# Introduction

My idea is to make a 2d, survival, rogue-lite game. The current idea for its title is something short and relevant to the project, such as: ‘Wave survivor’, ‘Sir. Vivor of Whaves’ or ‘Self savour’. I like these names because of the simplicity and self-explanatory nature of them. The aim of this game is to survive an endless number of waves with increasing difficulty for as long as you can, unlocking upgrades along the way and earning new abilities after X number of total kills.

The game will open to a start menu with options to start a new run, continue a run, or restart the save. This will then put the user into a run (New if they chose to start a new save or new run, otherwise, it would continue the run).

# Similar products

Some similar applications to mine are: Brotato, Vampire Survivors, and Hades

|  |  |
| --- | --- |
| Brotato – Blobfish & Seaven Studio | |
| * Screenshots taken by me | |
| What I like   * I like the game over screen showing what you got in your run and how far you got * I’m a fan of the colouring scheme to show what rank each item/weapon is * I like the acclimation of weapon types giving you stat buffs | What I dislike   * I don’t think the art style used would work for me, and would likely mean a slight revamp on the GUI to fit whichever style I go for,   I think the pause and game over screen show too much information at once, and I would |

|  |  |
| --- | --- |
| Vampire survivors – Luca Galante (Poncle) | |
| * Screenshots taken by me | |
| What I like | What I dislike |
| * I’m a fan of the limitations on the player that no re-rolling causes * I like the upgrades and characters you can unlock outside of waves * I’m a fan of the GUIs simplicity | * I don’t think I’ll be using the art style as although I think it works in this game, I don’t think I’d be able to pull it off well * I find some of the weapons later in the game are too flashy and would be difficult for me to implement whilst also making them feel balanced * I think the pause menu is too bare bones and could do with some information on your stats and weapons |

|  |  |
| --- | --- |
| Hades – Supergiant Games | |
| * Screenshots taken by me | |
| What I like | What I dislike |
| * 8 directional movement system * I like the boons system that grants the players various skills and weapons more abilities/functions making each run feel unique * (From hades 2) Final boss ‘Chronos’ Has an ability that prevents the player from pausing | * I dislike the limit on casts (A ranged attack option) within the game and feel it interrupts the flow of combat within the game * I will not be using the art style of this game as it is too complex for me to replicate on top of everything else, I will have to do to make this * I’m not a fan of the armour system for enemies since I find it hides the amount of damage you deal to them in each hit compared to how much health they have |

# Requirements:

|  |  |  |
| --- | --- | --- |
| necessary | Preferably | optionally |

|  |  |  |  |
| --- | --- | --- | --- |
| What | What it needs | description | Justification |
| Movement | Vertical and horizontal movement keys | The system that’ll be primarily used to move the player. This will be done by changing the players velocity in a C# script attached to the player object. | If the player can’t move, I won’t be able to implement any dodging mechanics, and the player will be forced into making a defensive character build |
| Main menu | A start run button, a continue run button and settings | A hub for information that helps direct the player into a run. This will be made in a separate scene as the game and will function primarily off of a canvas made to the size of the camera’s view. With the games title at the top and a button by the middle of the screen with some C# code attached to direct the player into the game when pressed. | Provides a hub for the user to navigate the game and prevents them from booting up straight into the game when they may be unprepared and feeling that they lost unjustly |
| Pause menu | Likely bound to the escape key, needs to prevent the wave from progression and stop enemies and players from making any moves | A basic menu that will open and pause the game to let the player take their time with whatever they choose.  This will be done by having the game code reliant on the ‘pause’ variable being false, if true, then the pause menu will show and pause the current game. This could be bound to a key with some C# or it could also be made with an object to create an onscreen button which some C# code would be linked to pausing the game. | I think this will be easy to implement and give the users a place to relax and think |
| Weapon types | An attribute to each weapon to scale its damage along with its related attribute | An attribute linked to a weapon based off its type that determines what stats affects its damage output.  Made with a variable attached to the weapon’s class upon its generation. | I think this will give the game some more complexity and depth whilst keeping it easy to pick up. |
| Player stats | * Experience / currency system * Player variables used to alter damage, speed, etc. | Stats the player can level up outside of waves with items or with materials collected inside of combat.  Each enemy defeated would drop some money/experience that would count towards your stat increase. This would be executed before any enemy instance is destroyed in the code, by calling a function in the players class to increase their money/experience by an amount corresponding to the enemies health. Then after a round change the scene to a level up or shop menu to let the player change their stats. | The aim of this is to give the player something to entice them to do more waves and combat to increase their stats and abilities strength |
| Currency | A number that will go up after each enemy kill and will be used to upgrade weapons/player stats | A number that goes up depending on how many enemies you killed (and if possible, how difficult they were), to be used on upgrading weapon stats or player stats (Undecided). Base system behind upgrading player stats, each enemy giving the player a percentage of their total health in currency/experience before their instance is destroyed. | This will help encourage the player to take on harder waves as it will increase the amount they earn and let them get stronger quicker |
| Weapons/ upgrades store | A menu that pops up after each wave with purchasable items in it | This will be a menu that will display 4 random items (Weapons or upgrades) to help improve the players survivability, each item having a predetermined cost based off the stats of the weapon/upgrade. The store menu will be on a separate scene than the game and will pas items and their attributes into the players inventory (An array of items stored within the player class). | This will be what gives the player the ability to upgrade during a run and make custom builds, to help put some enticing strategy into the game |
| Boss battles | A strong enemy at the end of every 5 or 10 waves, with stronger attack & defence | A stronger enemy meant to act as a challenge to players and have them focus on single target damage instead of crowed damage for a brief time. This would be done by scaling up an enemies stats and using a unique sprite for the boss. | This should entice the player to spend more time on the game trying to beat the next boss and overcome its challenges |

# Limitations:

|  |  |  |  |
| --- | --- | --- | --- |
| What | Why you can’t | What it would do | Justify |
| Jumping | It would have the same practical effects as dashing up | Raise the players height momentarily before they fall back. | It would be too difficult for the player to know they are jumping considering the game will have a top-down viewpoint |
| Dash/dodge mechanics | Extra key (Likely left shift) to dash in the direction you’re facing with an upgrade to give I-frames (invincibility / immunity frames) during the dash | Some code to multiply your velocity for a brief period after the key is pressed, made to get away from attacks, optionally I might add a brief period where you’re slowed afterwards | I think this will make the combat more immersive and cause for a better user experience (If it’s possible, to implementt). |
| Inventory | Likely bound to ‘e’ it will open a menu that lists the players inventory and upgrades | A menu that will open after the designated key is pressed and display an icon for the items a player has and the name next to it | This will help the player keep track of their items and weapons to make planning your next pick-up easier (If possible). |
| Weapon stats | A different damage modifier to each weapon | This would be a set of stats (E.g. attack speed, damage, etc.) bound to each weapon (Possibly with some variation) to give each weapon a unique feel and would link into the weapon types | This would make the game more engaging and give it more depth, however as it is not required and I do not currently know of how difficult it would be, I will only do it the time restraints/limitations allow |
| Level ups | A counting system of the enemies defeated, and amount required for a stat bonus | An increasingly higher amount of enemies to be slain in order to gain a slight buff to all stats | I felt this wouldn’t work well with the other upgrade methods and decided to only implement it if I get time at the end to merge the two systems well |

# Software and Hardware:

## Hardware:

The minimum required hardware for unity (on windows) is:

* X64 architecture with SSE2 instruction set support (CPU)
* DX10-12 or Vulkan capable GPUs (Graphics API)
* Additionally requiring ‘Hardware vendor officially supported drivers’

^All data from (Unity, 2025)

I will also code it to work with a keyboard and mouse making them both necessary.

These are necessary for my game to run as it will be coded in and run on unity meaning the hardware requirements above would be needed by default. And the keyboard and mouse necessary are because it’s what I have so it is what I will be testing the game off of and will be unable to properly test if the code functioned with a controller/touchscreen.

## Software:

Software requirements

* Windows 10 or 11 (OS)

^All data from (Unity, 2025)

Given that windows 10 has been out for quite some time now and there is a newer version of windows out now, most devices should be able to run my game. Making it more accessible for new players trying to get into playing games. I will also be developing this game on windows, and I am only able to test that it works on a windows device, limiting the options for operating systems to use.

# Survey results:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

### How old are you?

For this question all the answers were about 17 or 16 years old.

17 with 6 results

16 with 3 results

I asked this to understand the general age of my audience and determine if and how to add any nostalgic features to the game that may favour specific age ranges. This is something I will keep in mind during the development process, but because of how young the age range is I would struggle to find something old enough to fit into the game whilst giving that nostalgic feeling.

### Out of; Gameplay, graphics and the soundtrack of a game, which is most important to you?

For this question the majority of respondents had answered with gameplay as most important, then graphics and finally the soundtrack.

I had asked this to help prioritise the order of my development and omit any ideas that were not favoured by the stakeholders, with these results I can tell features based off the sound design of the game can be postponed and worked on later on in the development process if I have spare time. I can also tell that I should focus on making the gameplay more enticing and feel more fluid before worrying too much about how it looks or sounds.

### Please say a feature that you would like to see:

For this request there were quite a few people that had responded with something along the lines of: ‘Cool weapons which can be combined into stronger weapons’. Given this is a rather specific request it should be a primary point of the weapons system and should be implemented after the enemies and player have a basic working model.

### Out of Enemies abilities or player abilities, which would you prefer?

For this question I got a surprisingly even number of responses for both sides, with 56% saying player abilities and 44% saying Enemies abilities. This tells me that I should focus on making both feel balanced and well-designed whilst prioritising other areas that the stakeholders feel more strongly/unanimous with.

### Overall:

Overall with this survey I have deduced that my stake holders are 16-17 years old and prefer smooth gameplay mechanics than the looks and sounds of a game. I can also tell, in order to favour them, I should start with developing a working game before moving on to trying to design a fitting character and background.

With this, I can design a better game that will be more enriching for the stakeholders to play and give them a more fun way to spend their time playing.

# Success criteria

## General criteria:

* Start menu to continue a run or start a new run
* Pause menu to save the game
* Inventory to see what you have
* Creating a new game
  + Selecting starter weapon
* Loading a game

## Justification:

This (the start and pause menu) would provide the user with smooth navigation options to start pause and load their game, improving the flow of it and keeping the user immersed. The selection of starter weapons and management of their inventory will empower the player to make strategic choices and make each un feel different. Along with the above, a save/load mechanic would make this game much easier to pick up and play, due to not having the worry of losing all your progress as soon as the program closes/finishes.

## Gameplay criteria:

* Enemy spawns
* Scaling difficulty with time/wave number
* Shop menu at the end of a wave
  + Working currency system
* Auto attacking (player & enemies)
* Movement

## Justification:

The use of enemy spawning and scaling should create a sense of growing tension and excitement in the player and keep them enticed. The use of currency and experience in the game provides a rewards system to keep the user enticed, the stronger the enemies they defeat, the stronger they can become. The auto-attacking feature should help keep the game less aim focused and more strategic based off what you’ve collected along the way and how you use them. Finally, movement is a core mechanic that will be used to make the player feel they are more responsible for what happens in game and better help them strategize their next attempt to overcome any errors in their previous run(s).

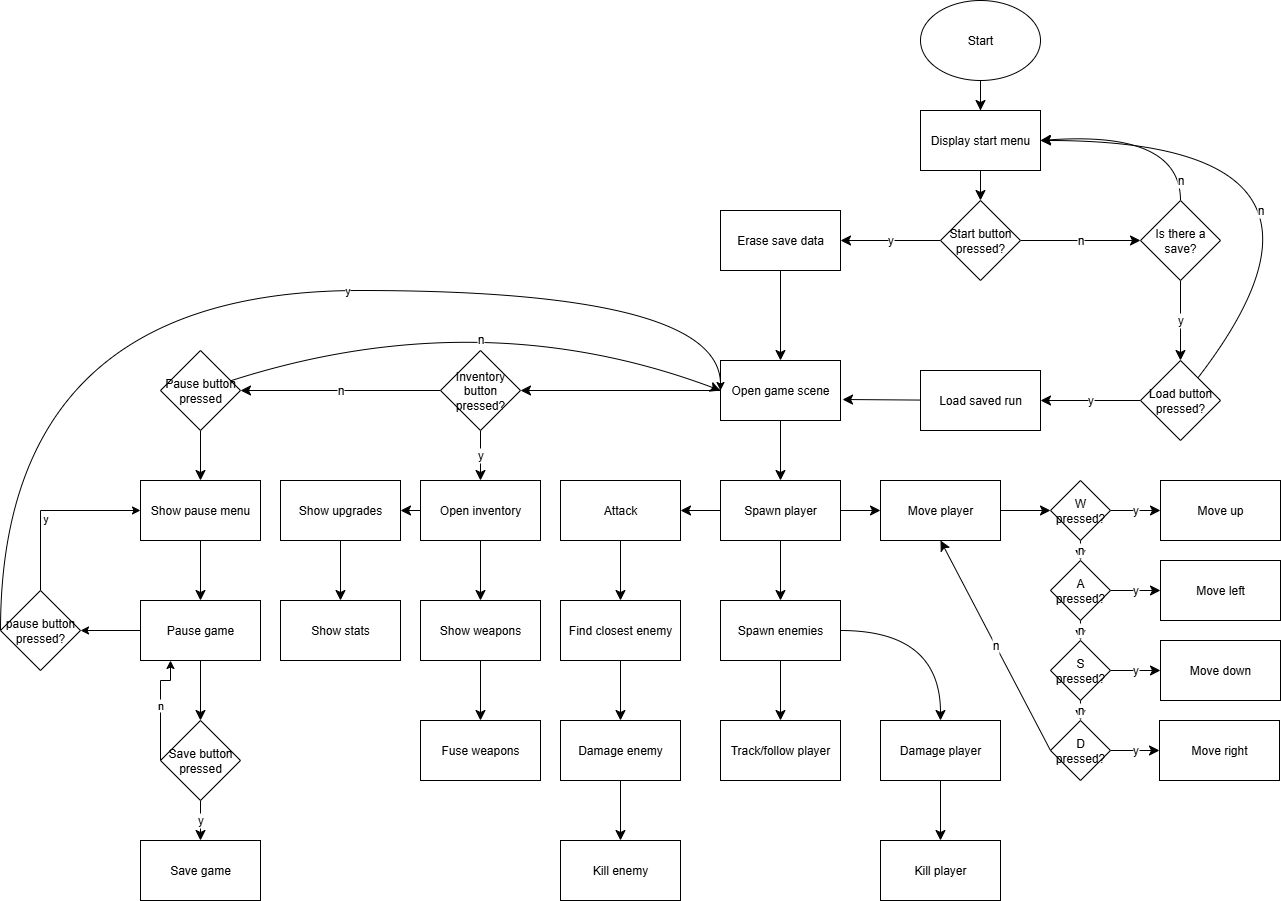
## Limitations:

Due to the potential scale of this project, I understand that I may not be able to finish everything and as such will have to impose some limitations on how I approach each task. Making sure the final product feels well done, whether it means leaving some features out or implementing some to replace others.

My largest restraint on this will likely be implementing boss fights. I say this since it would require making a unique enemy attack algorithm without adding all too much to the quality of the game.

Another limitation will be my artistic capabilities, or lack thereof, making the process of designing a UI much longer and requiring more effort and care than most other areas. Similarly, I may have to scrap making a soundtrack due to the time and commitment required for it to sound good and fit the theme / occasion.

# Flow chart:



# GUIs:

A diagram of a screen

AI-generated content may be incorrect.

## Justification/Description (Screen 1):

* I put the title there, because it’s the title screen.
* I put the start button there to start a new game, clearing the save data and loading the game scene (Scene 2)
* The load button there is there to continue a previous run and load the save data, also moves the user to game scene (Scene 2)

A screen shot of a screen

AI-generated content may be incorrect.

## Justification/description (Screen 2):

* The player, is the player character, which the user controls
* The enemy is an object that tracks and attacks the player
* The pause button, pauses the game, loading screen 3 (The pause menu)
* The inventory button also pauses the game and opens screen 4 (The inventory menu)
* The save data button, saves the runs data to be loaded at any time

A screen shot of a screen

AI-generated content may be incorrect.

## Justification/Description (Screen 3):

* I put the paused there to show the game is paused and reduce confusion for what screen the player is on
* The play button is there to be used to play the game and un-pause it
* Home button, quits the game and puts you back to the main menu

A screen shot of a computer screen

AI-generated content may be incorrect.

## Justification/Description (Scene 4):

* The back button is there to go back to the game and restore progress,
* The item slots are there to display the items the player has with an icon for each and to be colour coded for the item’s rarity

# Code:

## 1.

A computer screen shot of a program code

AI-generated content may be incorrect.

This is the beginning of some movement code for the player, this acts more as a first draft than anything to make the player move to help remind me of how C# works with Unity given it’s been a bit since I last used it. The next thing I’ll change will involve me either looking up my previous work in Unity to find that movement code or looking it up to remind myself.

## 2.

## A screenshot of a computer program AI-generated content may be incorrect.

This is the next template to test how the new unity input system works since Unity had an update since the last time I used it, which altered the input system, (Unity, 2025).

The variable horizontal is used to tell weather or not the player would move at all whilst the arrow keys are pressed, this is output to the console with “Debug.Log(horizontal);”

## 3.

A screen shot of a computer program

AI-generated content may be incorrect.

This is a working version of some movement code, currently it only works for moving horizontally, but, it has helped me to understand where the issues in my previous code were and how to fix it. The previous errors were from there being two vector functions that it was trying to use and the use of the old Unity input system.

The new code works by getting if the left or right key are down and alternating the value of a variable called ‘horizontal’, this is then checked when changing the position of the player on line 31 to determine which direction the player is moving. It is then multiplied by Time.deltaTime so that the code is not framerate dependant.

Next, I will add a variable to multiply the value on line 31 by so that the player moves faster and can have their speed changed by upgrades later on. I will also implement vertical movement.

## 4.

A screenshot of a computer program

AI-generated content may be incorrect.

In this I added the vertical movement and a variable called “VelocityMultiplier”. The vertical movement code is the same as the horizontal movement, but when the variable for it is used it alters the y co-ordinate instead of the x co-ordinate.

Next I will add the Enemy movement, which should track and move towards the player.

## 5.

A screenshot of a computer program

AI-generated content may be incorrect.

This is the code for the enemy tracking and moving. It works by getting the players x & y coordinates and comparing them to its own, the player game object has the tag “Player” allowing it to be referred to as it is on line 22.

Currently there is an issue where if the player moves to the right the enemy will move up, and if the player moves to the left it will move down. Next, I plan on finding where this issue is, and fixing it.

## 6.

A screen shot of a computer program

AI-generated content may be incorrect.

This is the updated code for the enemy movement system, the change made was altering line 28 so that it wouldn’t save the players X-coordinate to the variable supposed to store their Y-coordinate. Next I will try and change the enemies code so they move in a direct line to the player instead of only moving in one of 8 directions.

## 7.

Pseudo code for smoother enemy movement:

float function FindMovementRatio(char axis){

float y = Player.getY()

float x = Player.getX()

float totalLength = sqrt(square(x) + square(y))

y = y / totalLength

x = x / totalLength

if (axis == x)

return x

endif

Return y

End function

This is the pseudo code for a function that should be able to calculate and return a float value for the x and y velocities needed to head in a straight line towards the player character.

## 8.

A screenshot of a computer program

AI-generated content may be incorrect.

Used (Unity, 2025) for finding out how to square root in unity.

This is the first attempt of coding a more detailed movement system for the enemy, this code is supposed to track the player and travel in the most direct line towards them.

Next, fix the issue of the enemy starting to track the player, before going off in a seemingly random direction.

# Data dictionary:

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Name | Usage | Where, please? |
| Bool | Example variable | Not in code just showing how to use table | This line |
| float | movementX | Used to give the players movement a direction in the x axis | Line 7 in the “player script”, see in Code – 1 |
| int | movementForce | Used to set a magnitude for the players movement | Initialised on line 8 and set on line 13 of “player script”, see in Code – 1 |
| float | horizontal | To determine the horizontal direction that the player will move in | Initialised on line 21 and altered on lines 24 & 28 of the “PlayerScript”, see in Code - 3 |
| float | vertical | To determine the vertical direction the player will move in | Initialised on line 35 and edited on lines 38 & 42 of the “PlayerScript”  See code in Code – 4. |
| GameObject | playerObj | A variable used in the enemy code to refer to the player and track its location | Initialised on line 11 and used on lines 21 and 22. See in Code – 5. |
| Float | velocityMultiplier | A float used to multiply the speed of a game object | Initialised on line 13 and used on line 46. See code in Code – 4.  And:  Initialised on line 8 & used on line 38. See code in Code – 5. |
| Float | velocityY | A float used to determine the vertical direction that an enemy should move to get to the player | Initialised on line 5, changed on lines 30 and 27, used on line 38. See code in Code – 5. |
| Float | velocityX | A float used to determine the horizontal direction an enemy should move in order to reach the player | Initialised on line 6, changed on lines 33 & 36, used on line 38. See code in Code – 5. |
| Float | FindMovementRatio | A function to find the value that the x or y velocity should be to move in a direct line to the player. | Initialised on line 33 and used on lines 26 & 27, see code in code - 8. |
| Float | xCoord | Holds the value for the horizontal velocity of the player within the FindMovementRatio function | Initialised on line 11 and used on lines 37, 40, 42 and 44, see code in code - 8. |
| Float | yCoord | Holds the value for the vertical velocity of the player within the FindMovementRatio function | Initialised on line 12, used on lines 38, 47 and 49, see code in code - 8. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

I have not put the Start, update or other functions of the sort into the data dictionary as they are all Specific to unity and are in each script.

# Test Logs:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | Why & what results mean | Result | What to do | Screenshot(s) / Video(s) |
| E.g.  Movement test | Pass, player can move fine, fail, player can’t move | fail | Do any amount of coding | :( |
| Movement  (1) | I had finished a base draft of the player movement code, not meant to work but I still tested it to be sure. | fail | Further work on the PlayerScript | See the code in the code section under “1.” |
| Movement  (2) | I updated the code from last time to fit in with the new input code for Unity | pass | Update the code again to actually move the player instead of just output a value to the console | See the code in the code section under “2.” |
| Movement  (3) | I had once again updated the code so that the player would actually move, if it passes that means the player can successfully move horizontally | pass | Add vertical movement next and a variable to change the velocity of the player | See code in the code section under “3.” |
| Movement  (4) | I updated the code to include vertical movement and a variable “VelocityMultiplier” to alternate the speed that the player moves | pass | Finish up on the player movement for now, and start to develop the Enemy movement | See code in the code section under “4.” |
| attacking |  |  |  |  |
| Auto-aim |  |  |  |  |
| Enemy spawns |  |  |  |  |
| Player loading |  |  |  |  |
| Weapon selecting |  |  |  |  |
| Weapon fusing |  |  |  |  |
| Shop menu loading |  |  |  |  |
| Main menu loading |  |  |  |  |
| Game scene loading |  |  |  |  |
| Pause menu opening |  |  |  |  |
| Pause menu pausing |  |  |  |  |
| Enemy movement & player tracking  (1) | Test if the Enemy can follow the tracked player | Fail | Although the enemy would follow the player in the x-axis, it would go up when the player moved right, and down if the player went left.   * Need to find the cause of this error and make a solution. | See code in the code section under “5.” |
| Enemy movement & player tracking  (2) | Test if the enemy moves towards the player | Pass | Make it a more direct line | See code in the code section under “6”. |
| Enemy movement & player tracking  (3) | Test if the enemy takes the quickest route to the player | fail | Find out why the enemy goes off into a random direction and fix it | See code in the code section under “8”. |
| Enemy movement |  |  |  |  |
| Enemy attacking |  |  |  |  |
| Player gaining currency |  |  |  |  |
| Waves increasing |  |  |  |  |
| Kill count |  |  |  |  |
| Wave difficulty increasing |  |  |  |  |
| Worl map / background |  |  |  |  |
| Tracking where an enemy is and its proximity |  |  |  |  |
| Enemies tracking player |  |  |  |  |
| Back ground music |  |  |  |  |
| Sprites loading |  |  |  |  |
| Collisions |  |  |  |  |
| Death |  |  |  |  |
| Saving |  |  |  |  |
| Loading |  |  |  |  |
| Inventory system |  |  |  |  |
| Showing inventory |  |  |  |  |
| Restarting |  |  |  |  |
| Erasing save data |  |  |  |  |
| Weapon stats |  |  |  |  |
| Dash |  |  |  |  |
| Swapping weapons |  |  |  |  |
| Player upgrades |  |  |  |  |
| Starter weapon choice |  |  |  |  |
| Bosses spawning |  |  |  |  |
| Boss stats |  |  |  |  |
| Boss attacks |  |  |  |  |

# Post Development Testing:

Questions:

1. Did you understand the controls?
2. How was the difficulty?
3. What parts did you enjoy?
4. How far could you get?
5. Would you play it again?
6. How intuitive were the menus?
7. How would you rate the games performance?
8. How would you Rate the gameplay?
9. Where there any errors you came across?

# References

Unity. (2025, October 06). *Read keyboard input to control 2D character movement*. Retrieved from Unity Learn: https://learn.unity.com/course/2d-beginner-adventure-game/unit/player-character-and-movement/tutorial/read-keyboard-input-to-control-2d-character-movement?version=2022.3#64d2465dedbc2a00225234aa

Unity. (2025, 10 13). *Unity - Scripting API: Mathf.Sqrt*. Retrieved from Unity Documentation: https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Mathf.Sqrt.html

Unity. (2025, July 3). *Unity Documentation*. Retrieved from Unity: https://docs.unity3d.com/6000.1/Documentation/Manual/system-requirements.html