PowerUp32
-position : (x: 140, y: 0)
-strength : 1
-duration : 10
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getStrength() : number
+setStrength(strength : number) : void
+getDuration() : number
+setDuration(duration : number) : void
PowerUp2
-position : (x: 190, y: 0)
-strength : 1
-duration : 10
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getStrength() : number
+setStrength(strength : number) : void
+getDuration() : number
+setDuration(duration : number) : void
PowerUp3
-position : (x: 240, y: 0)
-strength : 2
-duration : 4
+getPosition() : Point
+setPosition(x : number, y : number) : void
+getStrength() : number
+setStrength(strength : number) : void
+getDuration() : number
+setDuration(duration : number) : void

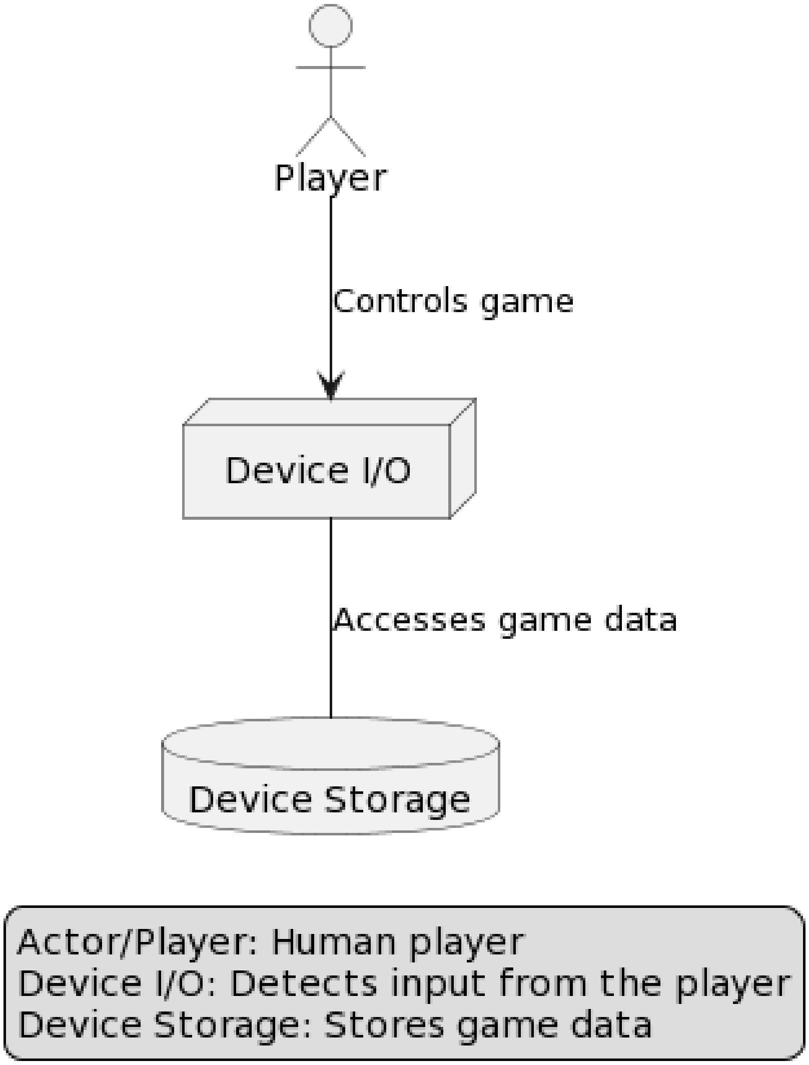
OD_01

4.5.3 Component Diagrams



CoD_01

4.5.4 Deployment Diagram


Player
Controls game
Device I/O
Accesses game data
Device Storage
Actor/Player: Human player
Device I/O: Detects input from the player
Device Storage: Stores game data

DD_01

5 Implementation Technology

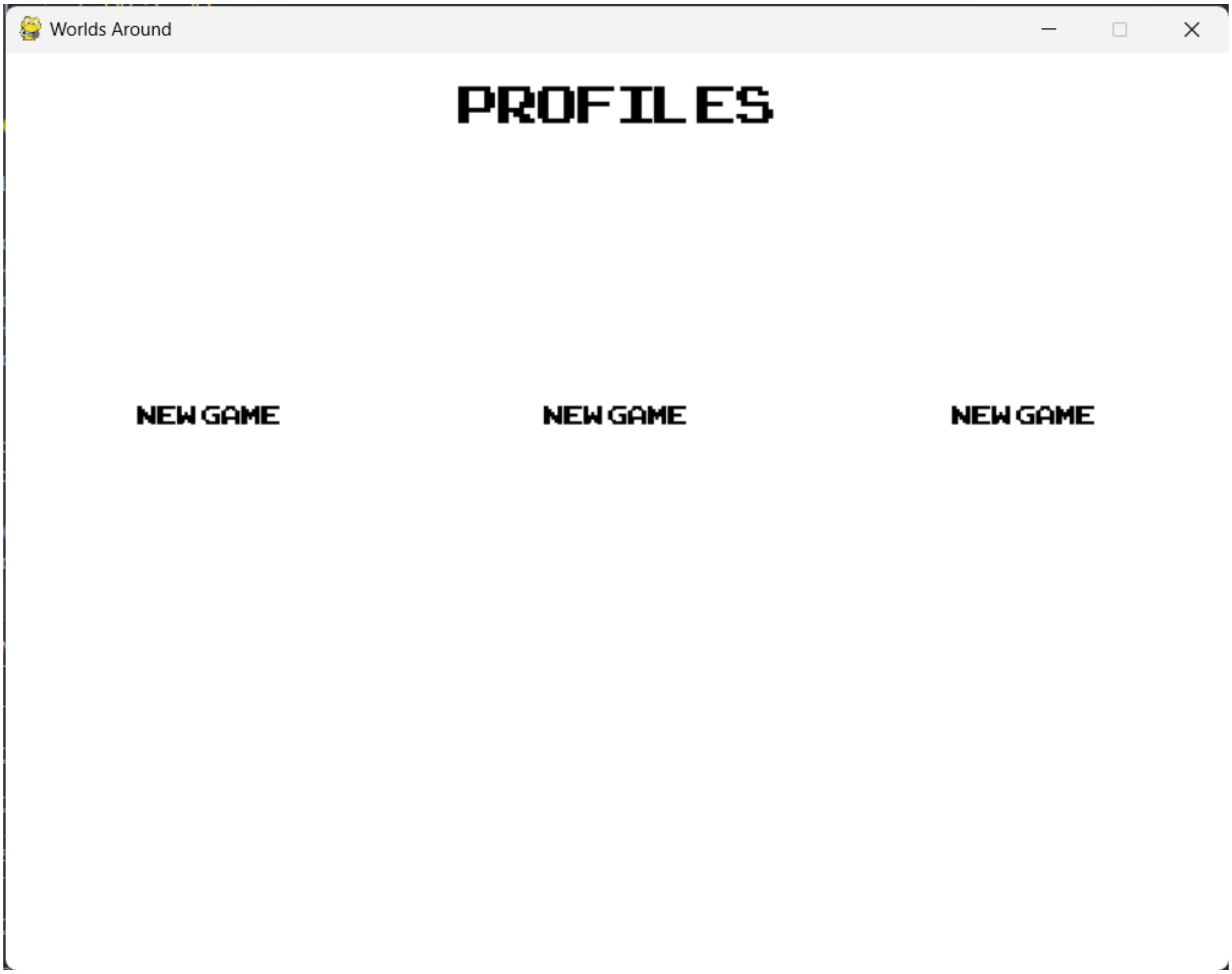
Technologies used:

- Python 3
- PyGame
- PlantUML
- Json data format
- Immediate Mode GUI library (in-house solution)
- Rect-cut layout algorithm

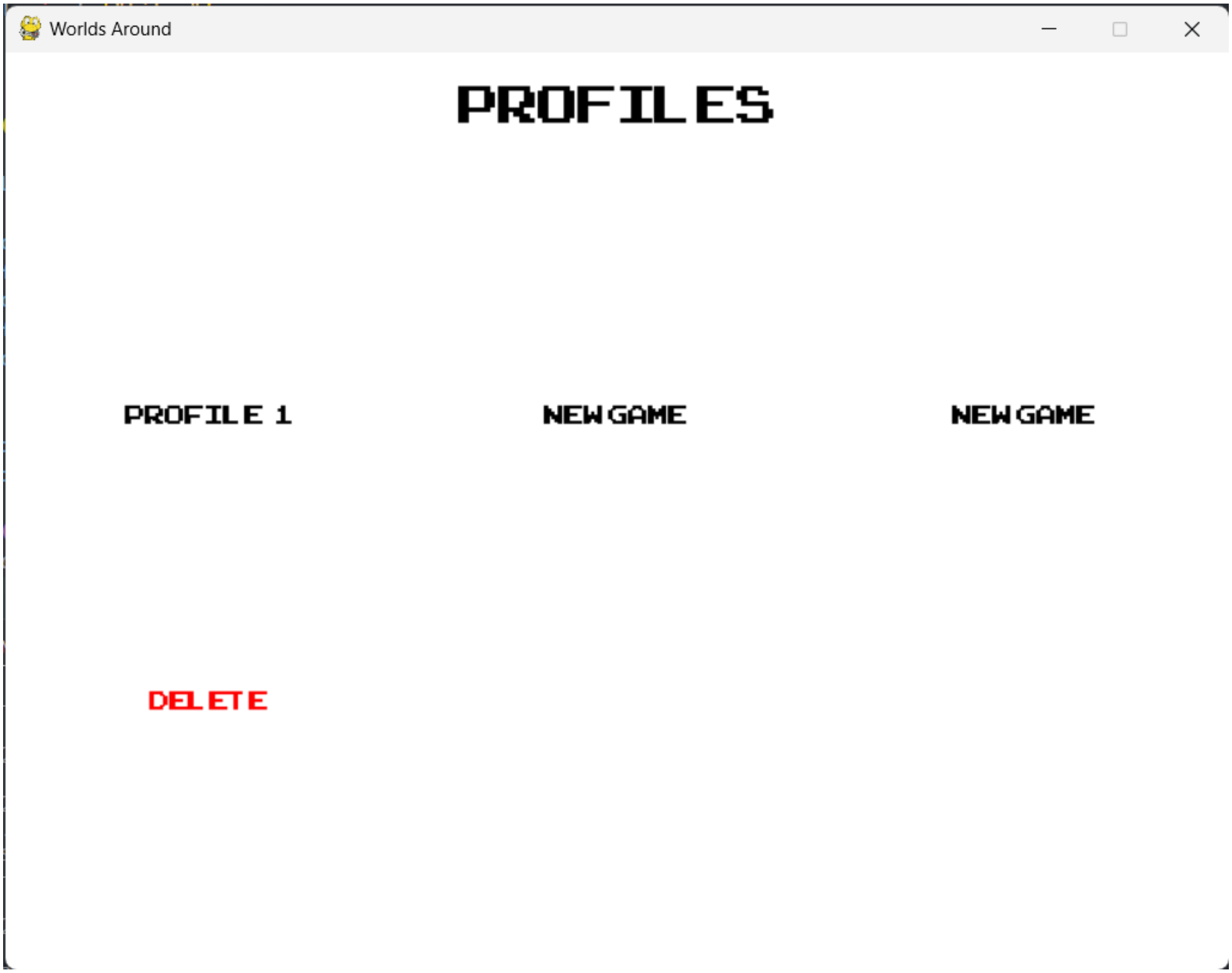
The dependencies of the project are kept to a minimum, to make sure that the code is tailored to the project's needs, which has the added benefit of performance that games aim to achieve. For this reason, the only dependency in the project is the pygame library. The library provides support for managing the window of the program, rendering to it, and notifying the program about events that are related to it.

The user interface of the game is provided in the code base, but it is built on top of existing UI paradigms. The UI is immediate mode (rather than the classical retained mode solutions) and the layout is built by using the rectcut algorithm.

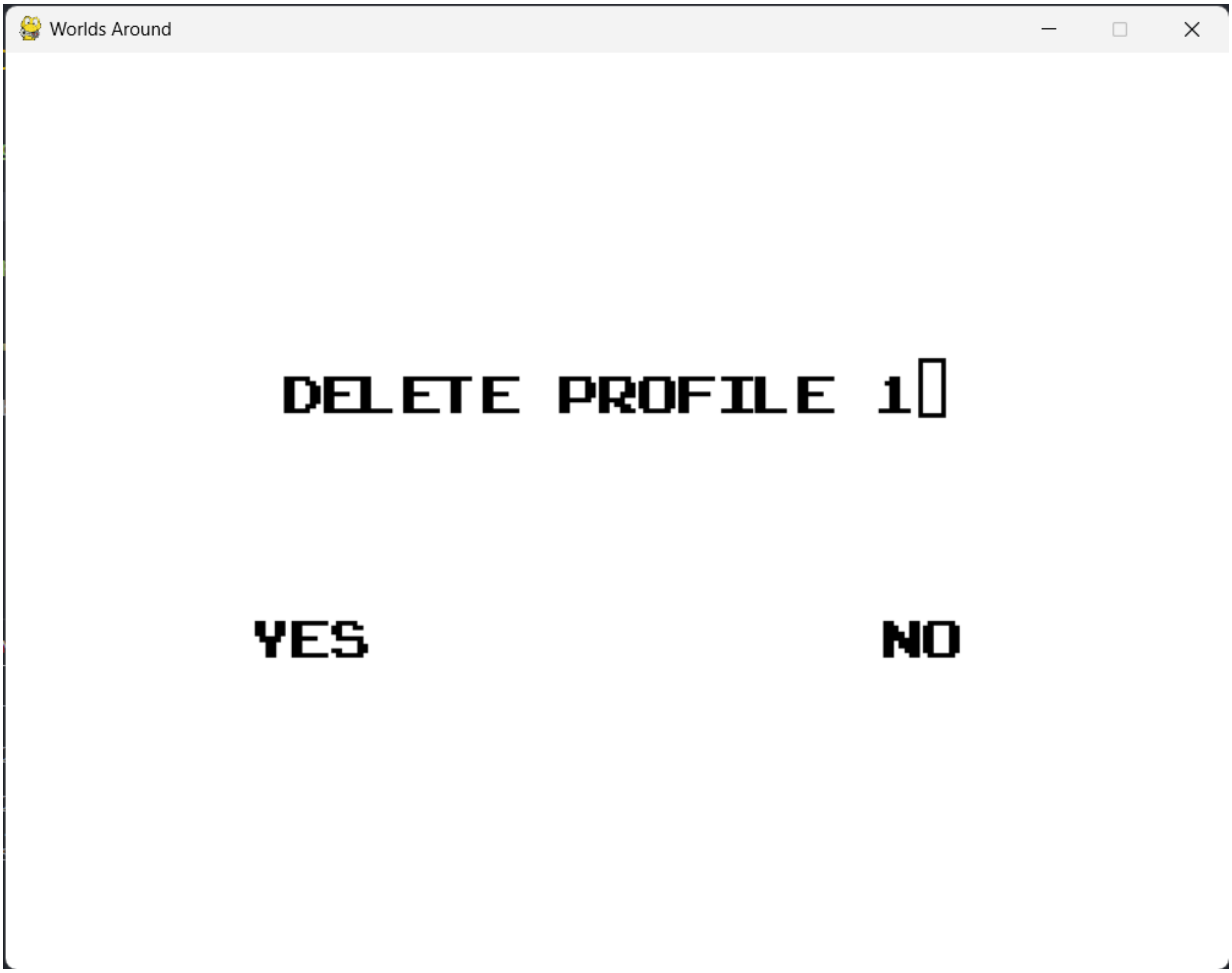
The levels of the game are stored in json files and are compounded of at most two files. First, there's the file that stores the layout of the objects that are part of the level; this file is known as the level file. The level file references another file, which maps each unique tile or object that can appear in a level to a number, also known as the tileset file. To reduce the size of the levels in memory, the tileset file can be reused across different levels.



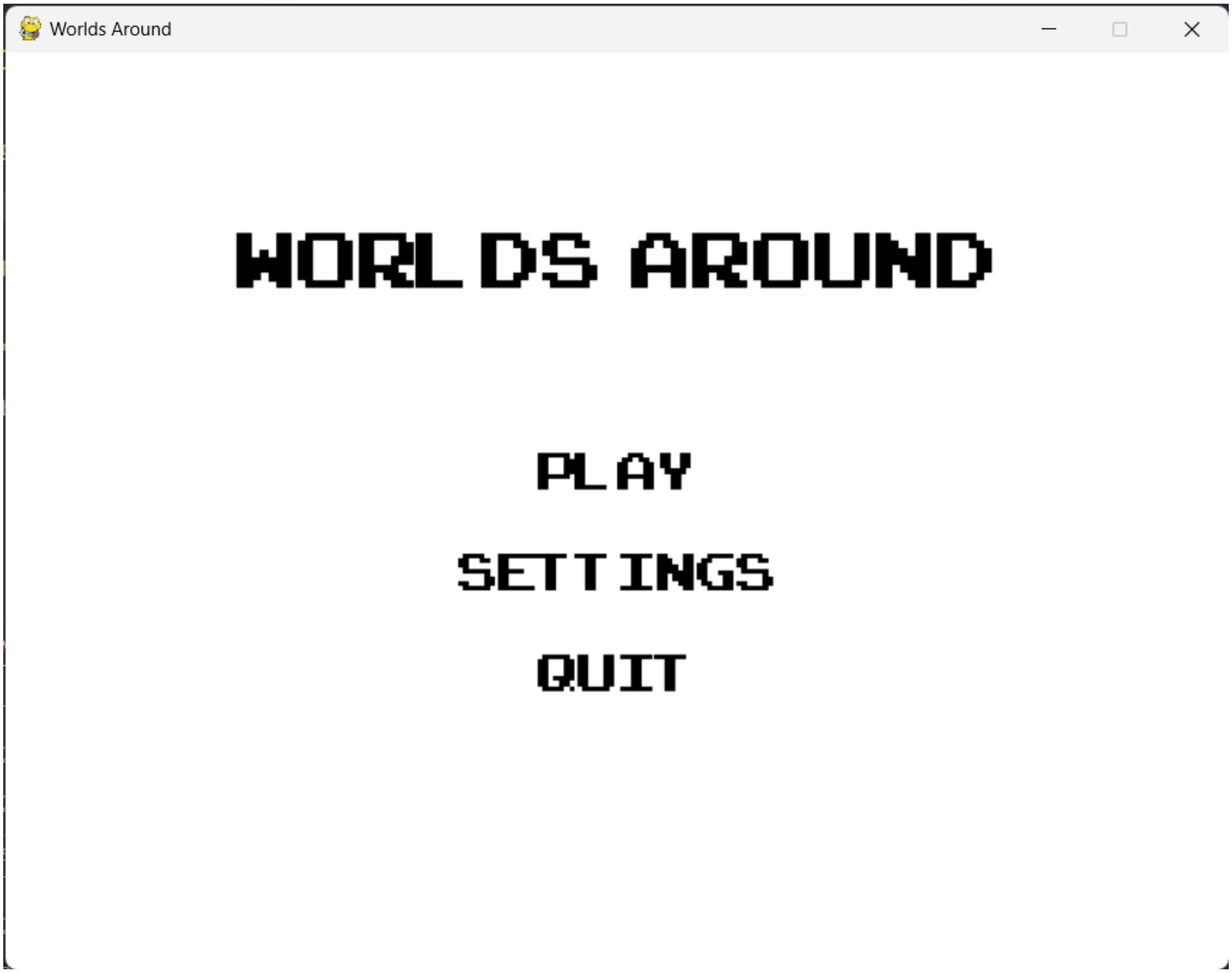
When the game starts, the profile selection menu is shown. If it is the first time that the game is opened, the user is asked to create a profile, which is as simple as picking a slot. The game keeps track of at most three profiles. When a new profile is created, a delete button is shown allowing the user to clear the progress from said profile.



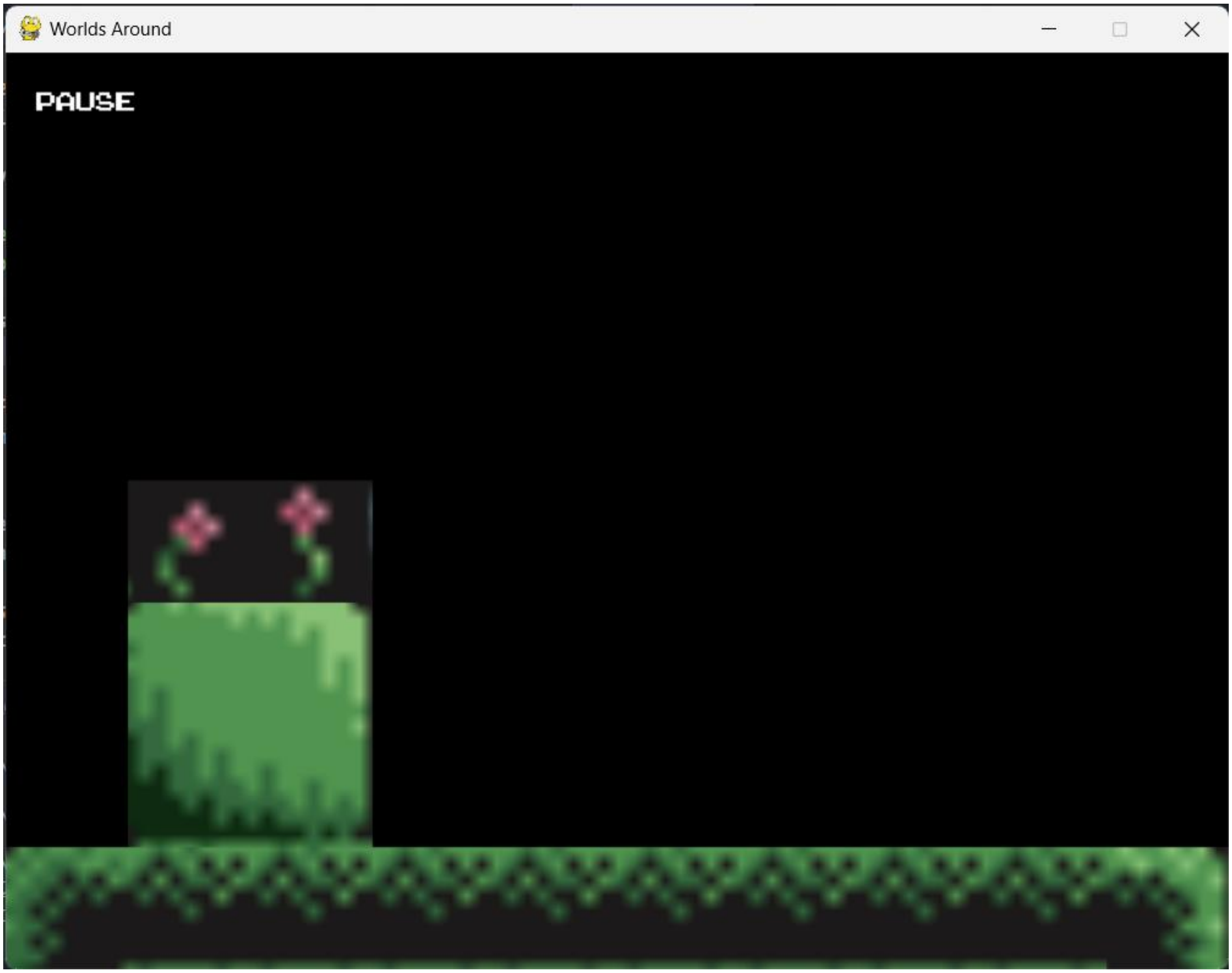
If the user wants to delete a profile, they can press the “DELETE” button under the respective profile’s name. To avoid possible mistakes, the user is asked if they are sure that they want to delete their profile, as shown below.



If “YES” is chosen, the profile is deleted and the slot is freed, if “NO” is chosen then nothing happens. In both cases, the user is redirected to the profile selection menu. From there, the user can select which profile to play by simply selecting the name of the profile, which redirects the user to the main menu.

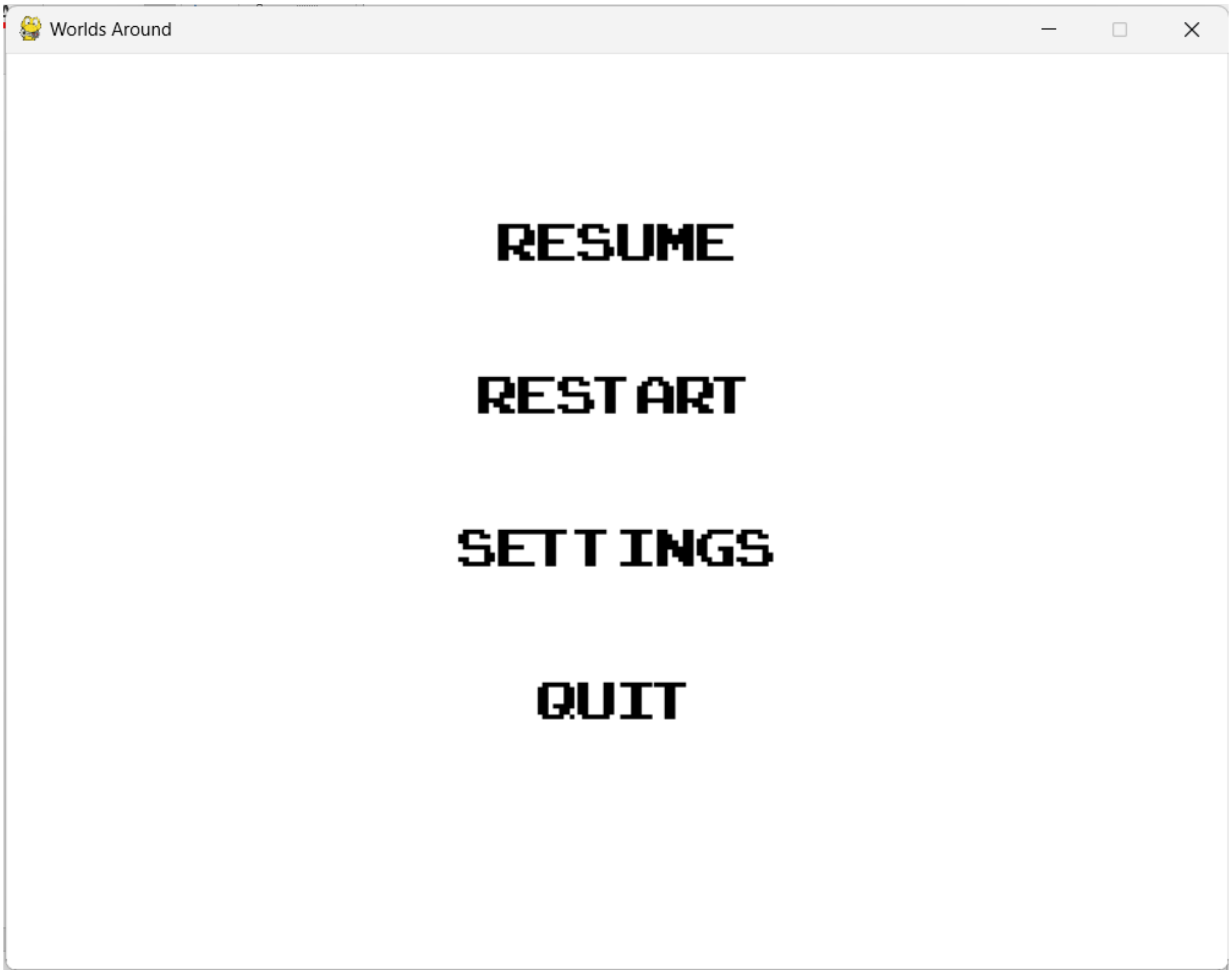


In the main menu, the user can choose to play from where they left, go to the settings menu, or to simply close the game. If the "PLAY" button is pressed, the earliest level that the user hasn't finished is loaded. As mentioned before, maps representing each level are stored locally in the Resources/Maps folder in json format. A "PAUSE" button is shown to allow the player to pause the game as needed.

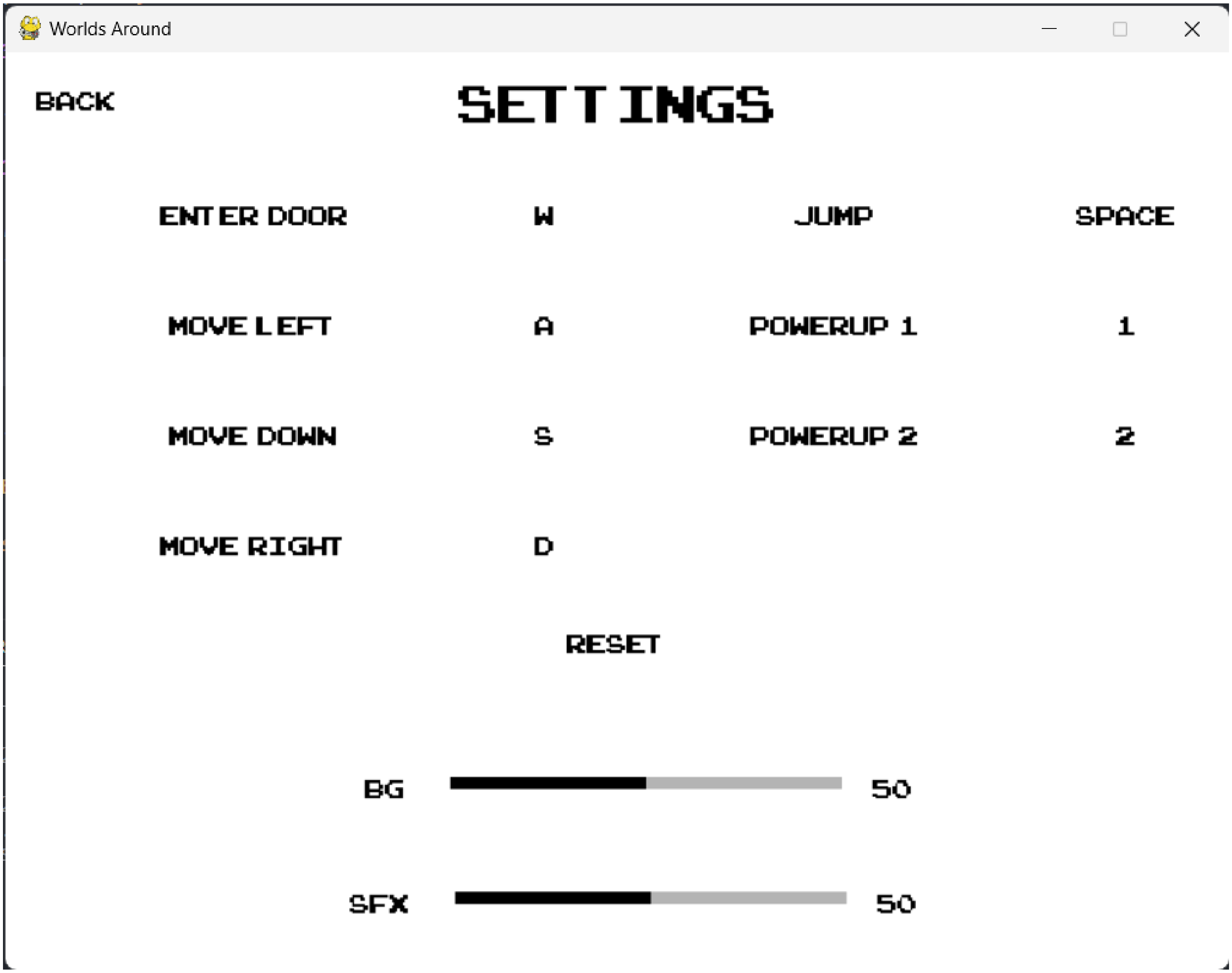


A sample level loaded.

When the “PAUSE” button is pressed, the game’s objects stop and the player is redirected to the pause menu. The menu bears resemblance to the main menu, but got its own functionalities as well. The first button, “RESUME”, simply resumes the gameplay from where it was left. The “RESTART” button restarts the current level from the beginning, as if the user just opened the level from the main menu. Furthermore, the “SETTINGS” button opens the settings menu, similar to the same button from the main menu. Finally, the “QUIT” button simply closes the game.



Finally, the settings menu is the most complex menu of the game. The menu allows to tweak configurations of the current profile. To allow for on-the-fly tweaking, this menu can be accessed both from the main menu (ie. Before gameplay starts) and from the pause menu (ie. During gameplay). Changes are immediately applied when the “BACK” button is pressed and the settings menu is pressed.



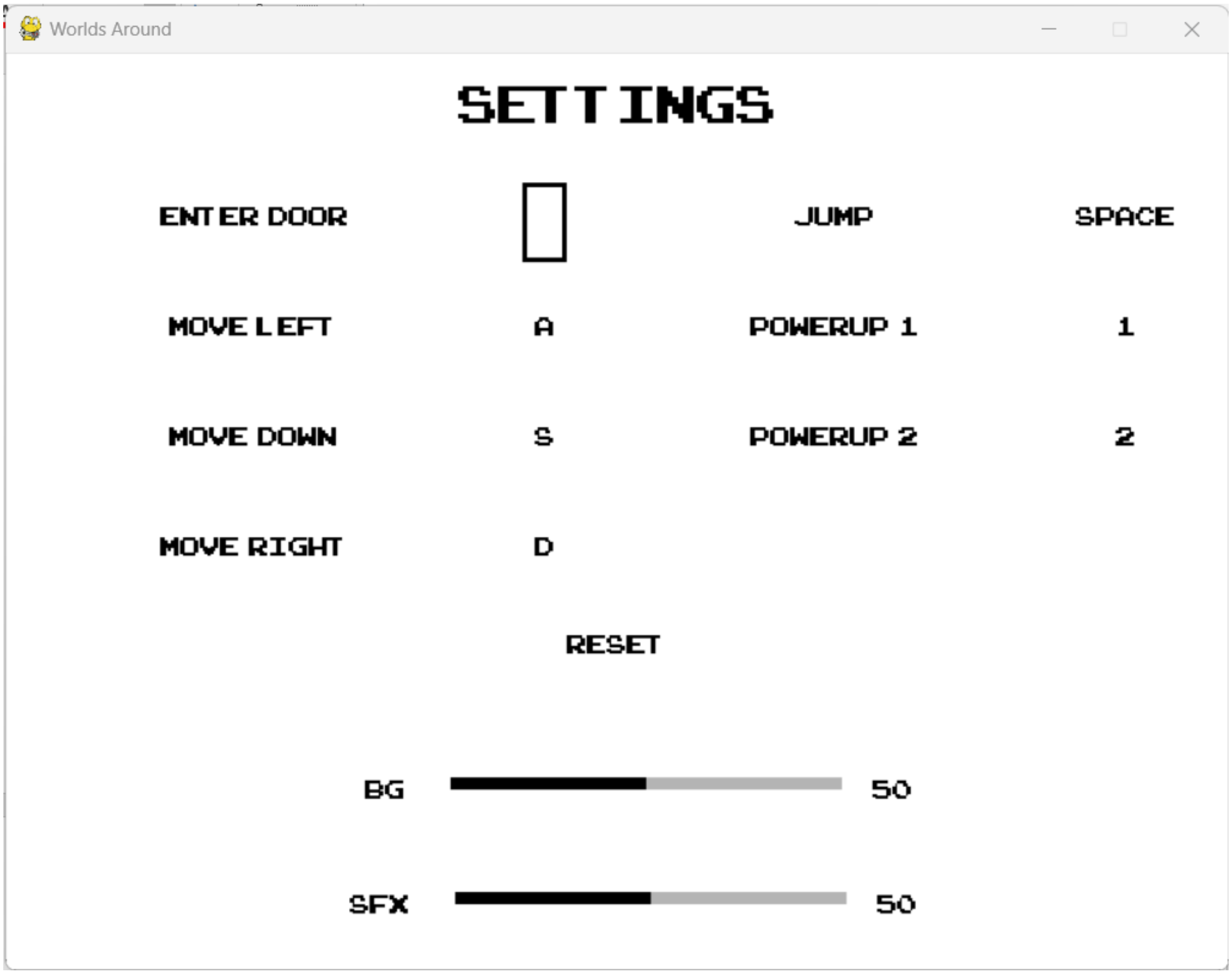
SETTINGS

ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

BG 50

SFX 50

On the upper half, the user can modify the keys that control the player. Each of the actions can be remapped by first clicking on the current key assigned to the action. When the key name is clicked, it is replaced by a black-bordered box, to denote that the current action is being remapped. To avoid potential bugs, the “BACK” button is not shown while an action is being remapped.



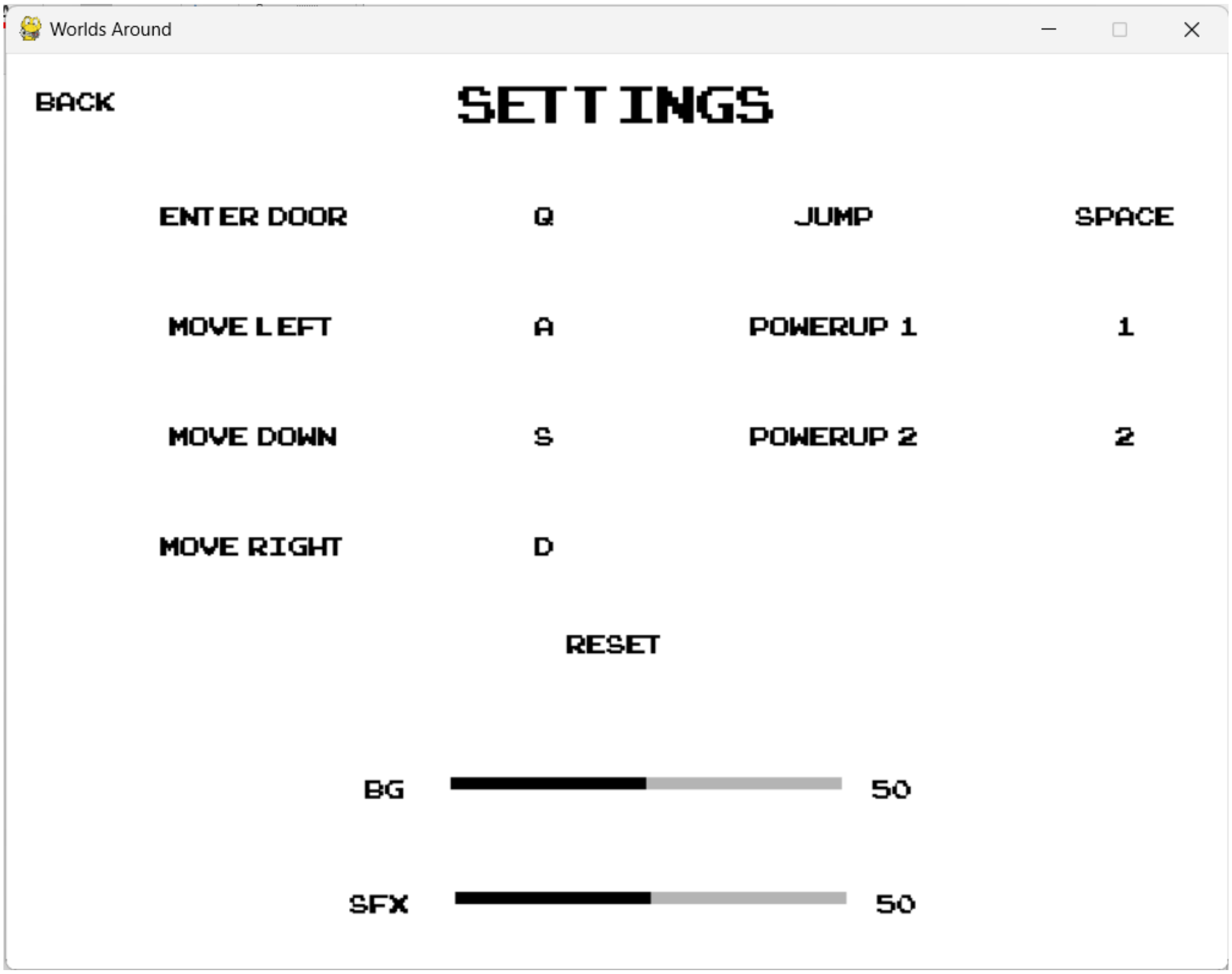
SETTINGS

ENTER DOOR	□	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

RESET

BG 50

SFX 50



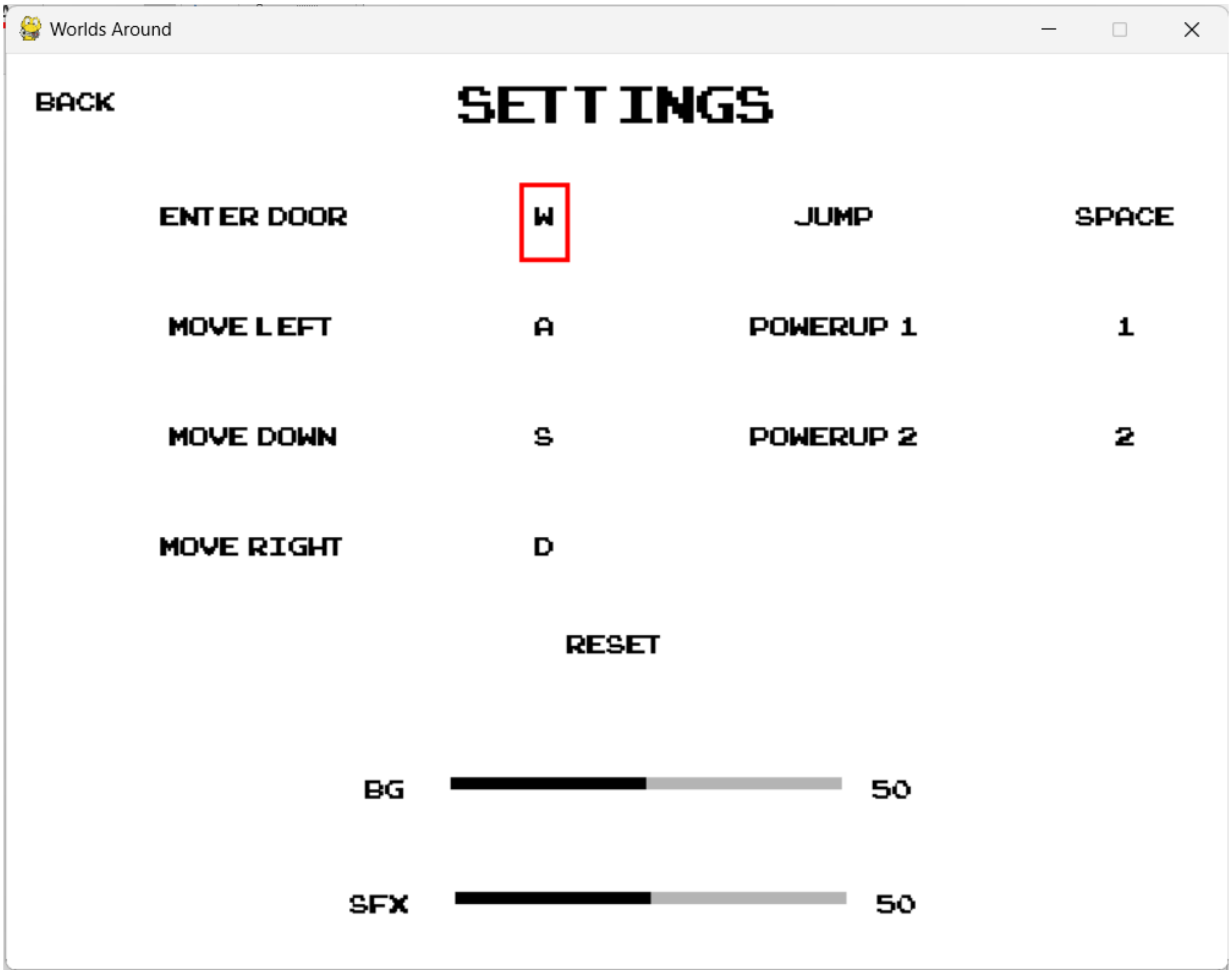
ENTER DOOR	Q	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

BG 50

SFX 50

Remapping the "enter door" action to a new key.

When the user presses a key from the keyboard, it will assign the pressed key to the action, if the key is not used by other actions. Otherwise, the old key will be reassigned to the action and the border of the rectangle in the near the action will become red. This is done to denote that the key is already in use by some other action.



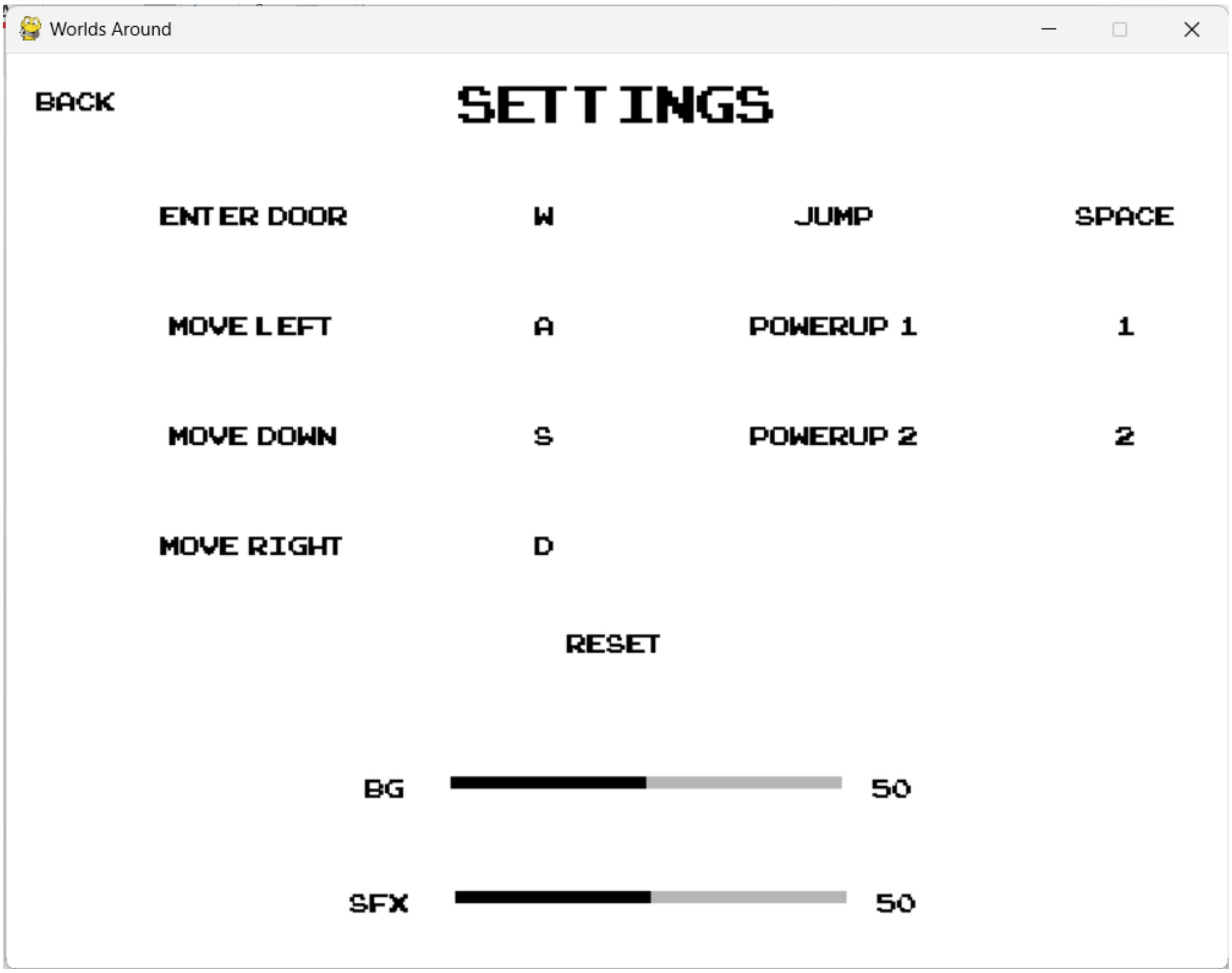
ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

BG 50

SFX 50

Trying to assign a used key to the "enter door" action results in an error.

Further down, the "RESET" button resets all the settings that can be tweaked (controls + volume) to their default state, which are shown below.



SETTINGS

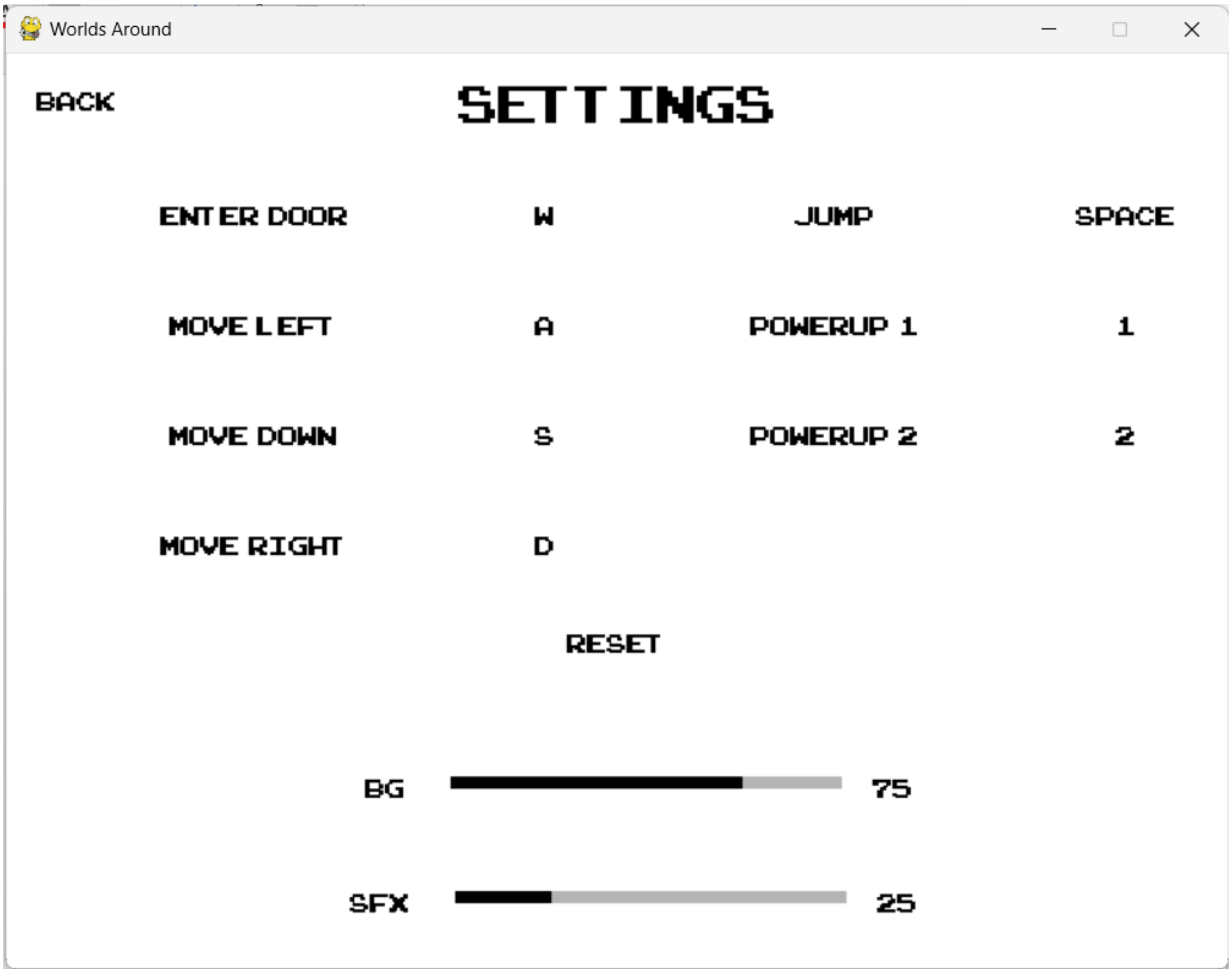
ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

BG 50

SFX 50

The default settings for every profile.

Finally, the user is also free to tweak the volume of the background music and sound effects from 0 (silent) to 100 (full volume).



SETTINGS

ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

BG 75

SFX 25

Tweaking the background volume and sound effects volume.

6 Project Planning

Project Name: Worlds Around

Start Date – End Date: 11 March 2022 - 10 June 2022

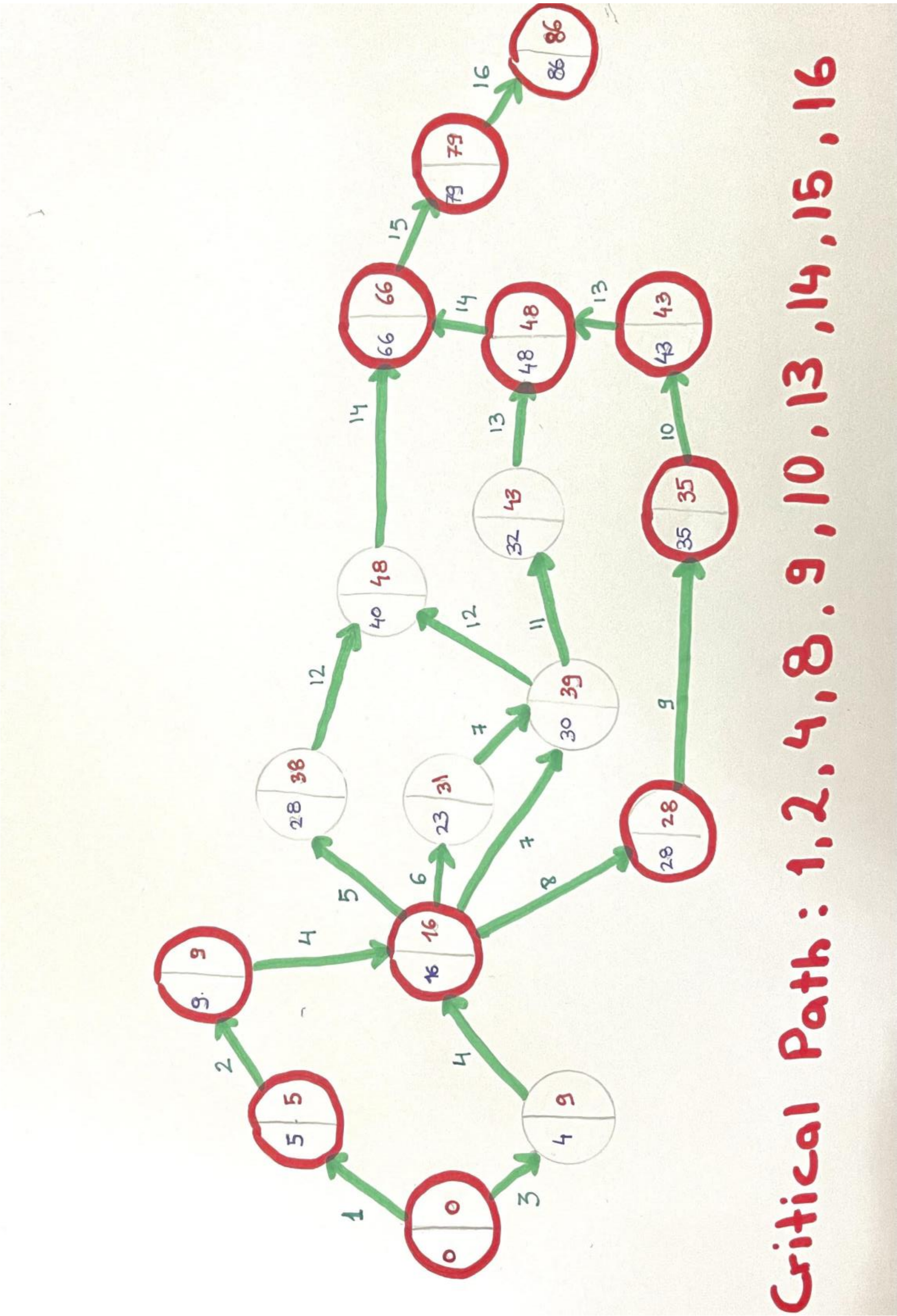
Estimated End Date: 5 June 2022

Total Duration (Real): 91 days

Total Duration (Estimated): 86 days

Activity ID	Activity	Duration (days)	Depends on	OT (days)	PT (days)
1.	Project Ideas	5			
2.	Research & Idea Flow	4	1		
3.	Project Management and work methodology	4			
4.	Requirements	7	2,3	6	9
5.	Prototype	12	4	10	13
6.	Use cases & User Scenarios	7	4		
7.	ERD	7	4,6	5	8
8.	Behavioral Diagrams	12	4	10	14
9.	Interaction Diagrams	7	8		
10.	Structural Diagrams	8	9	6	9
11.	Game Organization	2	7		
12.	UI	10	5,7	9	13
13.	Backend	5	10,11	3	7
14.	Coding	18	12,13		
15.	Testing & Refactoring	13	14	12	16
16.	Documentation	7	15	5	9
	TOTAL	128		113	145

6.2 Network Analysis



Critical Path: 1,2,4,8,9,10,13,14,15,16

7 Appendix

7.1 Appendix A- Definitions, Acronyms and Abbreviations

- AD_## - Activity Diagram followed by a number
- CD – Class Diagram
- CoD – Component Diagram
- ColD_## – Collaboration Diagram followed by a number
- DD – Deployment Diagram
- DFD_## - Data Flow Diagram followed by a number
- ERD – Entity Relation Diagram
- FR_## - Functional Requirement followed by a number
- OD – Object Diagram
- PDF – Portable Document Format
- PMS – Polyclinic Management System
- PS – Print Screens
- SD_## - State Diagram followed by a number
- SeqD_## - Sequence Diagram followed by a number
- UC_## - Use Case followed by a number
- UI – User Interface
- US_## - User Scenario followed by a number

7.2 Appendix B- References

www.pygame.org/docs/

www.halt.software/dead-simple-layouts/

www.solhsa.com/imgui/

www.gameprogrammingpatterns.com/object-pool.html

www.gameprogrammingpatterns.com/component.html

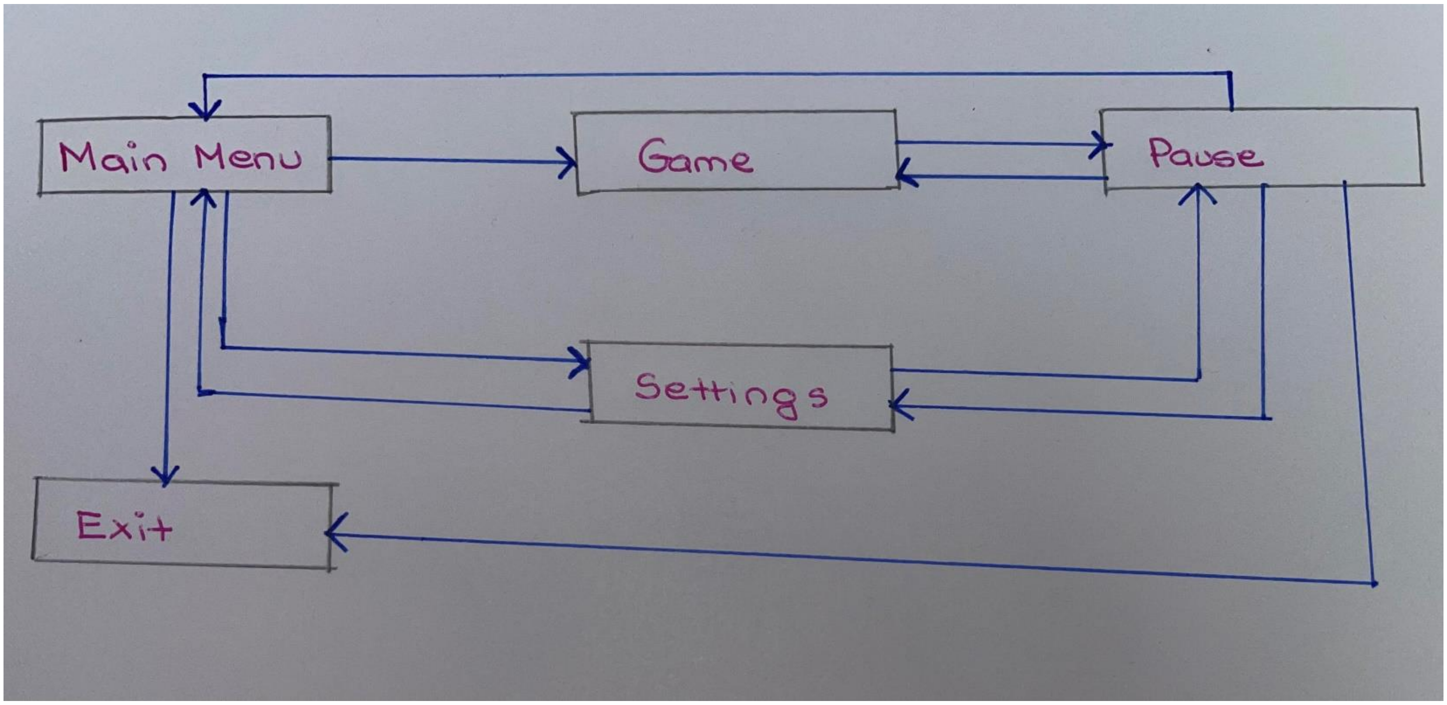
www.gameprogrammingpatterns.com/command.html

www.gameprogrammingpatterns.com/game-loop.html

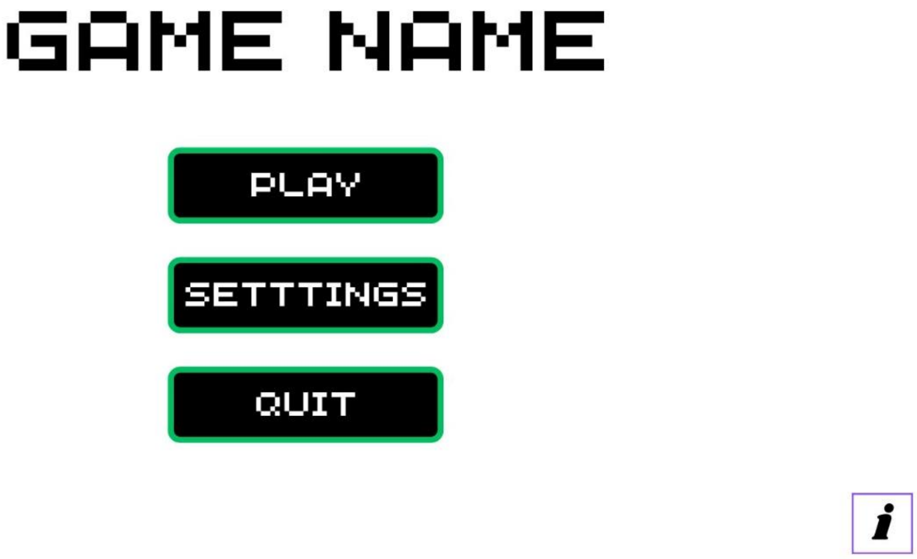
www.plantuml.com

www.json.org/json-en.html

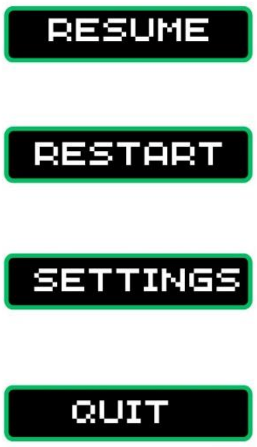
7.3 Appendix D- Sketches



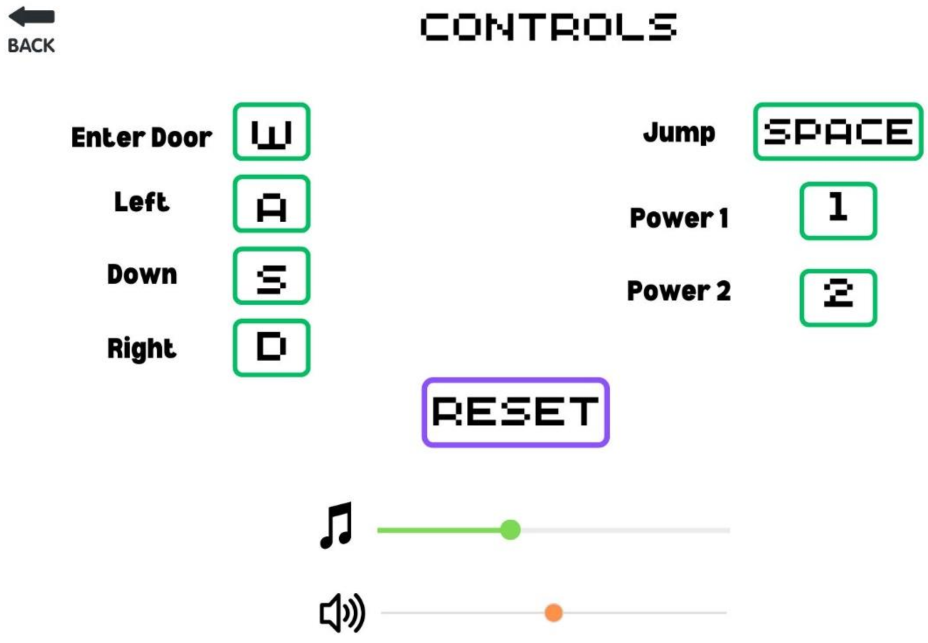
Game Organization


GAME NAME
PLAY
SETTTINGS
QUIT
i

Main Menu


RESUME
RESTART
SETTINGS
QUIT

Pause Menu



CONTROLS

Enter Door W

Left A

Down S

Right D

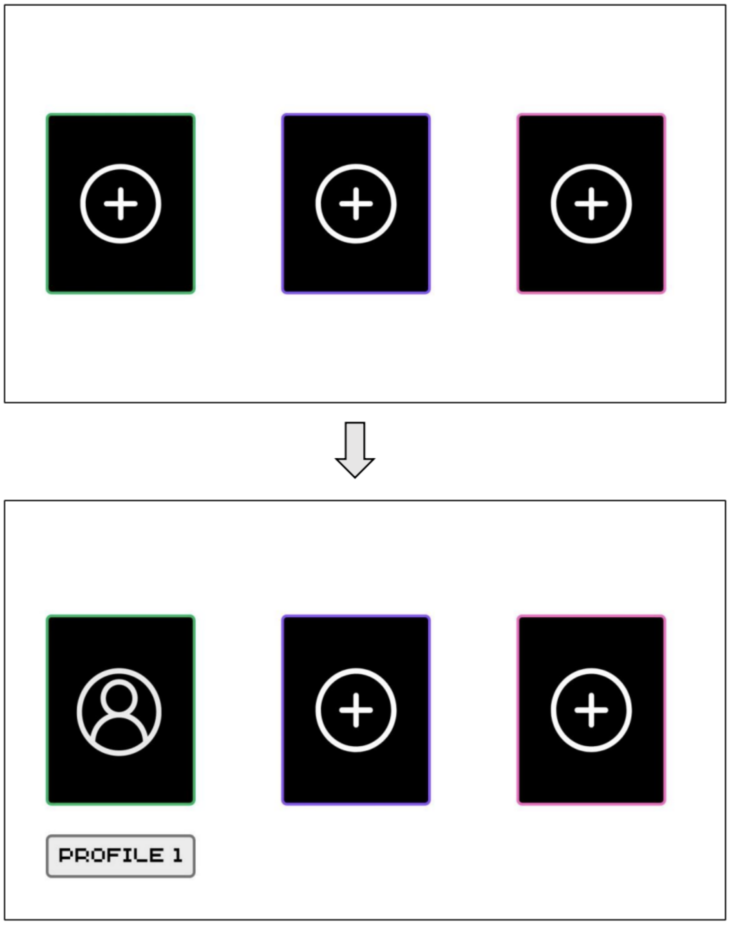
Jump SPACE

Power 1 1

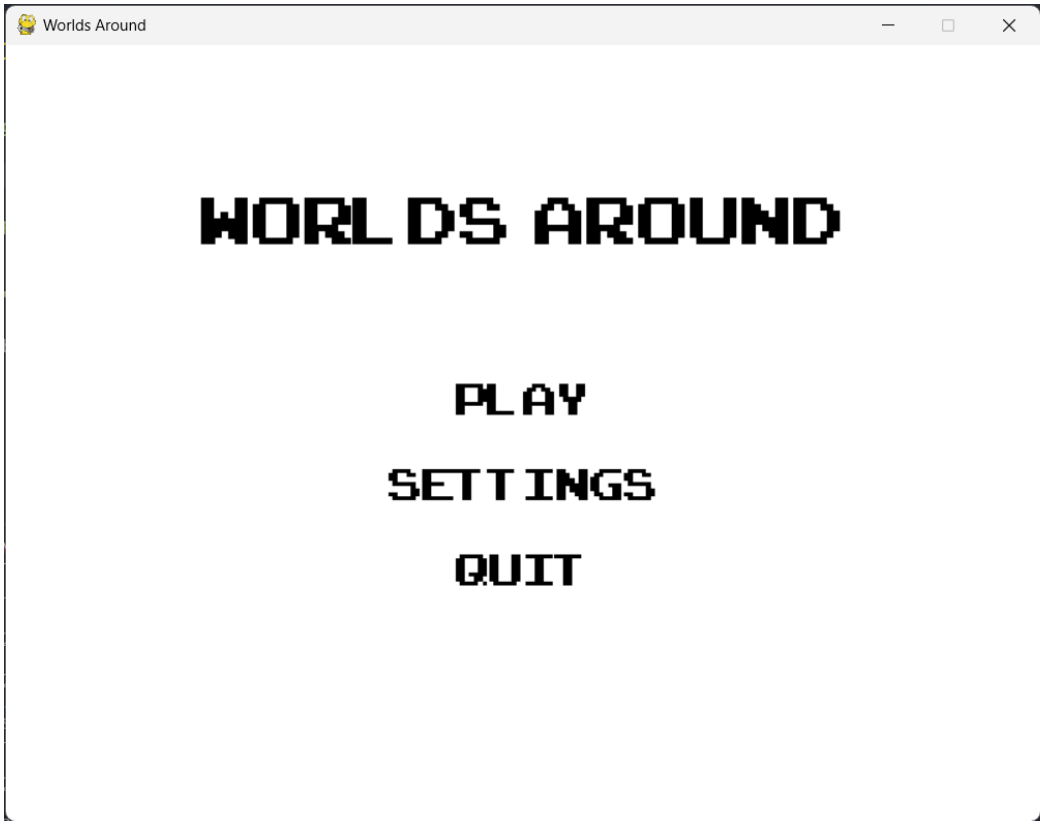
Power 2 2

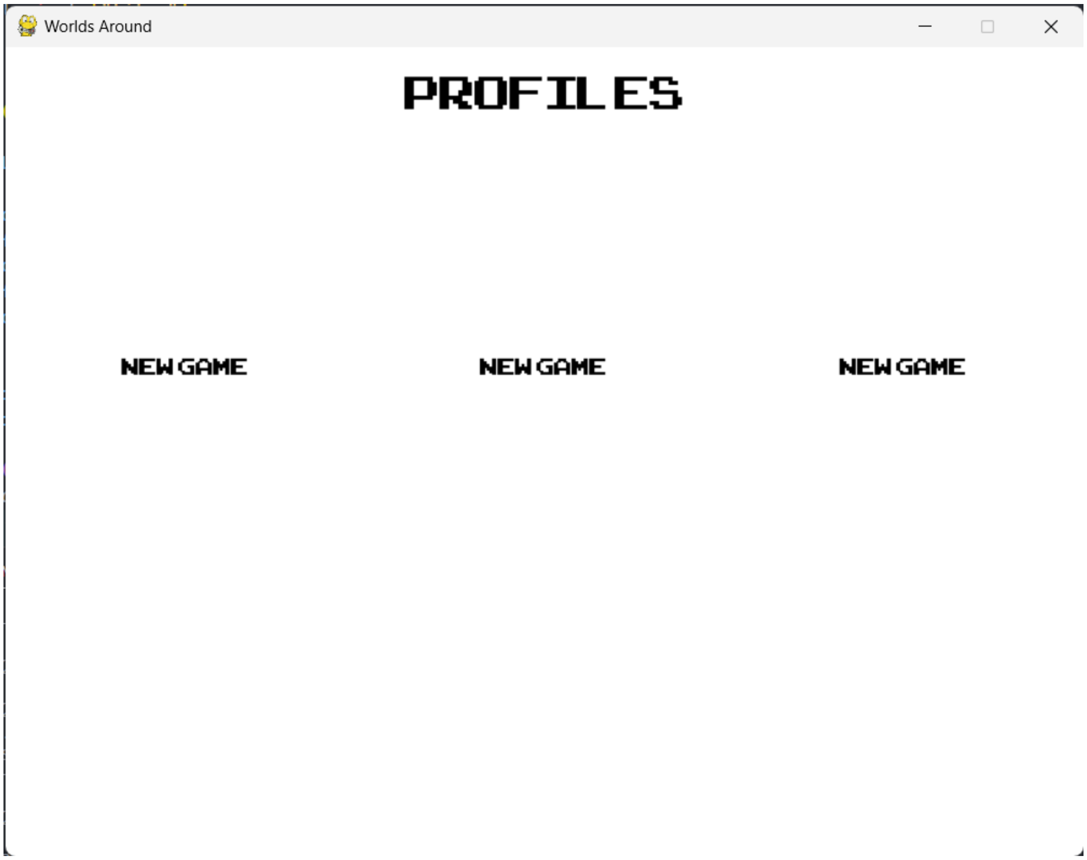
Settings Menu

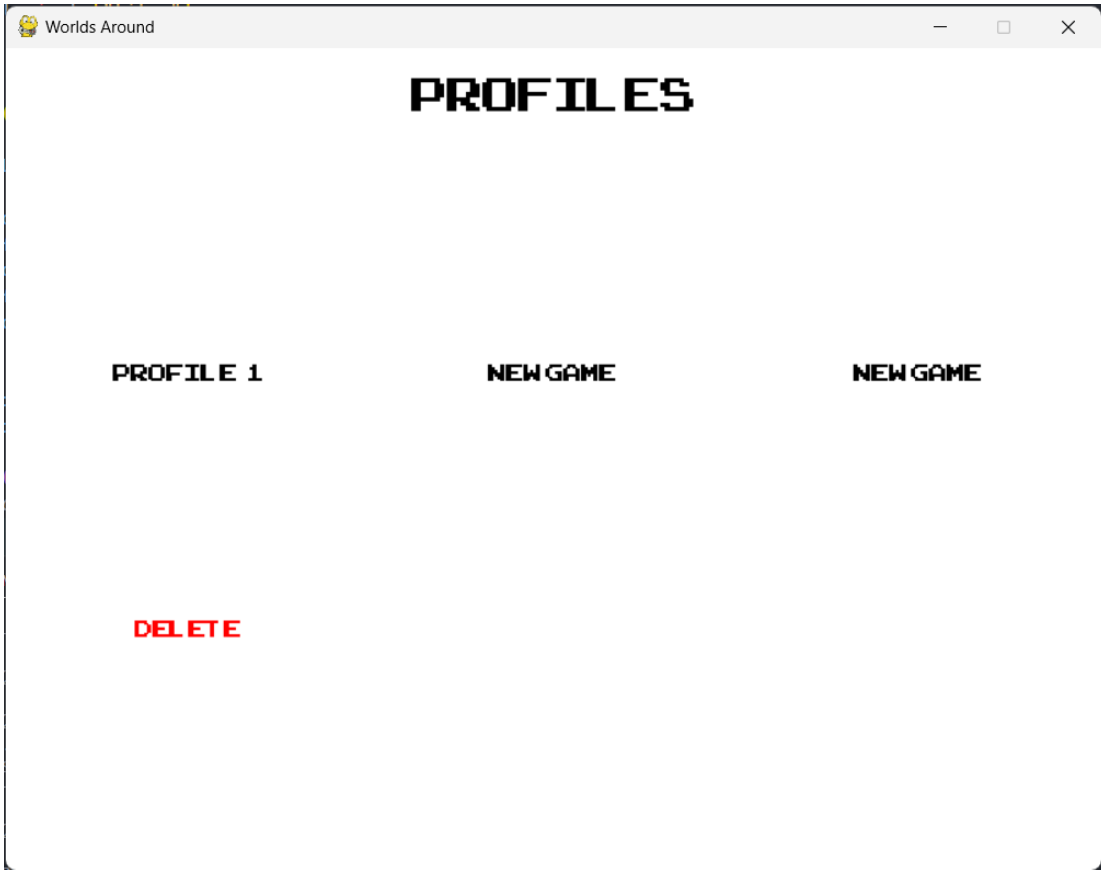
ADD PROFILE PAGE

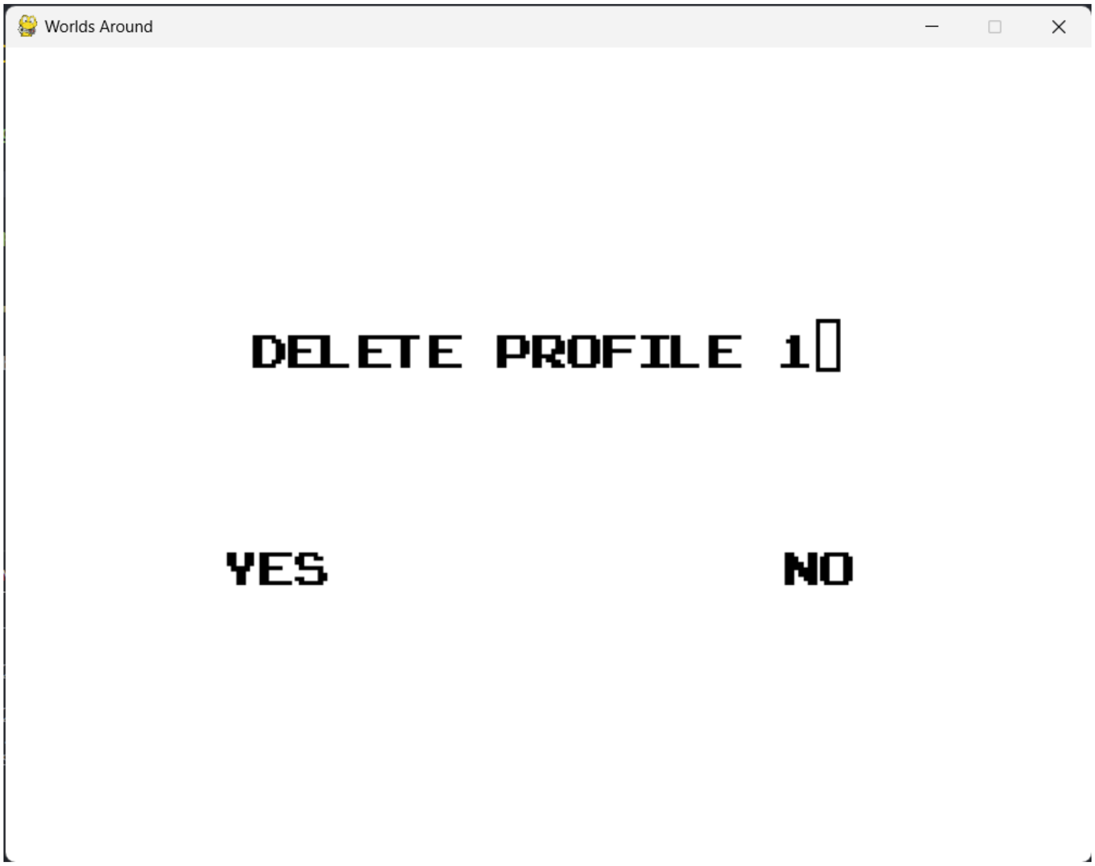


7.4 Appendix E- Detailed Designs

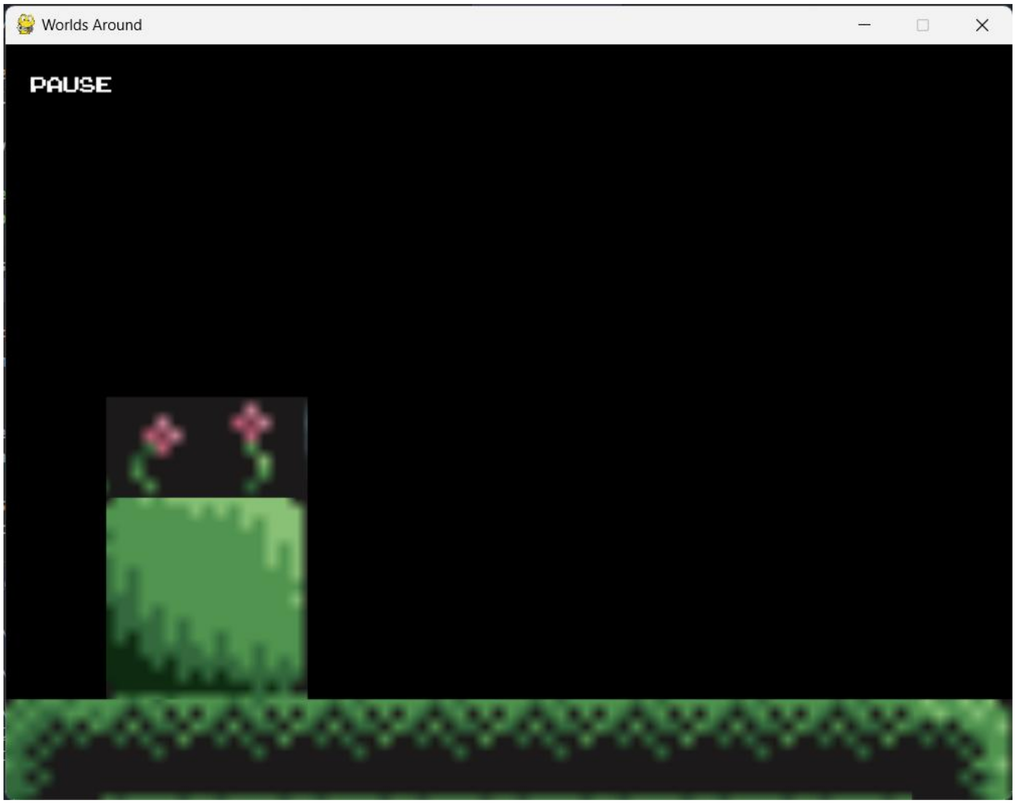


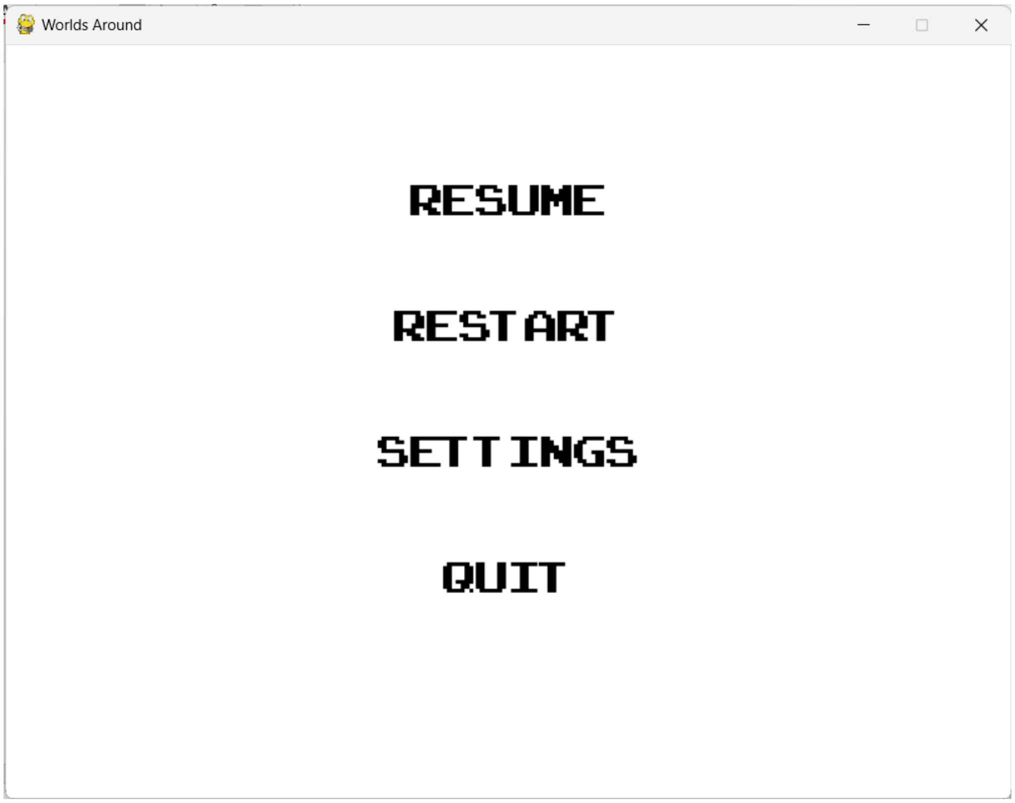


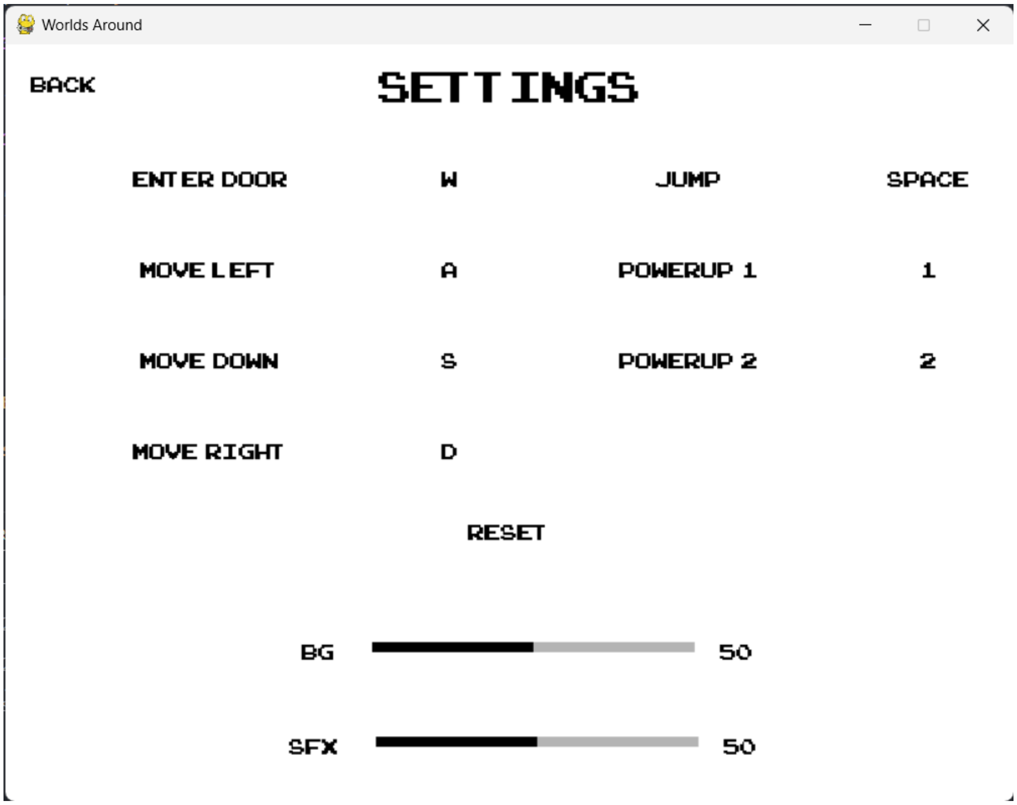




DELETE PROFILE 1▯


Worlds Around
PAUSE


Worlds Around
RESUME
RESTART
SETTINGS
QUIT



BACK

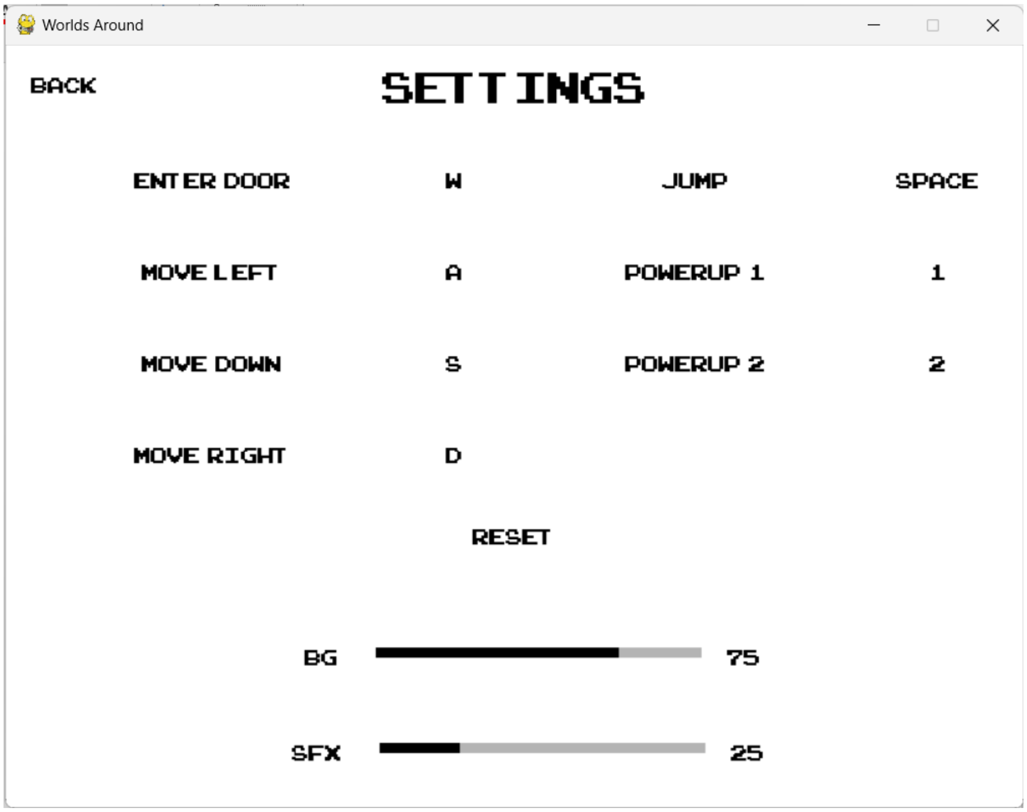
SETTINGS

ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

RESET

BG 50

SFX 50



BACK

SETTINGS

ENTER DOOR	W	JUMP	SPACE
MOVE LEFT	A	POWERUP 1	1
MOVE DOWN	S	POWERUP 2	2
MOVE RIGHT	D		

RESET

BG 75

SFX 25