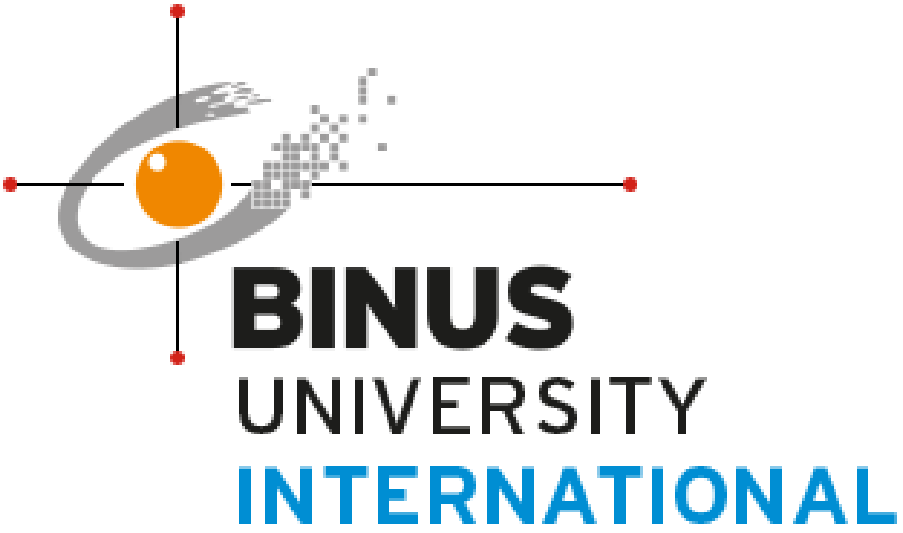


ALGORITHM AND PROGRAMMING FINAL PROJECT

(SAMURAI DASH)

Made By :

Harris Ekaputra Suryadi

Binus University International Program Computer Science - 2802400502

Lecturer :

Jude Joseph Lamug Martinez, MCS

BINUS INTERNATIONAL UNIVERSITY

2024/2025

TABLE OF CONTENTS

CHAPTER 1

Project Specification

1.1 Project Description

For my algorithm and programming final project, I made a platformer game that has 3 levels on it. To go to the next level you need to eliminate all the enemies on the map. So the Samurai (Player) needs to defeat or eliminate all of the sorcerers on the map to proceed to the next one, and each level the numbers of the sorcerers will increase.

The Samurai can move up, down, left, and right by pressing the keybind (W(up), A(left), S(down), D(right)), he also has the ability to dash so he can easily roam around the map and search for the enemies. Dashing also can be used to eliminate the enemy by dashing into them, so the Samurai needs to dash into the sorcerer to eliminate them.

The Sorcerers also have the ability to shoot projectiles at the Samurai. So whenever the Samurai is close to the Sorcerer it will shoot the projectile, and the projectile is one hit to the Samurai. So the Samurai will not eliminate the Sorcerers easily, because there's still a resistance from the Sorcerer to shoot projectiles at him.

1.2 Project Link

I publish my project at github, here's the link :

https://github.com/HES2209/FinalProject-AlgoProg.git

1.3 Essential Algorithm

1. Initialization :

- Importing Modules : Importing some necessary modules and classes from other files for the game to run (pygame, math, sys, random, os).
- Setting up the display screen : Create a window for the game, set the name and fps for it, and set it in a certain resolution.

2. Loading Assets :

- Load images : Load a single or multiple images for the elements of the game (Player, Enemy, Decorations, Background, Tiles).
- Load animations : Create a frame by frame movement by putting the image one by one per frame that has its own scaling time, that is applied to the player and enemy, so it looks like they are moving smoothly.
- Load sound effects : Load the sound effects from the destined file and set the volume for each sound effect audio.

3. Creating The Game Objects :

- Initialize the main objects and entities : Display the elements or components inside the game world, such as the players, the enemy, and the map.
- Create clouds : Import the clouds class from another file to display the clouds on the game world, and also set the position, layering, and size for each cloud.
- Create tilemap : Import the tilemap class from another file to display the map for the game. The size, physics, and position of each tile is already being set and determined.
- Create the player : Initialize the Player sprite and display it to the game that has its own position and size, which has been determined.

- Create levels : Set and load the data for each map or difficulty of each level, and also set the max level for the game. After all of the levels are completed, it will initialize the win screen to be displayed whenever all of the levels are completed.

4. Game Loop :

- Game logic :
    - Player actions : The player can move up, left, and right, the player also can jump and has the ability to dash to eliminate the enemies.
    - Enemy actions : The enemy will just walk around the map and it has the ability to shoot to eliminate the player.
    - Collisions : It will detect the collisions whenever the player or the enemy walk into a wall, so it won't go through the wall but it will have collision to it. Also the hit box for the projectiles and the dashing ability to detect if it hits it will eliminate the player or the enemy.
- Update and render the entities : Keep updating the position of the entities and components in the game and render them (The player, enemies, projectiles, sparks, clouds, and particles).
- Check player condition : If the player is dead then the level will be restarted, if not it will keep looping the level until the player eliminates the enemies on the map.
- Check enemies : If all of the enemies are eliminated then go to the next level, and if it reaches the max level then display the win screen. And if there are still some enemies left on the map, then loop the same level until all of them are eliminated.
- Tick clock : Set the fps of the game loop, so it will constantly running on the determined fps.

5. Event Handling :

- Pressed key events (Key Down) : Handle the events from the user input whenever they press a key down :
    - Move left whenever the ‘A’ key is pressed
    - Move right whenever the ‘D’ key is pressed
    - Jump whenever the ‘W’ key is pressed
    - Dash whenever the ‘Shift’ key is pressed
- Unpressed key events (Key Up) : Handle the events from the user input whenever they don’t press a key :
    - Stop moving left whenever the ‘A’ key is not pressed
    - Stop moving right whenever the ‘D’ key is not pressed

1.4 Modules Usage

1. Pygame Module :

The purpose of using this module is to create a game, pygame will allow us to access more code libraries that are related to creating a game, so this module will be a big help for us for creating a game.

The usage of this module is to create a game window, handle events from the user inputs, and also render graphics.

2. Sys Module :

The purpose of using this module is to provide functions and variables that can be used to control different parts of the python runtime environment.

The usage of this module is to automatically close the game whenever the user closes the window.

3. Random Module :

The purpose of this module is to provide a function that can generate random numbers.

The usage of this module is to generate a random number for the movement of the enemies and the particle effects on the game, so there's not going to be a pattern, but it's fully random.

4. Math Module :

The purpose of this module is to provide functions for mathematical tasks.

The usage of this module is to do calculations on the physics of the game, such as collisions and also particle movements.

5. Os Module :

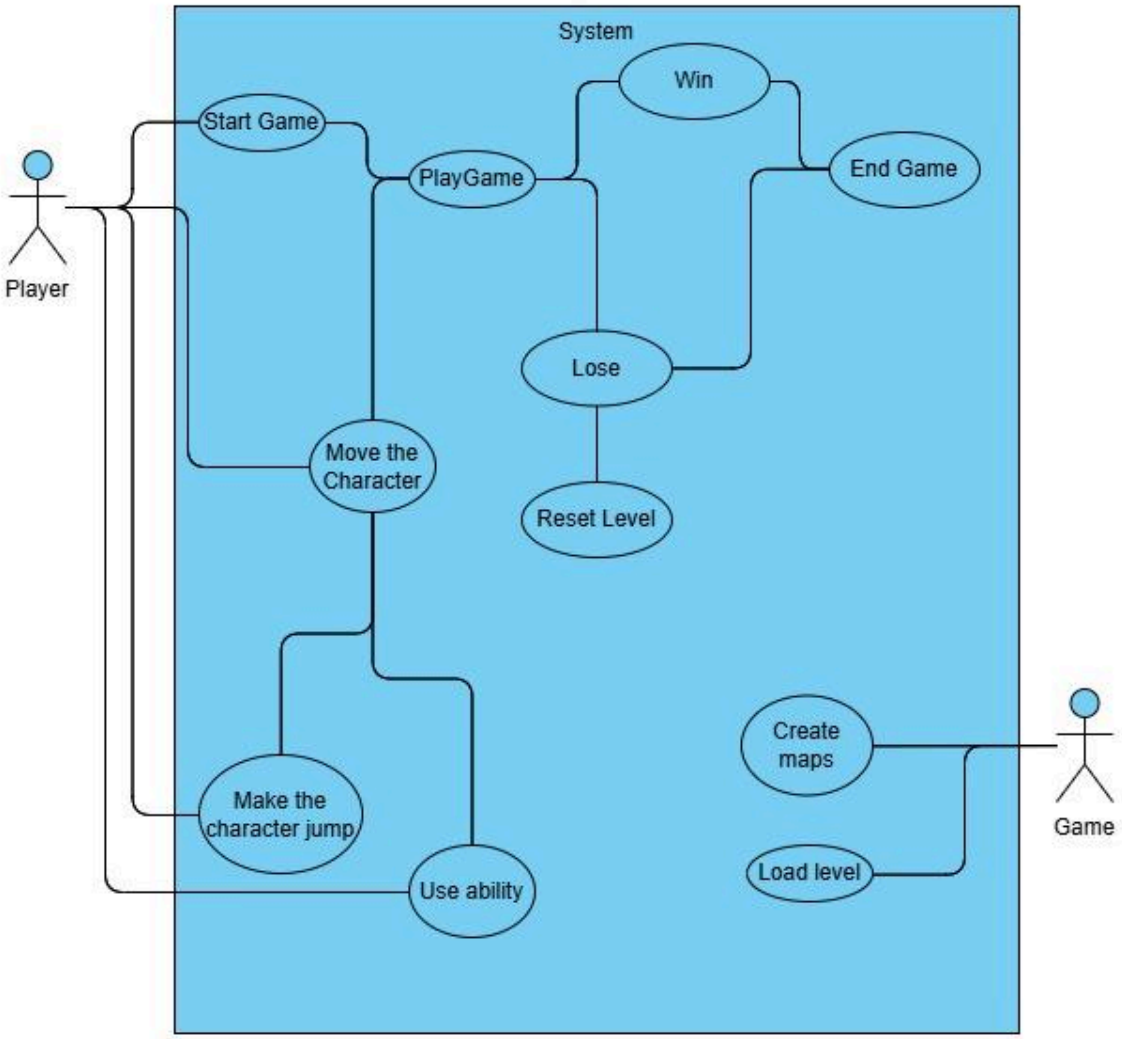
The purpose of this module is to enable us to interact with the operating system.

The usage of this module is to load files and assets for the elements of the game.

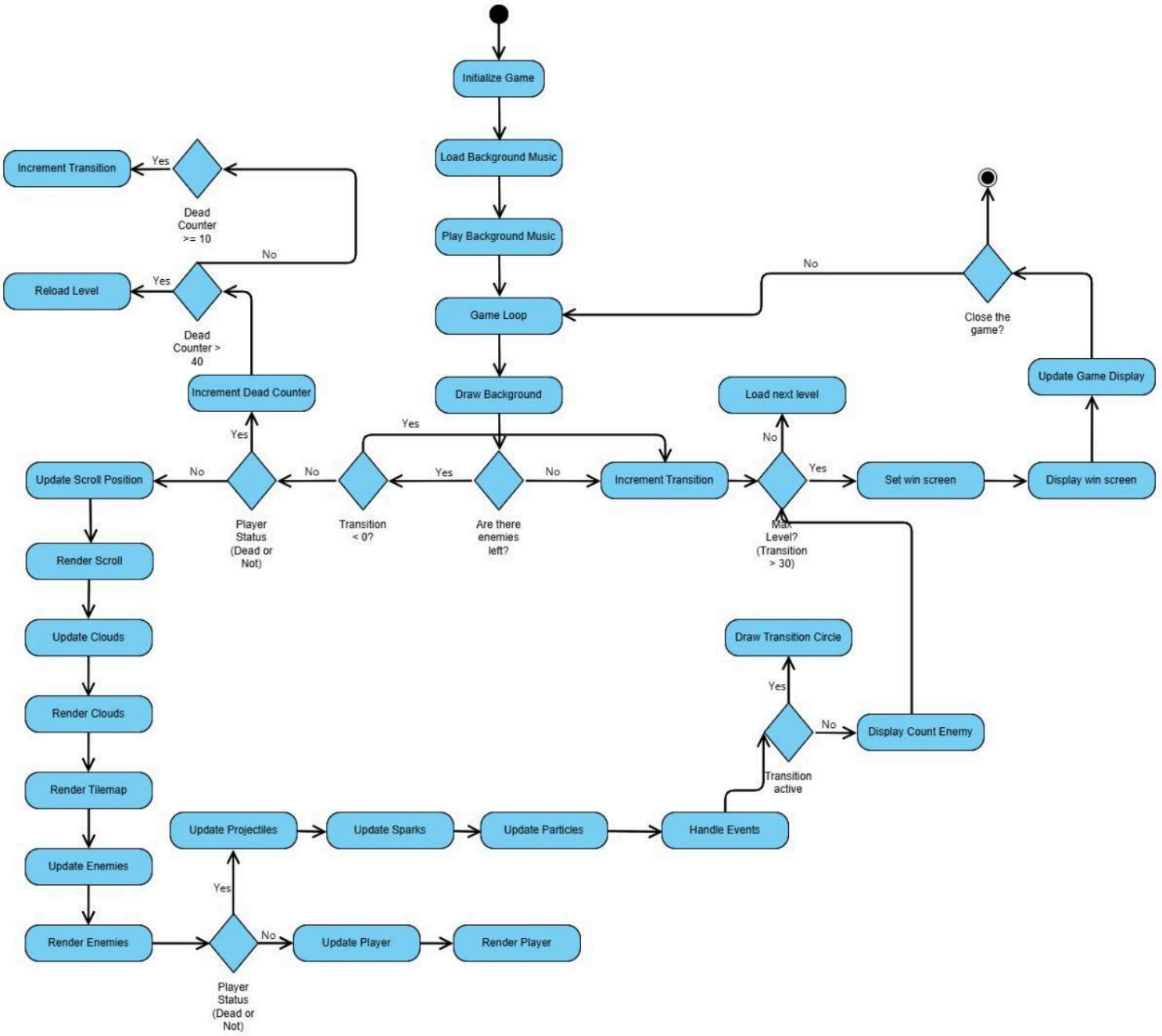
CHAPTER 2

Solution Design

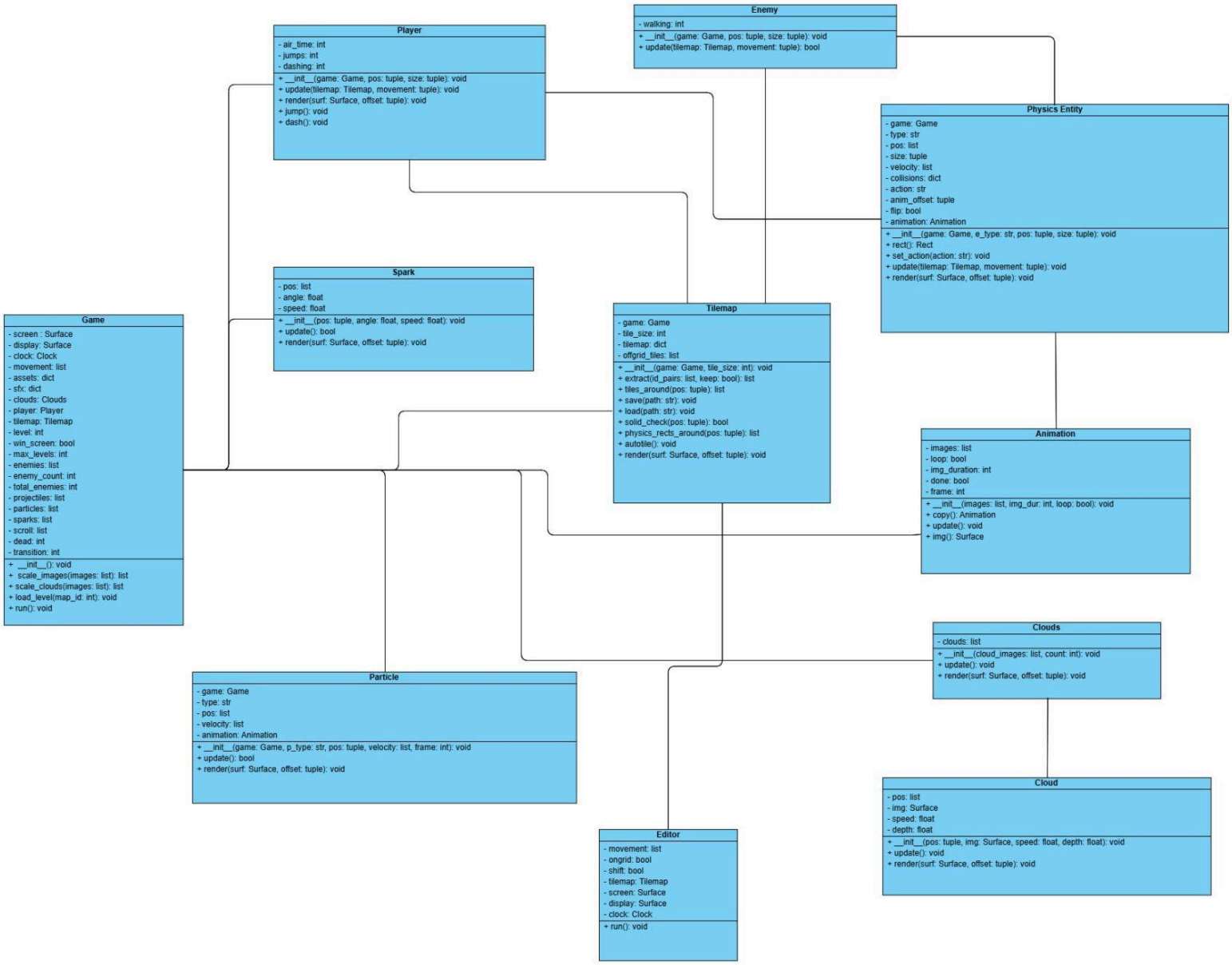
2.1 Use Case Diagram



2.2 Activity Diagram



2.3 Class Diagram



CHAPTER 3

Documentation

3.1 Screenshots

Level 1

Level 2

Level 3

Win screen

3.2 Video Demo

Here is the link of the video demo that I have recorded and upload it to Google Drive :

https://drive.google.com/file/d/1aTKiqxKs1ou5ulQWpxsKqohnXStutEbR/view?usp=sharing

3.3 References

To make this game I've been following a guide from youtube that teaches me how to make this game. Here is the link : https://www.youtube.com/watch?v=2gABYM5M0ww&t=20736s

I also use some assets from the guy for the map. And I really thank him because I have managed to create this game because of him.

Algorithm and Programming - Final Project Report

For the sprites I took it from this website : https://craftpix.net/categorys/sprites/ because it has a good design for character games.

CHAPTER 4

Evaluation And Reflection

4.1 New Things I Have Learned

From this project, I learned many things about coding and the logic about it. Mainly I could understand more about pygame and the other modules that I use. I learned to make animations, to manage files, to load assets, to create a window, and many other logics that I've learned from making this project.

On the other hand, I learned that coding is really hard, not everybody can code well. Maybe because they don't understand the logic, or it's just because it's not they're passion. Because to create a project like this could take a long time for someone to make, you can't do this in one day because it will just burn you out. It's because you will face many problems, such as bugs and errors in your code, which can be triggered by some simple mistakes, for example typos, indentation, or naming the variables, but there are still many mistakes that could trigger an error in your code.

That is why I learn how to be patient and never give up when making this project. Because I find so many errors and bugs, even the bugs that took days to debug when I'm making this project. Coding is not just typing, making something work, or debugging something, but it's also about being patient and never giving up when you are coding, because without that mindset, your code will never be done.

4.2 Future Improvements

I actually want to add some features to my game for this project, for example other types of enemies, bosses, more levels, and also more abilities for the player. But because of the lack of time I can't imply all of that to my game for this project, and also the code that is used to make the new features might be really complex than what I've learned, so it will take more time.

The things I'm planning to improve is to add more enemy types, bosses, and levels, such as skeleton guard, witch, ronin, and other types of enemies. And all of them will have their own special ability, which will improve the mechanics of the game. Also add levels, and every 5 levels that have been completed, the next level will be a boss level. The player has to fight the enemy that has more health, more attack moves, and more abilities.

Even though I don't have more time to imply and add all of that and other improvements, I'm still really grateful about the current project that I've done. Maybe it's not perfect yet, but as long as it works and it fits what I want, I'm happy with it. I hope in the future I can make a better project than this with more features and improvements.