



# **Final Project**

## **Algorithm and Programming**

Project Name: “CyberSafe”

Student Name: Kevin Makmur Kurniawan

Student ID: 2802547553

Class: L1CC

**BINUS University International**

**Jakarta**

**2024**

# Table of Contents

<b>Project Specifications.....</b>	<b>3</b>
A. Introduction.....	3
B. Inspiration.....	3
<b>Solution Design.....</b>	<b>4</b>
A. Requirements of program.....	4
B. Sign Up System.....	4
C. Database.....	5
D. Main Game Page.....	5
E. Leaderboard Connection.....	6
<b>Implementation Process.....</b>	<b>7</b>
A. Use Case Diagram.....	7
B. Class Diagram.....	7
C. Activity Diagram.....	8
<b>Program Functionality.....</b>	<b>9</b>
A. GitHub Repository.....	9
B. Input and Output Design.....	9
C. Extensibility of Program.....	12
D. Video Demo.....	12
<b>Evaluation.....</b>	<b>13</b>
A. Lessons Learned.....	13
B. Future Improvements.....	13
<b>References.....</b>	<b>14</b>

# Project Specifications

## A. Introduction

The creation of CyberSafe was heavily inspired by the iconic childhood game of Space Invaders. This project has implementations of a dynamic 2D game where users control their player navigating through different types of enemies bullet types, and collecting coins for further upgrades going through the waves. Built with pygame, this game blends better smooth graphics and responsive controls, providing players with a thrilling space-themed experience.

Furthermore, I've taken into continuing and upgrading a much more efficient code and better graphic design from my 1st try of making a game and improving it with more advanced techniques. The primary goal of this project was to explore game development principles, including physics-based movements, collision detection, and interactive UI components.

## B. Inspiration

An old iconic game from Space Invaders was heavily inspired by the idea of the game, CyberSafe. However, new features are being added to the game such as coin features to be implemented and used for updates in the attack and health, since Space Invaders is limited to its default health and attack. This new feature addition could increase the balance of difficulty when players are reaching for higher new scores going into higher waves of enemies.

Additionally, movements and types of attacks have been implemented differently to give players a bit more challenging views. This gives players a new sense of challenge and the adrenaline of fun with the increased amount of bullets that need to be dodged. With this, I've decided to turn this game idea into reality to create a more challenging Space Invaders.

# Solution Design

## A. Requirements of program

### Requirements to run the program:

- **Pygame:** Design the overall game and visualize it in a window
- **Tkinter:** Design for the GUI by giving a set of tools and widgets to create desktop applications
- **Pyodbc:** To connect within the Access file into the program to create a database
- **Random:** A pseudo-random number generator function that generates a random float number
- **Sys:** Provide functions and variables used to manipulate different parts of the Python runtime environment

## B. Sign Up System

The Sign Up System was created in a class consisting of Tkinter widgets such as the label button, and consisting of a function that gets the input from the *username\_entry*, *password\_entry*, *repassword\_entry*, and *security\_entry*. Additionally, it'll go to the next step further of going through the data validation and verification. It also creates any avoidance of having the same username so the system doesn't get confused with which password goes to what account. Upon submission, the system checks if the entered username already exists in the Users table to prevent duplicate accounts.

**Error handling** is ultimately used after going through the data validation which will be executed to the Access file by putting to the respective field in the database, otherwise, it will show an error when during the registration. Overall, there are security considerations of password storage where it stores passwords in plain text, and input validation to ensure that users' inputs are validated to prevent any forms of input manipulation.

## C. Database

CyberSafe leverages the pyodbc library to facilitate a connection with a Microsoft Access database, which is stored in the *.accdb* format and plays a central role in handling user authentication; within this database, there exists a users table designed to store essential credentials, including username and password fields, which are likely structured as text fields to allow for flexible input and retrieval.

The database connection is established using an ODBC driver, specified as *Microsoft Access Driver (\*.mdb, \*.accdb)* in the connection string. It incorporates the exact file path of the database to ensure accurate and consistent access. Key operations performed on the database include checking for the existence of a username in the Users table during the sign-up process to prevent duplicates, inserting new user records into the database via an **SQL INSERT query** when valid credentials are provided, and executing **SQL SELECT queries** during login attempts to verify that the provided username and password match an existing record in the database, ensuring that users can only access the system with valid credentials.

## D. Main Game Page

The *MainGame.py* file in the CyberSafe project acts as the central gameplay engine, initializing the Pygame display with specific screen dimensions, loading essential assets such as images and sound effects, and utilizing several classes to create a functional and interactive gaming experience; the *CyberSafe class* represents the player's spaceship, which is controlled using keyboard inputs to move horizontally and shoot bullets, while also managing the player's health, tracking its depletion upon collisions with enemies, and resetting the position when necessary; the *Enemy class* governs the behavior of enemy ships, which are spawned at random intervals and positions at the top of the screen, descending towards the player with unique movement

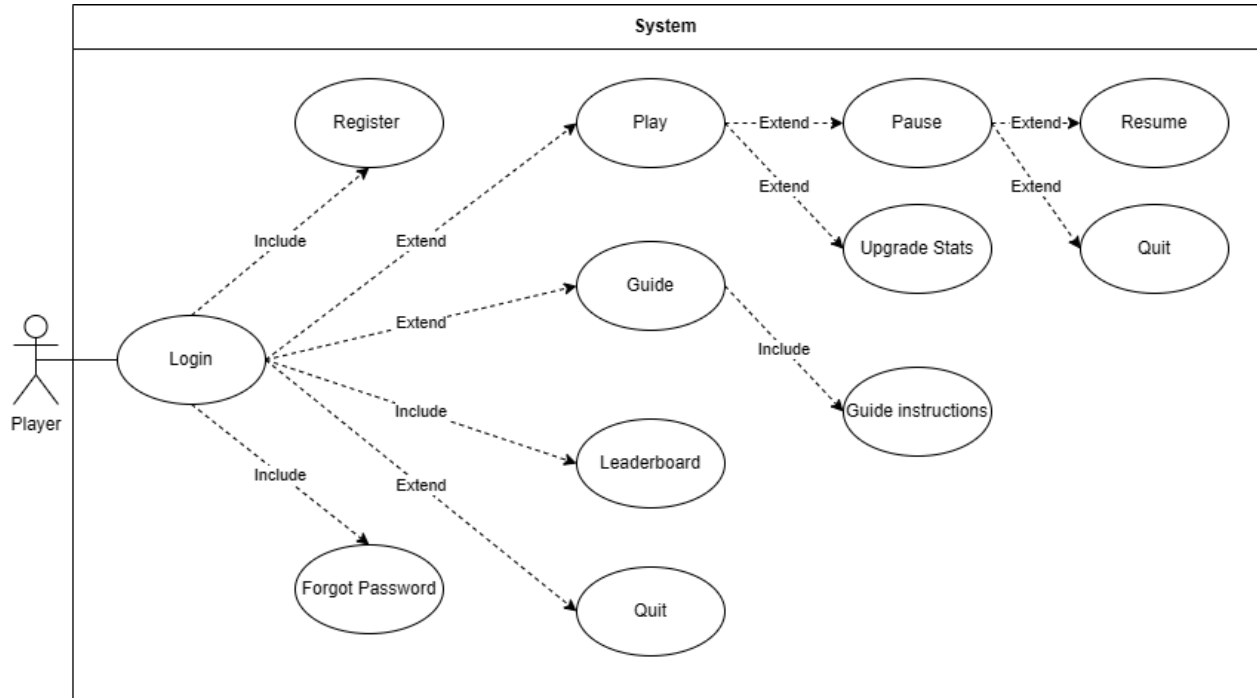
patterns and speeds, making gameplay increasingly challenging as more enemies are introduced; the *Bullet function* handles the player's projectiles, including their position updates as they travel upward, checking for collisions with enemy ships, and removing them from the screen when they leave the play area or hit a target; the game loop ties everything together by continuously processing user inputs, updating the states of all game objects, rendering updated visuals, and implementing collision detection to determine whether bullets have struck enemies or the player's spaceship has collided with an enemy, while dynamically updating the score when enemies are destroyed, displaying the player's remaining health and score on the screen, and saving the final score to the database using pyodbc when the game ends to facilitate leaderboard functionality, creating a cohesive and engaging gameplay experience with progressively increasing difficulty.

## E. Leaderboard Connection

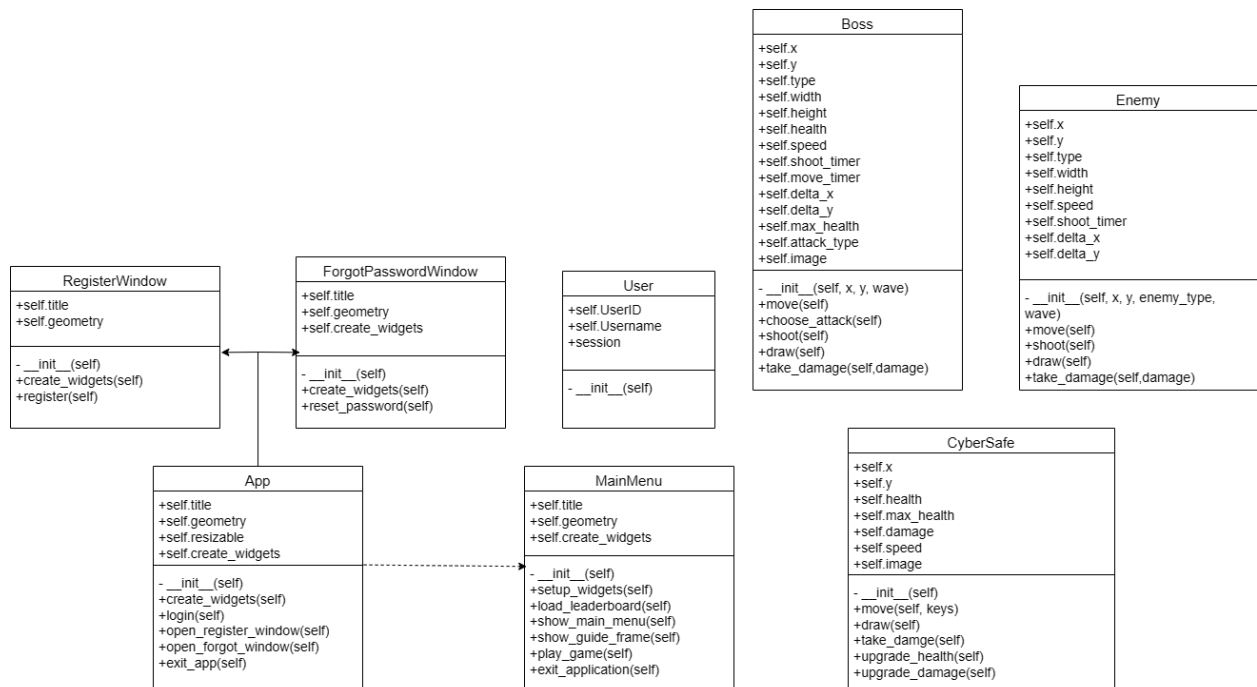
In CyberSafe, the leaderboard system is created by connecting to a Microsoft Access database using the pyodbc library, where the database connection is established via a configured ODBC driver and a specified file path for *CyberSafeDatabase.accdb*, allowing the game to execute **SQL queries** to insert new player scores and retrieve existing ones for ranking purposes; the leaderboard itself is constructed using a **Binary Search Tree (BST)** technique, where each node represents a player's score, with the tree organized in a way that maintains sorted order to enable efficient insertion of new scores and fast retrieval of the top scores. This structure ensures optimal performance for managing and displaying rankings, particularly in scenarios involving large numbers of players, as an in-order traversal of the **BST** can produce the scores in ascending or descending order for display, thereby seamlessly integrating the database-stored scores into a dynamic and efficient leaderboard system.

# Implementation Process

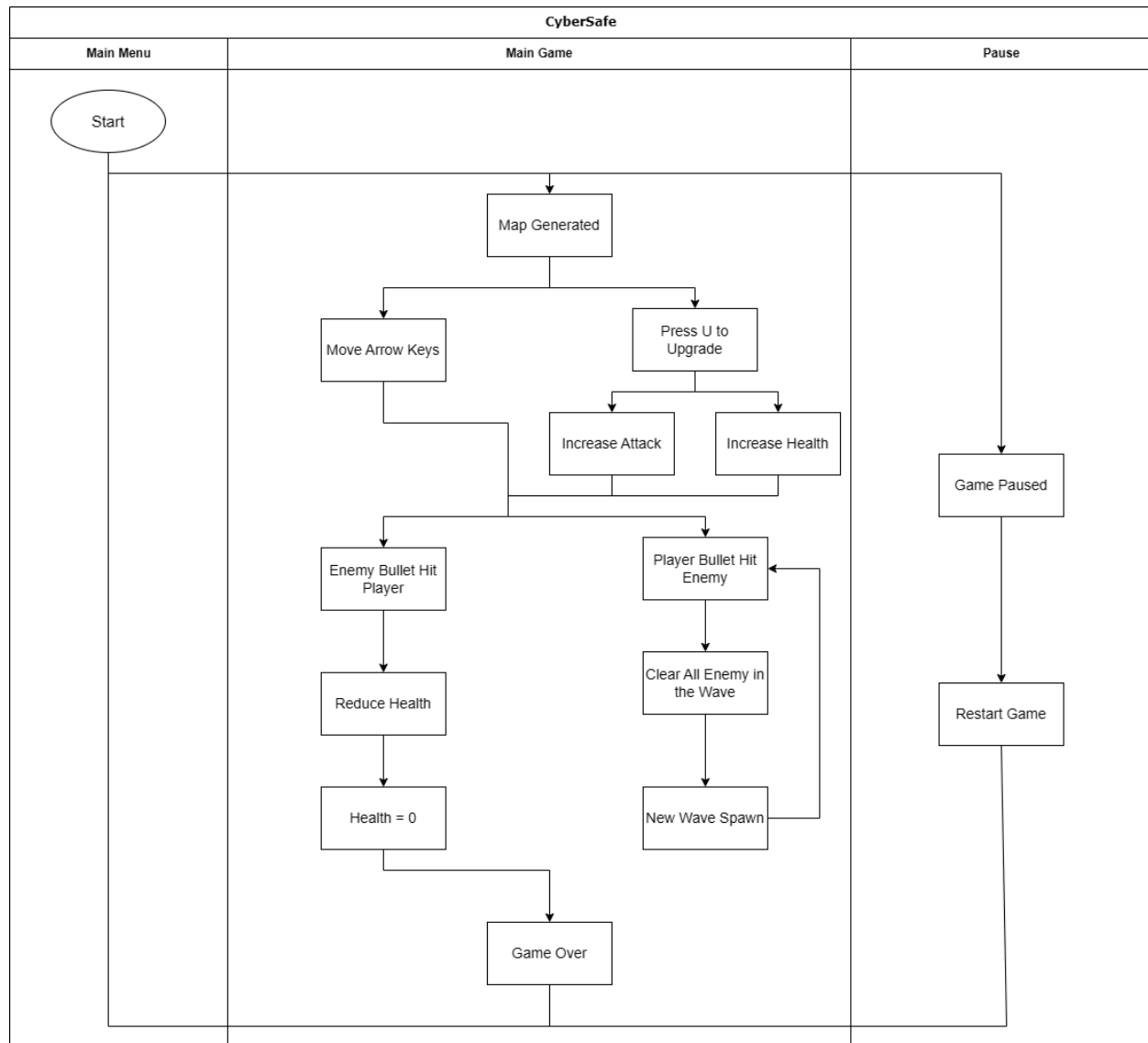
## A. Use Case Diagram



## B. Class Diagram



## C. Activity Diagram





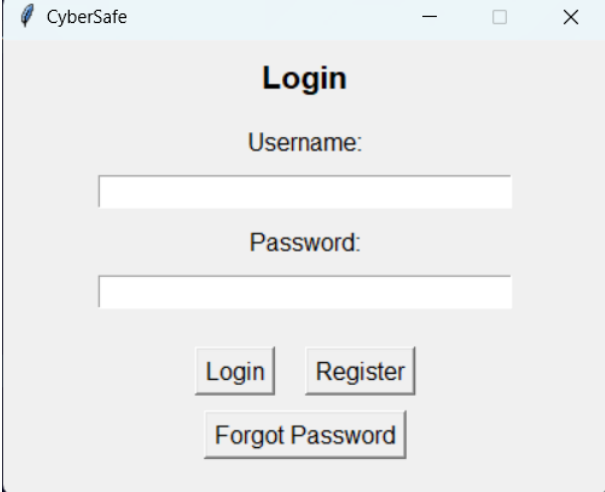
# Program Functionality

## A. GitHub Repository

(<https://github.com/kevMkr/SpaceShipGame>)

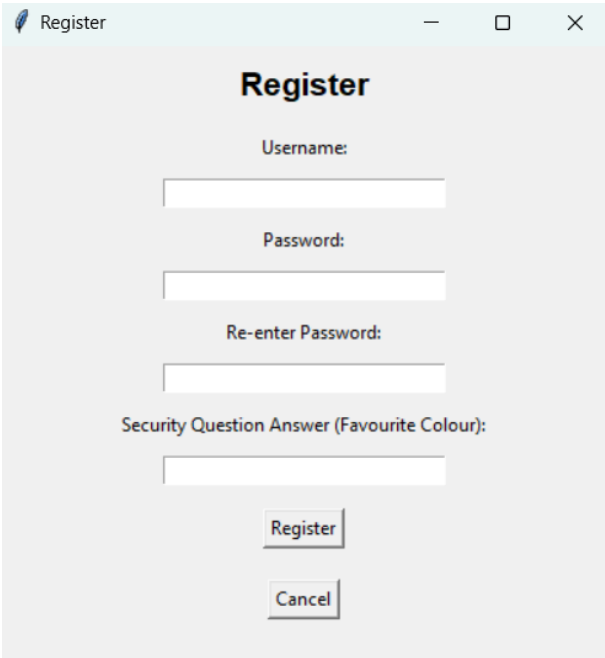
## B. Input and Output Design

Figure 1. Login Page



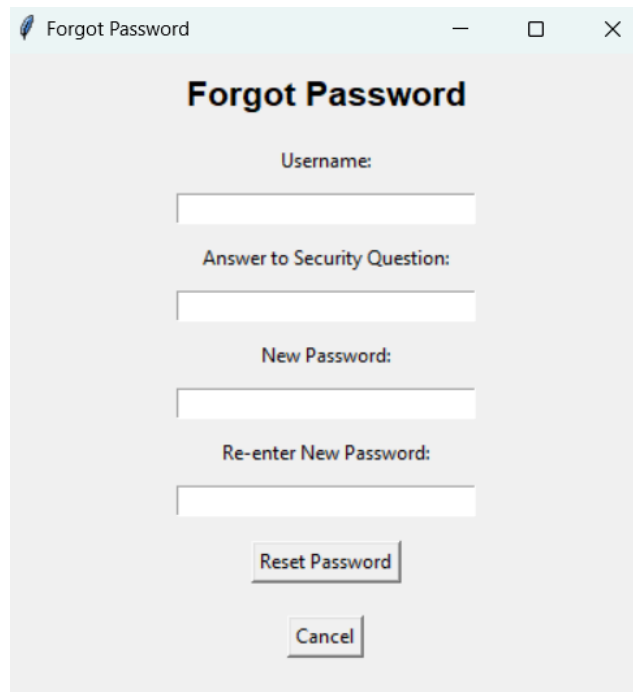
The screenshot shows a window titled "CyberSafe" with a light blue header bar. The main content area has a light gray background. At the top, the word "Login" is centered in bold black text. Below it, the label "Username:" is followed by a white text input field. Underneath, the label "Password:" is followed by another white text input field. At the bottom, there are three buttons: "Login", "Register", and "Forgot Password", all with black text and thin black borders.

Figure 2. Sign Up Page



The screenshot shows a window titled "Register" with a light blue header bar. The main content area has a light gray background. At the top, the word "Register" is centered in bold black text. Below it, the label "Username:" is followed by a white text input field. Underneath, the label "Password:" is followed by a white text input field. Below that, the label "Re-enter Password:" is followed by another white text input field. Further down, the label "Security Question Answer (Favourite Colour):" is followed by a white text input field. At the bottom, there are two buttons: "Register" and "Cancel", both with black text and thin black borders.

**Figure 3. Forgot Password Page**



A screenshot of a web browser window titled "Forgot Password". The page has a light gray background and contains the following elements:

- Username:** A text input field.
- Answer to Security Question:** A text input field.
- New Password:** A text input field.
- Re-enter New Password:** A text input field.
- Reset Password:** A button.
- Cancel:** A button.

**Figure 4. Main Menu Page**

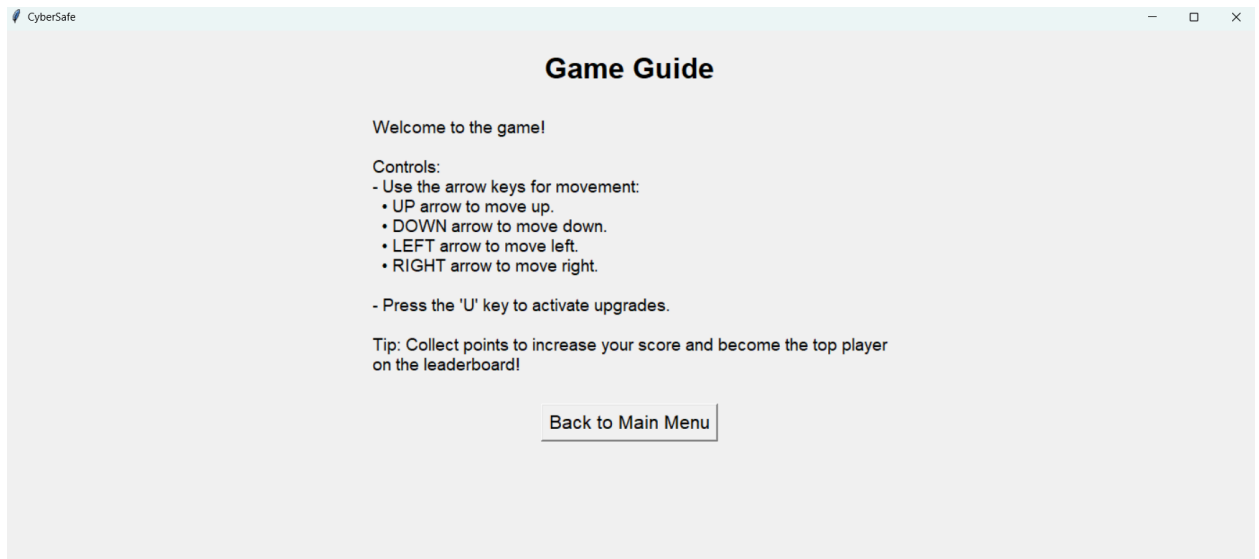


A screenshot of a web browser window titled "CyberSafe". The page has a light gray background and contains the following elements:

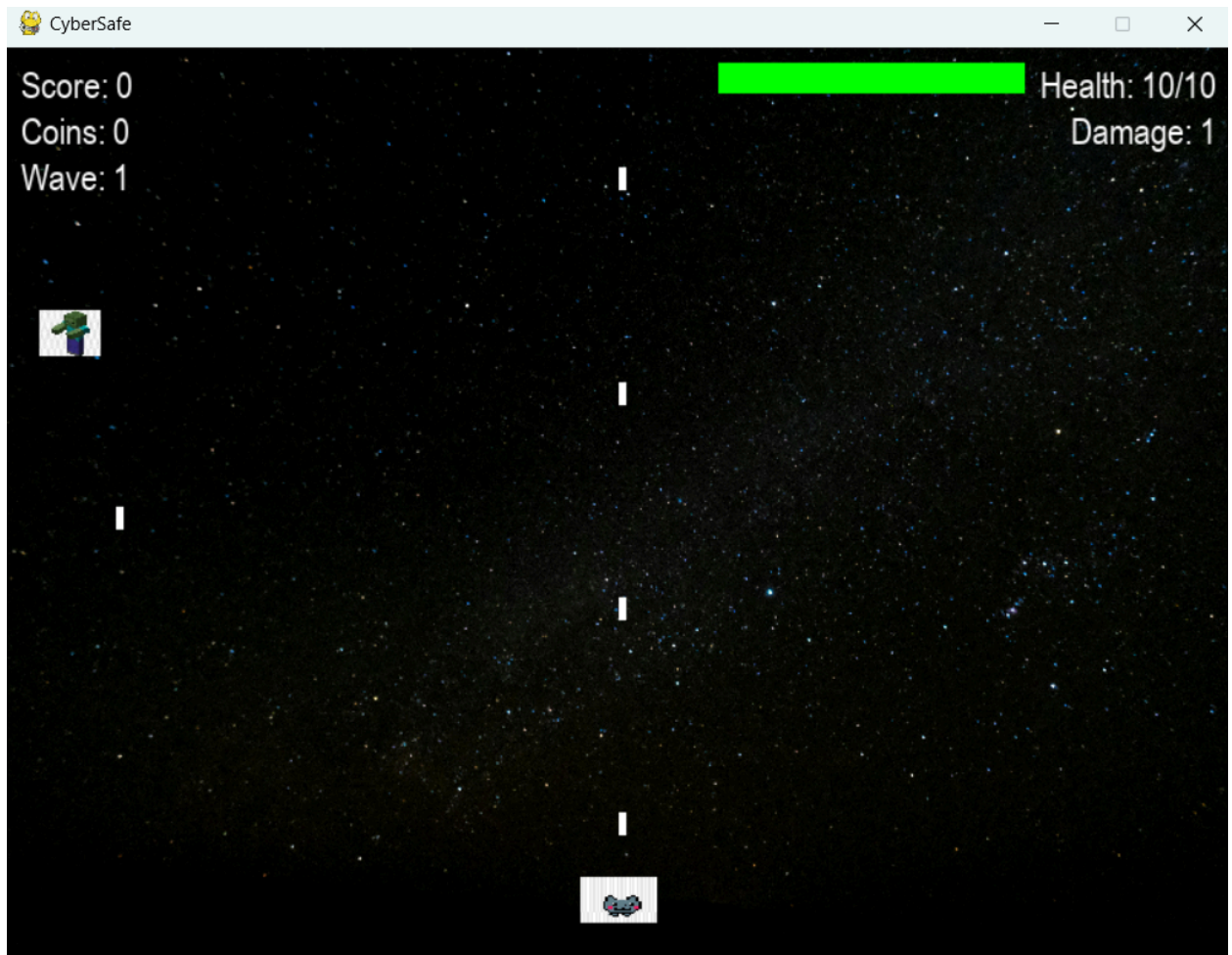
- CyberSafe:** A large heading on the left side.
- Play Game:** A button.
- Guide:** A button.
- Exit Application:** A button.
- Leaderboard:** A table on the right side.

Rank	Player	Score
1	clara	2950
2	clara	2750
3	test7	2700
4	clara	1150
5	test7	850
6	1234	500
7	test31	300
8	test6	150

**Figure 5. Game Guide Page**



**Figure 6. Main Game Page**



### **C. Extensibility of Program**

1. Comment lines are implemented on some parts of the code to further explain of what each function or code of line serve its purpose for other programmers to create their own.
2. Easy identifiers for variables to allow much easier what the variable does
3. Database use meaningful names to make it easier to understand the data stores
4. Data structure for a database is given for future programmers to understand the content of each entity that is recorded in the database.
5. Program structure is divided into files, functioning as a library for the main file, making it easier for organizing modules.

### **D. Video Demo**

- **Video Link:**  
([https://drive.google.com/file/d/1wXiqKJDA0a2Wfb0vXlmFuDE6SSZJDZix/view?usp=drive\\_link](https://drive.google.com/file/d/1wXiqKJDA0a2Wfb0vXlmFuDE6SSZJDZix/view?usp=drive_link))

# Evaluation

## A. Lessons Learned

From this project, I've learned how to fully utilize and make game application using pygame by using advanced techniques such as binary tree, database connection, class inheritance. It teaches me on how to implement these techniques on certain algorithms for it to work efficiently. Additionally, its also help me understand the importance of implementing features of collision detection where responsiveness is key for majority of real-time games, while creating a balanced difficulty that can create a hard time to pass by for players to kill the enemies. Overall, developing a game application has taught me the importance of balancing creativity with technical constraints, emphasizing the need for efficient coding, and player-centered design to create an engaging experience.

## B. Future Improvements

For future references, a lot of improvement can be still made within giving more time and effort by implementing a better AI-movement enemies, customization options for players to customize their player such as bullet colours, or aura effects. Additionally, abilities could be added to make the game much more fun where it can be replayed without losing boredom easily.

Multiplayer implementation could be one of the hard techniques that can be learn and incorporate to increase the engagement and expand the audience. With the usage of local co-op mode or online multiplayer can increase the excitement level of the game and avoid boredom of playing singleplayer for most of the time.

## References

CyCoderX (2024). *Python sys Module essential Intro Guide | Python's Gurus*. [online] Medium. Available at:

<https://medium.com/pythons-gurus/python-sys-module-beginner-guide-e7585684c26c>.

GeeksforGeeks. (2020). *Introduction to Tkinter*. [online] Available at:

<https://www.geeksforgeeks.org/introduction-to-tkinter/>.

Toppr-guides. (2021). *Python import random module | Why do we use Python import random?* |. [online] Available at:

<https://www.toppr.com/guides/python-guide/tutorials/modules/modules/random/use-random-module-to-generate-random-numbers-in-python/>.