OCR A Level

COMPUTER SCIENCE PROJECT

H446-03

Name : Sufyaan Hafiji

Candidate Number : <[redacted]>

Huddersfield New College : 37514

Title of Project : Pyro Gunner

H446-03 – PROJECT CONTENTS

TABLE OF CONTENTS

A. ANALYSIS

OUTLINE OF THE PROBLEM

For my NEA project, I am making a 2d platform game that needs the player to beat enemies and do successful parkour to reach the end of the level. This game will be a offline game and I have taken inspiration from Mario which also has enemies to beat and parkour needed to complete. The character will be controlled by the player and the enemies will be ran by a AI doing loops of code until they are killed. The player will have to shoot and collect gems in a time frame to complete the level and there will be a score displayed at the end to show how well the player did and if they are improving or not. After completing the levels the game will also unlock the next level for the player to continue allowing for a less repetitive game that can be replayed by having variety of levels and difficulty. There will also be powerups that boosts the players stats like the shooting speed or the running speed and can also give health back to the player allowing for a more interesting game.

For this game to work it requires:

- A main menu
- Settings menu
- Game screen
- Level selector
- Death screen
- Level finished screen
- Player that the user will control
- A timer
- Ammo for the player
- Health for the player
- Health bar
- Scorebox
- Gems
- Platforms
- Enemies
- Ending picturebox (Finishing flag)

STAKEHOLDERS

The game I am creating is suitable for teenagers and above but I am primarily targeting teenagers who don't need to invest a lot of time into a game and to just be able to play a simple arcade game to kill some time. The platform for this game is PC, which is suitable for the target audience, as most of them will have one for doing things like work or if they simply cannot afford or want to buy a console. The game will also have a guidance process of showing the controls but also a tutorial level, which shows the basics of how to play the game allowing me to make the game more complex as the target audience will have an understanding of how to play. For me to create a good successful game, I need to have stakeholders to allow me to get regular feedback on changes I make which is also creates agile programming methodology environment since im making regular changes with feedback.

REDACTED will be my stakeholder and he is a 17 year old college student doing Maths Physics and Economics and he is a valuable stakeholder in my project because he will bring

a fresh perspective and understanding of the target audience I'm aiming for. He is an avid gamer that spends 2-4 hours a day on games so his input into my project will allow for meaningful feedback and he is not afraid to tell me what's good and bad about the game and where I need to improve on which means he is a valuable stakeholder in the development of my game. His input will ensure that my game aligns with the interests and demographic which will enhance the user engagement and keep them playing it. Having a college student also allows me to add into consideration the multitasking lifestyle and how a student has to balance their academic life with their personal life and what they do it in it to relax. This will allow for regular feedback from my stakeholder and they can contribute to refining the user experience the user gets and it also allows the stakeholder to address potential issues and problem early in the development process which allows me to change it easily rather than trying to solve it after making the game.

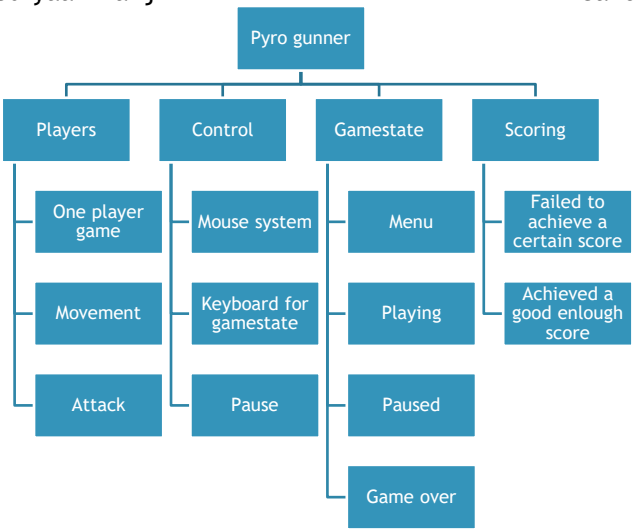
HOW TO SOLVE THE PROBLEM WITH COMPUTATIONAL METHODS

THINKING AHEAD

I need to decide what to do in my game that is necessary and what isn't. This is why thinking ahead is essential for me making the game.

- I am planning to make my game in Visual Studios 2022 and the justification for this is that its an easy to use compiler that allows for a more visual look for code which makes it easier for me to understand and look at what the stakeholders see when they play my game. I also started off with visual studios meaning I have prior experience in it so it makes it easier for the development process of my game
- The inputs for my game will only be Left arrow, Right arrow, Up Arrow, spacebar and P. The arrow input will be for controlling the player left and right and jumping and the spacebar will shoot for the player and the P will pause the game. The justification for this is that its very easy to use because it allows the user to play with keys that they may already be familiar with as its common inputs for the game im making. Simple controls also puts less stress on the player as its less to think about and my game is focused on the experience the player will have as I want them to be a recurring player
- The output would be the visuals and sounds effects of the game and the score indicator and health bar. The justification for this is that it will allow me to think ahead and plan expected outputs making the development process easier allowing me to meet my deadlines for the stakeholders such as REDACTED and also the exam board for the NEA project.

THINKING PROCEDURALLY AND DECOMPOSITION



The game has been broken down into 4 problems, by thinking procedurally, that should be solved in order of the game to have basic functionality.

PLAYERS

As this game is only a one player game, only the player can control the movement and attack and aim at enemies and it will have a visual of a character that will also have animations for the interactions of the player. The justification for this is that it will allow me to focus on things to make it the highest quality possible to meet my stakeholders requirements and standards

CONTROLS

These are necessary to interact with my game so the player should be able to click on the game to interact with it to allow the game state to change for example pressing “P” will bring up the pause menu. The justification for this is that the player needs to be able to play my game and they must be able to play it easily so making the controls clear and easy to use will create a better game environment for the user

GAME STATES

The game states will create an illusion of depth for the game as it gives the game some structure. There will be a start-up screen, a main menu, level selector and the actual levels. These give the player control over their experience and make it not feel like the game is playing it for them. Furthermore, if certain conditions are met in the game, it will change the game state to game over such as starting the game which will pop up with the start screen and allow the user to choose out of 2 options which are the controls screen and the level selector screen. It also changes game states to the pause, death screen and level finished screen when their conditions are met. The justification for this is that it will allow the user to have control of their game experience and do what they want in the game

SCORE

A score will give the game sense of achievement and different amount in score will trigger different events for example if the player gets a perfect score it will reward the player. On the other hand, if the score is too low, the player will not be able to go onto the next

level. The justification for this is that it will make sure there is difficulty in the game because its not allowing the user to just skip levels if they do bad in the level.

THINKING LOGICALLY

Thinking logically helps me analyse the situation and coming up with solutions that allow me to make it easier in development for the potential issues. The justification for me thinking logically about the way I make my game makes it essential because of the complexity of certain aspects of my game such as enemy AI and the death process for the player. As the game is running there will be a constant iteration running to check if certain conditions are met to end the game, which is also the game over state, and there will be a constant iteration of checking if enemies are being hit.

The main aspect of my game is that the game has a scrolling background which is a complicated process as I need to make sure not everything moves with the background and only certain things such as the player but not things like the score and health bar. The justification for this is that it allows me to come up with solutions to the broken-down problems making the game easier to program.

THINKING CONCURRENTLY

Thinking concurrently is the process of implementing parts of a solution or program concurrently or side by side for example by determining what parts of a problem can be solved at the same time. What I could do/make concurrently is updating the score and drawing the image at the same time.

The game im making will have a timer which will constantly be updating and the game has to render the background as the player moves to match what I made inside visual studios. The timer will help me with the background rendering as it will have the same tick intervals in which both update making it more a seamless game and the timer will also determines the players progress in the level.

THINKING ABSTRACTLY

Abstraction is the process of removing all the unnecessary features of something to focus on something that is essential to be able to make it to the highest standard but also allowing it to be less confusing and intensive on me programming the project as I have a lot of things I need to do. It will allow me to do things one by one without me being overwhelmed and is also a sort of checklist of what I need to do to complete my game. My justification for this is that my game is very complicated needing parts of it to be completely separate and allowing me to implement them one by one.

- I will add a timer to display how much time the player has to complete the level. The justification for this is that it will check how much time the player has been in the level for and whether the game should end if they take too long and I have a lot of aspects in my game that do not relate to the timer so it will help the game more simple and less complex.

- The player will only have Left arrow key, Right arrow key, Up arrow key, Spacebar and P for controls and the justification for this is that my game is just a two player game which does not require any mouse movement when in the level making it a lot less complex for me to code and a lot less complex for the player as they need to be using to input devices at the same time.
- I am going to add a pause feature to allow the user to stop playing but also save their current process and the justification for this is that it will not allow the AI enemies to attack and the timer to go down when the game is paused making it simple for the player to do if they need to stop the game.

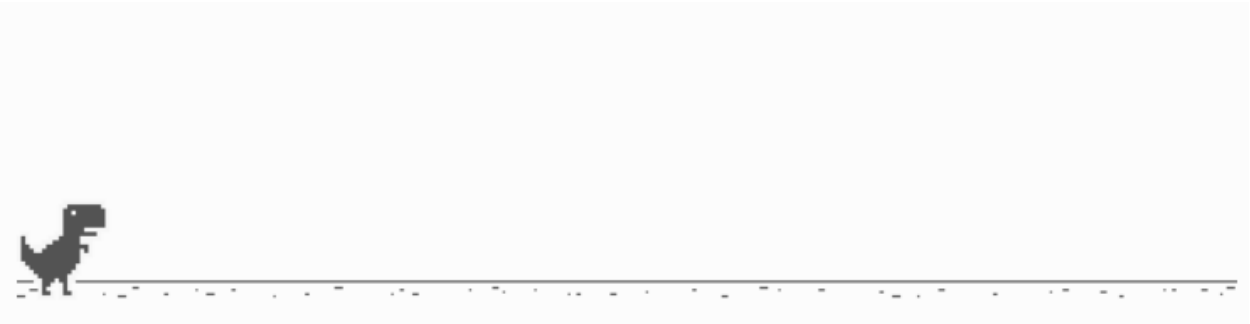
My overall justification for thinking abstractly is that a game like Mario is too complicated as its in the 3rd dimension and has a lot more controls so a simple approach is essential to the development of my game

CONCLUSION

The previous examples have shown that computational approaches can be used to solve the characteristics of my problem, making it appropriate for a computer program. These techniques are helpful for providing the issue some structure so that it can be resolved quicker, making it suitable for a computer program. If the problems are successfully solved using a computer program, the stakeholders will be able to play a game where they attack moving targets giving them points on their score and this is aimed to be a form of entertainment for the stakeholders.

RESEARCH

LOOKING AT A PREVIOUS GAME: GOOGLE'S NO INTERNET DINOSAUR GAME



T-Rex Game is an endless side scrolling game made by Google that they have implemented when you have no internet as a source of entertainment to kill time until you reconnect. It is a single player game because it only shows when you have no internet so it would be impossible for you to play online. The aim of the game is to get the highest possible score you can get while jumping

over cacti and dodging flying pterodactyls to stay alive. The game is based upon a 2D platform game, which my game will also be based upon.

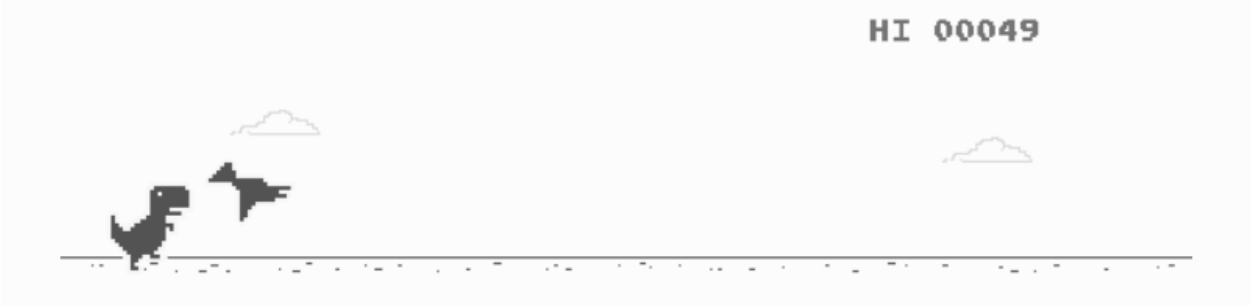
What I will be taking inspiration and changing in my game that is from the T-Rex game is the infinite scrolling background and the score concept. On the other hand, my game will have a pause feature, lives, higher difficulty, ability to move left, right, jump and stop while the T-Rex game is a constant run.

Controls: The game on desktop can control the dinosaur jumping up with the space bar or up arrow key but also ducking down with the down arrow key which is different to the mobile version as it only allows the player to jump and not duck.

Lives: The T-Rex game only has one life so if the dinosaur comes into contact with either cacti or pterodactyls it ends the game instantly and shows your score and a restart button.

This displays your current score on the right side and the highest score on the left and it saves each time you play it. The score is based on the distance you are travelling and the time as score from 0-100 roughly takes 10-11 seconds while score from 400-500 takes roughly 7-8 seconds.

After you come into contact with an obstacle the game ends and displays your score and replaces the high score if your current score was higher than the one that was saved. After clicking restarts, it resets you back to the start and resets your score to 0.



One of the obstacles in the game is the flying pterodactyl which either fly's low making you jump over it or it fly's high making you duck under it. If you touch it, the game will end showing the game over pop up.

The other obstacle in the game is cacti and they can either range from 1 large cactus to 3 grouped small cacti or a mix of both grouped together. Touching any of them ends the game

WHAT I LIKE ABOUT IT

Game features	Justification
Graphics	I like that the graphics are simple and are 2D, which makes the game look more retro. The game is made of a fixed platform/floor at the bottom of the screen with cacti on top ranging from a large single cactus to 2 small cacti together and flying pterodactyls.
Movement	The game has very simple controls, which are very user friendly and does not have an age limit as to whether who can play it as it only uses 2 keys. The up and down arrow keys are used to either jump or duck which makes it easy to play and understand the controls.
Score	I like that the game has a score to track your progression as to how far you have gone without dying which gives a sense of competitiveness because you want to keep trying to beat your high score.

WHAT I DISLIKE ABOUT IT

Game features	Justification
Pausing	The T-rex game has no pause feature, which is inconvenient because if you have reached a high score but you have to do something else, you have no choice but to end the game as it doesn't allow you to pause and come back at the same point as before.
Difficulty	I like that the game is simple but in my opinion, they have made it too simple because all you are doing is jumping or ducking so it doesn't have much sense of difficulty. Although it is a game to kill time while you wait to reconnect I wish it had other challenges to it.
Lives	In my opinion, I think having the T-rex have 1 life is a bit too harsh as anything from a small cactus to a large one can end the game. I think having 3 lives may have been better and when you come into contact with an obstacle, it should deduct how many lives according to the size of it for example, a small cactus takes 1 and a half health and a large cactus takes 3.

LOOKING AT ANOTHER GAME: STICKMAN ARCHERO

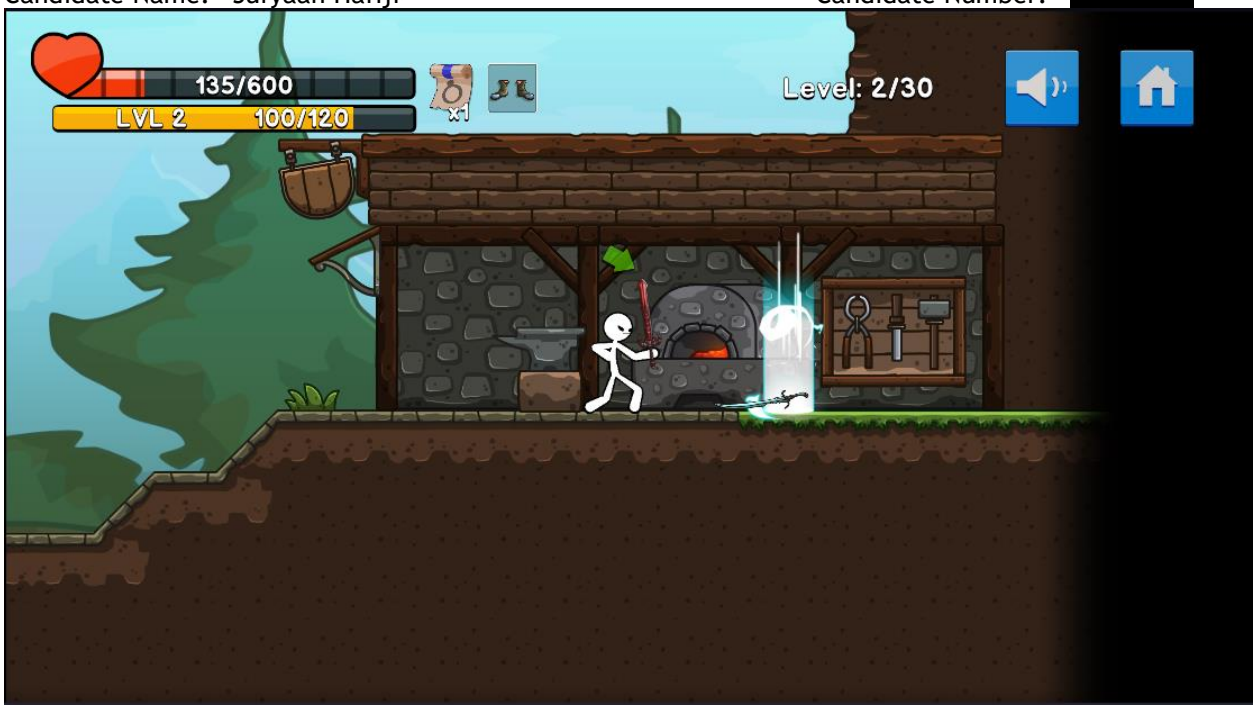
Another game I will be looking at is Stickman Archero Fight, which is a fighting game that allows the user to move left right up and down and you can fight enemies. This game is different to T-Rex game as it does not have an infinite background and is in a confined space. It has a lives system, a levelling system, an energy system and a gem economy. This game is a lot more complex than the dinosaur game and requires a lot of skill naturally making the game target age audience higher to teens and above but in this case, it is mostly targeted at teens.

The GUI is relatively user friendly and does not require you to learn or search up what they do and mean. When you start off, you have 50 energy in the game, which 5 is required each time you want to play and it regenerates on a timer only allowing you for example to play 10 times before you have to wait until your energy recharges. This is something I will not be implementing into my game, as I want the player to be able to play anytime they want and however many times they want unlike this game where it requires you to play, leave for some time to let it recharge and then comeback. This will detour people who want to casually play the game at any time without any commitment to it.

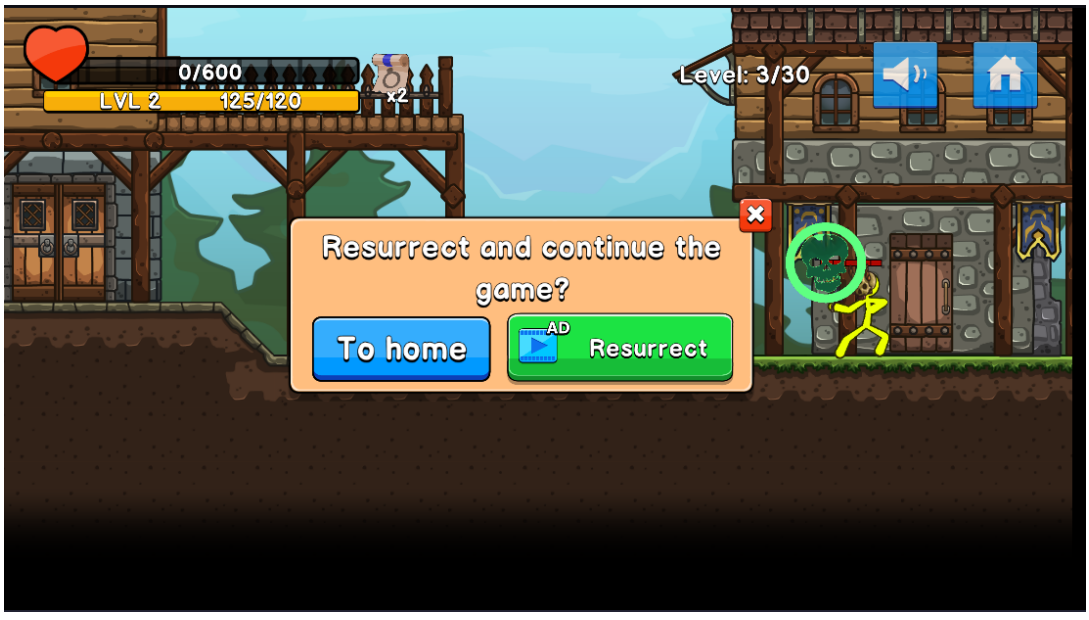
What I will be taking inspiration from this game is the health system and movement. I want to allow the player to move at their own pace but also have a bit more user activity and thinking by letting them move freely. The health system is great as it gives the player the intensity if they are on low health but also not too intense when they have full health.

When you first start the game, it gives you 600 health and you also start at Level 1 which will increase as you defeat more enemies but also depletes health anytime you get hit for example in the screenshot, 1 hit from the enemy took away 41 health. This game allows the user to jump, punch, pick up weapons and throw them.

Defeating enemies have a random drop chance of a weapon or armour, which will increase the players hit strength to the enemies greatly increasing how much health it depletes and the armour increases the health of the player.



After defeating enemies in the level, it allows you to go onto the next level by using a portal that changes the level when you come into contact with it.



After your health has depleted to 0, the game gives you the option to go back to the home screen or resurrect yourself by watching a ad. This gives the player a chance to carry on playing if they have reached a level they have not reached before giving a saving grace to the user.

WHAT I LIKE ABOUT IT

Game feature	Justification

Movement	Allowing the player to completely stop and run in any direction is a good feature unlike the T-Rex game which you have no control where the dinosaur is going
Health System	Introducing a health system is great for players as a 1 hit ending game generally is more frustrating and can detour players
Level system	A system where difficulty starts increasing is good for players who want to play something to ease them into the game but progressively gets harder challenging the player

WHAT I DISLIKE ABOUT IT

Game feature	Justification
Energy system	This can be bad for players who don't want to commit to games as having to wait to play a game can become annoying
Pausing	This game also does not have a pausing feature which I think is annoying as this is not a constant moving game like T-Rex game.

HOW I WILL USE EXISTING GAME FEATURES IN MY GAME

No.	Description	Justification	Game reference
1	Scoreboard	I need a way to help the player keep track of their progress so they are enticed to play the game again so they have a sense of difficulty that they need to beat	Offline T-Rex game
2.	Arrows for movement	I need to keep the game controls easy and something people are use to and they will be able to use the arrow keys as it's a common way to play games	Stickman archero
4.	Layout	For the layout of the user interface, im going to be using both games to get a reference with and combining both designs as I need a health bar but also a score which both game have but not together	Stickman archero Offline T-rex game
5.	Audio	Im going to be adding audio into my game so its not boring for the user to play the game and it adds a level of immersiveness to it	Stickman archero

INTERVIEW PLAN

Main points:

- What makes a game successful
- What type of user interface and graphics do you prefer
- What things do you not like about games

Questions:

- Have you played the games?
- What do you like about it?
- What do you not like about it?
- Would you go out of your way to play the game?
- What device do you usually play games on?
- What makes a game addictive?

INTERVIEW WITH MY FRIEND REDACTED

What do you think makes a game successful?

Something that would make a game successful would be the type of the game that could be played more than once and still be enjoyable.

What kind of graphics do you like in games?

I generally prefer pixelated graphics because I do not want a game to be too realistic as it is an escape from reality

Do you prefer detailed user interfaces or simple ones?

I prefer simple user interfaces because I would rather spend more time playing the game rather than getting accustomed to detailed ones

What makes you not like a game?

If the difficulty remains the same throughout the game, I feel like I'm playing the same thing over and over without any variation

Have you played Googles offline T-Rex game?

Yes

Whats good and bad about it?

The good things about it are it advances in difficulty the higher your score gets and the introduction of new obstacles. The bad things about it is that it looks too simple and bland

When do you play the game?

Only when my internet goes off

What device do you usually play games on?

Laptop

What do you think makes you want to keep playing a game?

How do you feel about the 1 health system in the game?

I feel like 1 health is too less because I can die by something like a small cactus or a big one and theres no variation on how much health it should take and it should not take as much health when you hit a small obstacle.

A game that has the right amount of difficulty and a sense of accomplishment when you advance

Have you played Stickman Archero?

I have not

INTERVIEW TRANSCRIPT AFTER HE PLAYS THE GAME

Did you like the game?

Yes

What was good and bad about it?

There was a variety of weapons to equip however there was a limited amount of controls.

What were the graphics like?

They were decent but could improve on the characters

What was good or bad about the movement?

The jumping felt very fluid and I could move and stop at any direction at any time

How was the difficulty system?

The difficulty was very balanced as you progress the enemies get tougher but you also get stronger through skills and equipment

When would you play this game?

After school when I got some free time

How intense did it feel and if not, how would you add intensity?

It felt intense when I was low on health but other than that if I'm on full hp it's not really intense. To make it more intense I would add a timer

INTERVIEW DATA REVIEW

Graphics: Pixel animations are more appealing to my target audience as they do not look for high quality representations of reality. The cartoons will be taken from a sprite sheet and those sprites will be moving across the screen and changing gifs each time it turns or jumps so its not one gif sprite that will stay the same any time it moves.

Background: I will have a moving background that will move when the character moves so it does not have an edge limit on the window until the end of the level.

Players: With my research, it shows that players prefer single player games when they want to enjoy their free time so I will not be implementing multiplayer in my game.

Instruction Menu: Make a clear and effective instruction page to show the user what the controls are, this is important, as the target audience are teens, therefore they need clear instructions so they are not confused.

Competitive: I will be implementing a scoreboard to my game so the player has a sense of competitiveness every time they play so they can try beating their last high score. The game will be increasing in difficulty each level so it can entertain the player so they do not get bored.

Characters: You have the option to pick a name for you player to include a personal aspect which will hook the player into the game, to ensure that there is a higher chance they will return to play this game again.

Timer: I will be adding a timer to my game to make it more intense so it will put the player under pressure and not just allow them to take however long they want.

Score: The score will be at the top left under the health bar displaying how many gems the player has collected when playing the level and it will also show after the player completes the level and multiplies it by the time so if the player gets a high score and not a lot of time it wont be a high score at the end.

SUCCESS CRITERIA AND REQUIREMENTS

SUCCESS CRITERIA

Success criteria	Justification
Power ups	**REDACTED has requested the difficulty of the game to increase gradually but also make the character strong with it so there is not a power imbalance**
Controls	**He has requested to use W A S D for movement and spacebar to shoot**
Timer	**He has requested a timer to add intensity to the game**
High score system	**The score system will be calculated by how many gems you have collected times the time left on the timer when you finish. For example 30 gems and 20 seconds left (30x20=60 score)**
Lives	**I will be adding a health bar, not as much health as Stickfight but enough that a few hits wont instantly kill you. I may also add power-ups to add more hp to the player.**
Score displayer	**There will be a score bar that will display the amount of gems collected, the timer and the total score.**
Different Menus	**The menu will be simple and not too complicated. There may only be 2-3 options on the menu so its easy to navigate and easy to start the game.**
7 Background	**Game must have a variety of background so it looks more professional and immerisve**
3 Levels	**The game will have 5 levels starting from easy then progressing harder and harder every level.**

5 Gems	**There will be around gems every level and they will be placed in easy to reach spots not blocked my enemies and some will be behind enemies you have to defeat**
5 Second Power-up timers	**Power-ups will not last permanently and will last for 10 seconds before reverting the character to normal as it was before**
20 Player damage	**When the player interacts with either the enemies of the void they should either die instantly or slowly start to die**

REQUIREMENTS

Requirements	Justification
Main menu and in-game menu	The game must have a main menu and a quick menu in-game to allow for navigation in the solution.
Easy controls	REDACTED mentioned that he didn't want complicated controls for the game so it must be simple to use
Levels	REDACTED mentioned that he wanted a variety of difficulty and a way to achieve that was by adding levels and making them harder as you progress
Background must be side scrolling	To make the game more immersive the background must not be static but it should move with the character
Timer	Essential for giving the player a challenging environment as they are trying to beat the time
Game over screen	When the user fails to complete the level in time or dies due to a enemy or the void a death screen should appear
Power ups	The game must have some sort of boost or power-up feature so it isn't stale and the player has a way to improve at the game, which in turn increases the sense of achievement and the overall relaxing experience.
Style of the game	REDACTED requested that the game not be too realistic in design and should have a cartoonish style to it
Animations	To not make the game looks boring I need to add animations when the player moves and shoots

FEATURES OF THE PROPOSED SOLUTION

Feature	Explanation/Justification
Character/Player	• Im going to have a pixel art style character for my game that will be a GIF so it's an animation. The justification of this is that the stakeholder for my game wanted a cartoonish design for my game so I am fulfilling the requirements set by him. • The player will have 2 controls for going left and right and it will only be limited to that as it's a 2 dimensional game that and it also makes the game less complicated if I had more movement options • When the player presses spacebar it will shoot a fireball and it will be a quick speed fireball which will make it smooth as possible but it will also be invisible at the start of the game so I don't need to keep creating the fireballs every time they press space and can just keep reusing the same fireball. • The character will have different sets of animations for idle, moving, shooting and jumping. This will create a more immersive experience than a non picture moving character as it will change depending on the button that is pressed. • The player will instantly die if the healthbar reaches 0 or the player jumps into the void and will change game states.
Background/world	• The background will be nature themed and each game state will have different type of forests so it seems like a mysterious game to make it more immersive but also have the fiction theme the stakeholder wanted of a cartoonish theme • The background will be moving as it's a side scrolling platform game meaning the background will be rendered as the player moves in the game
Obstacles/Enemies/Parkour	• There will be 2 obstacles in my game and one of them will be the AI enemies and the other will be actual parkour for platforms which the user can hit and miss and fall into the void. This adds difficulty to the game as its not just about enemies but the player also needs to have a certain amount of skill to time their jumps. • I may have a variety of enemies but starting off I will have just 2 enemies

	that constantly move backwards and forwards. I might add a constant shooting enemy but that's only if I can make the game within the timeline with spare time left. • Platforms will be solid and the player can not fall through them and they can also hit them when jumping making them fall
Power-Ups	There will be 2 powerups starting off when I make my game and they are: • Health boost. The health boost will give the player some health back when they interact with the object • Speed boost: The player will have a increase in speed when they interact with the speed boost object These may change as I progress with the development of my game
User Interface	• There will be a starting screen to my game which can lead to either the controls menu or the level selector menu. I have made the menus as simple as I can to make it easy to navigate for the user but also a more simplistic style can make the game more enjoyable as things are just not crammed in. • There will be a runtime timer that will show how long the player has to finish the level and also executes a procedure if the counter gets to 0. • There will also be a gem counter and a ammo counter which will help the player keep track of how much ammo they have left and how many gems they have collected in the level which may be needed to unlock their next levels.

HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

Requirements	Justification
Keyboard	To control the player and navigate the game
Mouse	To control and navigate the menu
Monitor	To display the visuals of the game
1.8 GHz or faster x64 processor	To run the game with no lag
Minimum 4GB RAM	For the game to be able to load things while its running
Video card that supports minimum 1366x768 resolution	To be able to display the game

SOFTWARE REQUIREMENTS

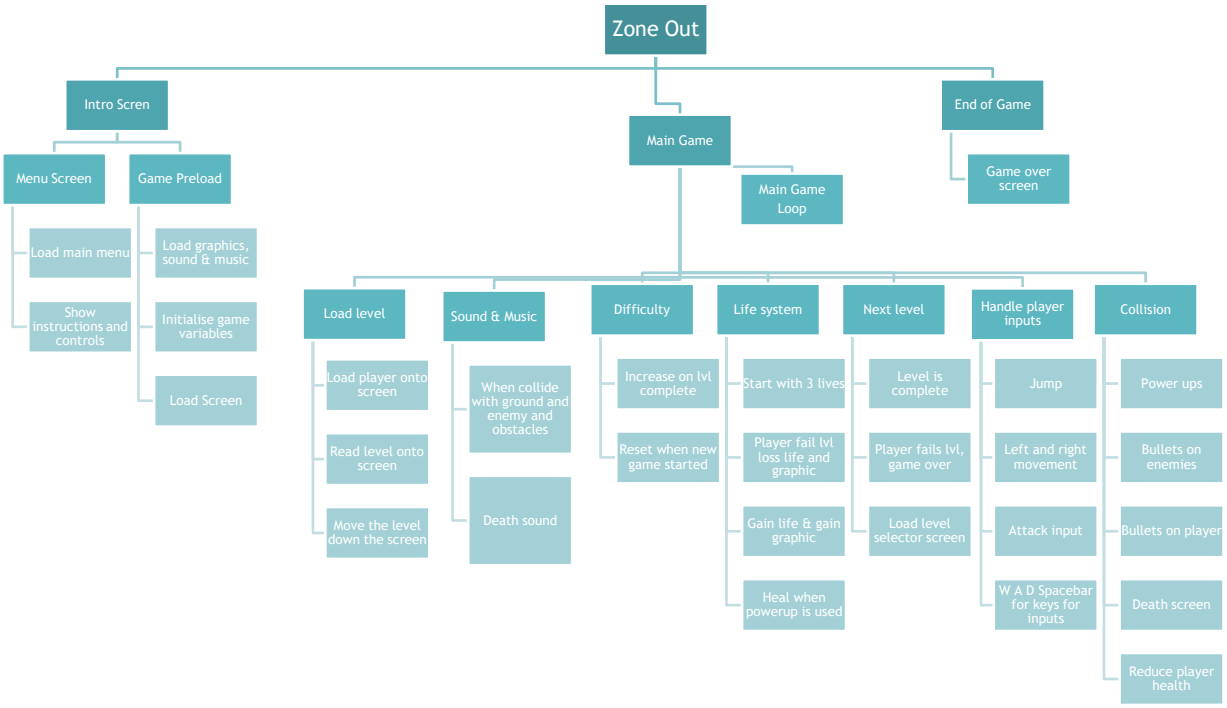
Requirements	Justification
Windows 10/11	Need a operating system to run the game on
Visual studios 2022	To be able to run the game

LIMITATIONS

Limits	Reason for limitation	Justification
Feedback and iteration	Limited feedback or opportunities for iteration may restrict the refinement of the analysis. Ongoing feedback and iteration are essential for improving the quality of the project	Feedback is essential for improving the game but having constant feedback from a busy stakeholder can be a limit as they may not have enough time
Bias	Bias may affect the game as I am only interviewing and getting one person to play the game so I would need to get a handful of people to get feedback on the game.	As im only using 1 stakeholder it will not be a true evaluation of my game and a authentic review as its someone I know
Scope	If I make my game too hard or technological it can mean that I don't meet time deadlines	In my timeframe it wont be viable for me to make a more complicated and a bigger game
Time constraints	Time constraints are a big part of the project as if I make the game too big it will be hard for me to meet the stakeholders expectations	As I got deadlines for my stakeholder and also my exam board I will need to be able to plan things accordingly in a timely procedure
Accessibility	My game will not be able to be played on any other devices except Microsoft desktops so it limits how many people can play my game	It will be too hard for me to make my game multiplatform in the timeframe I have
Diversity	My age range may be to concentrated on the teenagers and it may not even be catered to them and other age ranges may like the game	It will take too long and too many people to be able to have a more diverse scope of a target audience
Technological	I am limited to using Microsoft visual studios as my hardware cannot handle or be able to produce anything on a better engine	I only have a low end computer that can handle visual studios and not better game enginges such as unreal engine

B. DESIGN

SYSTEMS DIAGRAM



EXPLAINATION OF EACH MODULE

SCOREBOARD

New High Score

This places the user's final score on the leaderboard in its suitable position to correspond with other final scores and it needs to replace the old high score

New high score name and validation

This checks if the users name inputted meets the requirements and checks if the final score is bigger than the scoreboards lowest score.

JUSTIFICATION FOR SCOREBOARD

The justification for the scoreboard is to meet the requirements of the stakeholders as they need a sense of difficulty so they need a scoreboard to have goals to beat and try to improve on

PAUSE

Pause

When the "p" key is pressed, it changes the game from playing state to the paused state

Un-paused

When the “p” key is pressed again, it changes the game from the paused state to the playing state

JUSTIFICATION FOR PAUSE

The pause state is a requirement from my stakeholder so I need to add it into my game and it needs to be able to save the current game state so they can carry on whenever they want to because they may need to stop but want to carry on later

MENU

Level selector

By selecting a level in the level selector, it will change the game state from menu state to the playing state

Controls

By clicking the control option on the menu, it will give a description and change the menu state to the control menu state

Audio

By clicking the audio option on the menu, it will turn the audio on and off

JUSTIFICATION FOR MENU

The justification for this is that the stakeholder requested that it be a simple main menu that allows for easy navigation for the player and it makes it a more immersive experience as its simple and compact

GAME CONTROLS

Back to menu

By pressing the “exit” option, it will change the game state from playing to the menu state

JUSTIFICATION FOR GAME CONTROLS

It needs to be easy and accessible for the player to be able to navigate through game states so adding back to menu buttons make it easier for them.

GAME OVER

Game over display

If the players health reaches 0 it will change the game state from the playing state to the game over state and display the graphics

JUSTIFICATION FOR GAME OVER

If the player health reaches 0 or they fall into the void or the time runs out the game needs to stop and change screens for the player to know that they have failed and also allows them to try again or go back to another menu

START UP

Title of the game

The startup menu will show when the game is launched and prompts the user to press, “Enter” to get into the menu state

JUSTIFICATION FOR START UP

The justification for this is that the player needs to be able to see and be immersed into the game as soon as they launch the game and displaying the game name and simple buttons

PLAYING

Initializing game

When starting brand new game/level variables like lives, bullets and power up will be reset to 0

GRAPHICS AND ANIMATIONS

Jumping animation

When the “W” key is pressed, it will change the characters sprite to a jumping animation and will change back once it has landed back on the ground

Running animation

When the “D” is pressed down, it will change the characters sprite to running right, when not pressed it will change to a standing stance sprite, and then when the “A” key is pressed it will change the character sprite to running left

Dying animation

When the character lives reaches 0 they will “die” and this results in the characters sprite changing to a death animation.

SOUND

Gunshot sound

When the user clicks the attack/shoot button it will play a gunshot sound, display a field sprite of a bullet moving, and will play an empty click if the character has no bullets left.

Dying sound

When the player reaches 0 lives it will display the dying animation and will play a **(INSERT DYING SOUND HERE)**

Coin pickup sound

When the user picks up coins, it will make a sonic coin pickup sound

SCORING SYSTEM

Coin worth

Every time the player picks up a coin, it will add 1 to their score and if they pick up a big coin, it will count as 3 coins.

Time points

The amount of time a player spends in a level will be multiplied by the amount of coins picked up and this will be the total score for the leaderboard.

BULLET SYSTEM

Bullet removal

When the user clicks the attack button it will decrease, the bullets value by 1 until it is zero and then will play the empty click sound. The bullets sprite will be removed or turned invisible every time the attack button is click and once the ammo has run out, the ammunition box will changed the sprite to a empty box.

Bullet empty

When there is 0 ammo left the fireball needs to stop shooting when they are pressing the spacebar and it should only fire when the player gets ammo again

LIFE SYSTEM

Life removal

Each time the user plays the game it will reset the lives variable to 3.

CONTROLS

Go to main menu

If the main menu button is pressed, it will change the game state from whichever state it is in to the menu state

Pause

When the “P” key is pressed, it will change the game state from the playing state to the paused state

POWER UPS

Health boost

When the player picks a power up, if it’s a health boost it needs to add onto the player health.

Speed boost

When the player picks a power up, if it’s a speed boost it should increase the speed of the player

JUSTIFICATION FOR PLAYING

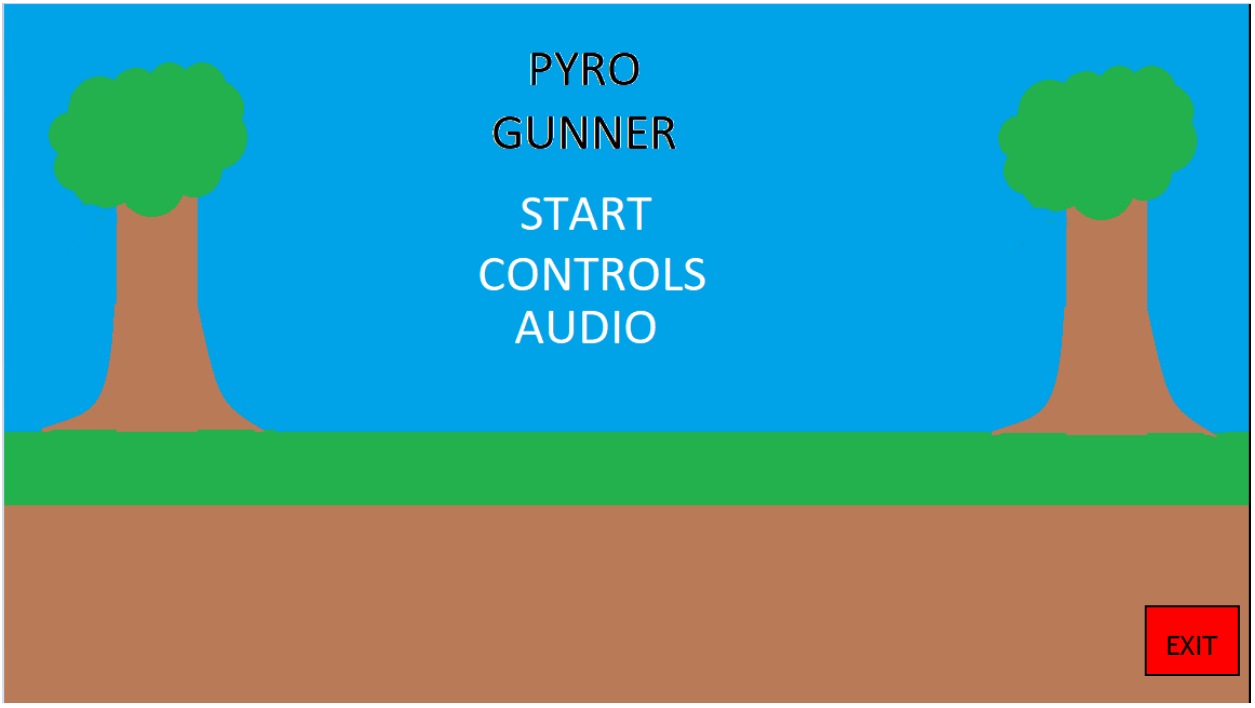
This is the main part of the game and this is the main focus of what I need to do and meet the requirements of the stakeholder. The success criteria mainly relies on this aspect of the game and these modules from the systems diagram help me achieve this

VISUAL REPRESENTATION OF THE GAME IDEAS

GAME START

- Introduction screen loads when game is loaded
- Player will have to press enter to start the game to get to the main menu
- In the picture, you can see my first design of what the start menu may look like in my game. I have shown two main parts of the graphics, which will be the platform and the sky.

HIGHSCORE



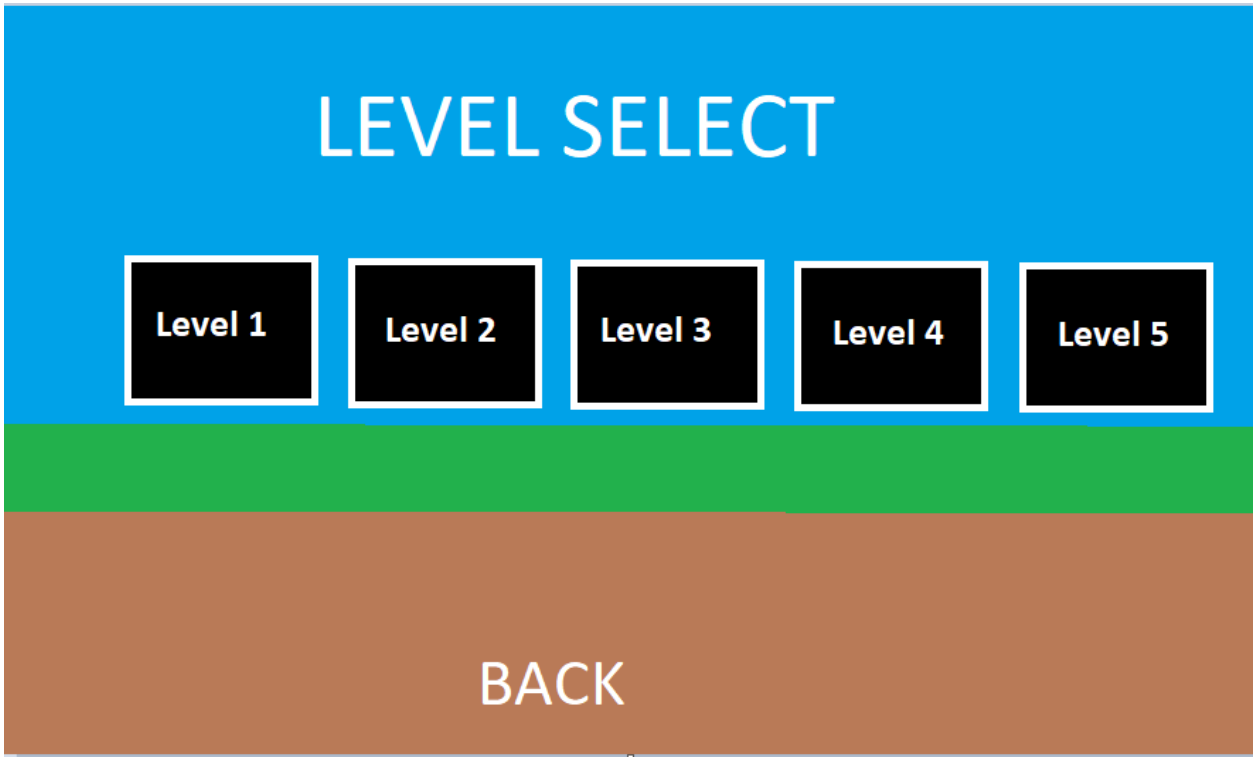
MAIN MENU

- This screen is shown after the player presses enter and unloads the game starting screen
- In the design of my game it will have 3 options which are: Start, Controls and Audio
- When clicking on any of these options it will show the corresponding page except for audio which will just show AUDIO or AUDIO with a streak through indicating audio is off
- There will be an exit button on the main menu to give the option to the use to quit the game

EXIT GAME

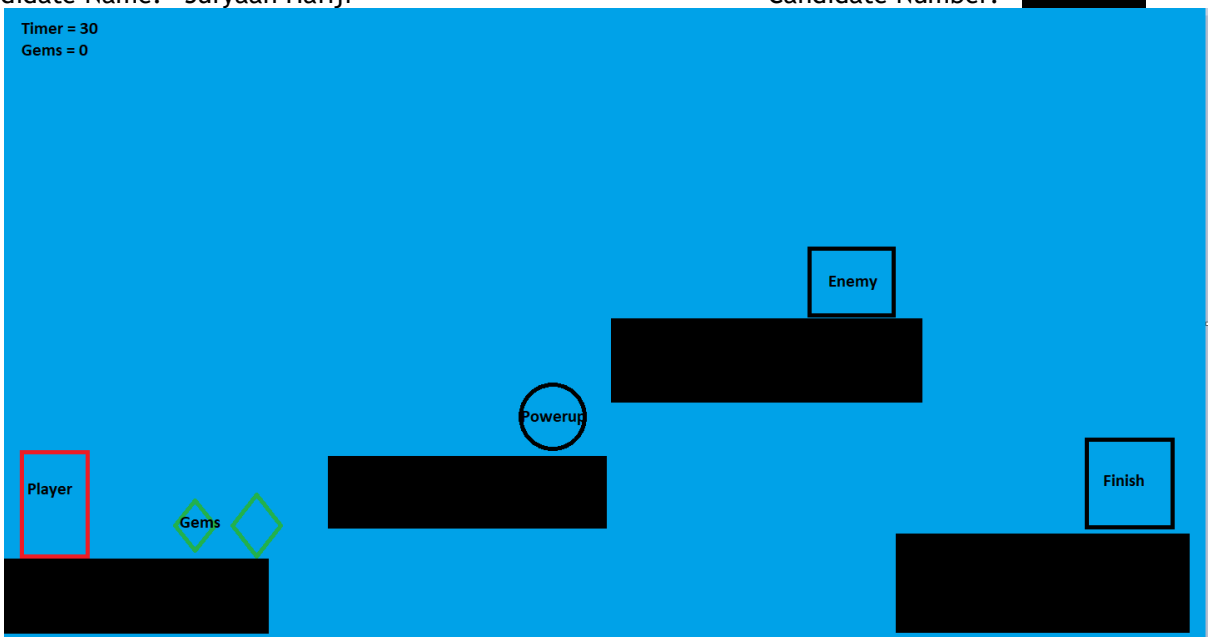
- When the exit button is pressed on the menu it unloads all of the screens
- Goes to desktop

- This button will be the only exit button for closing the game



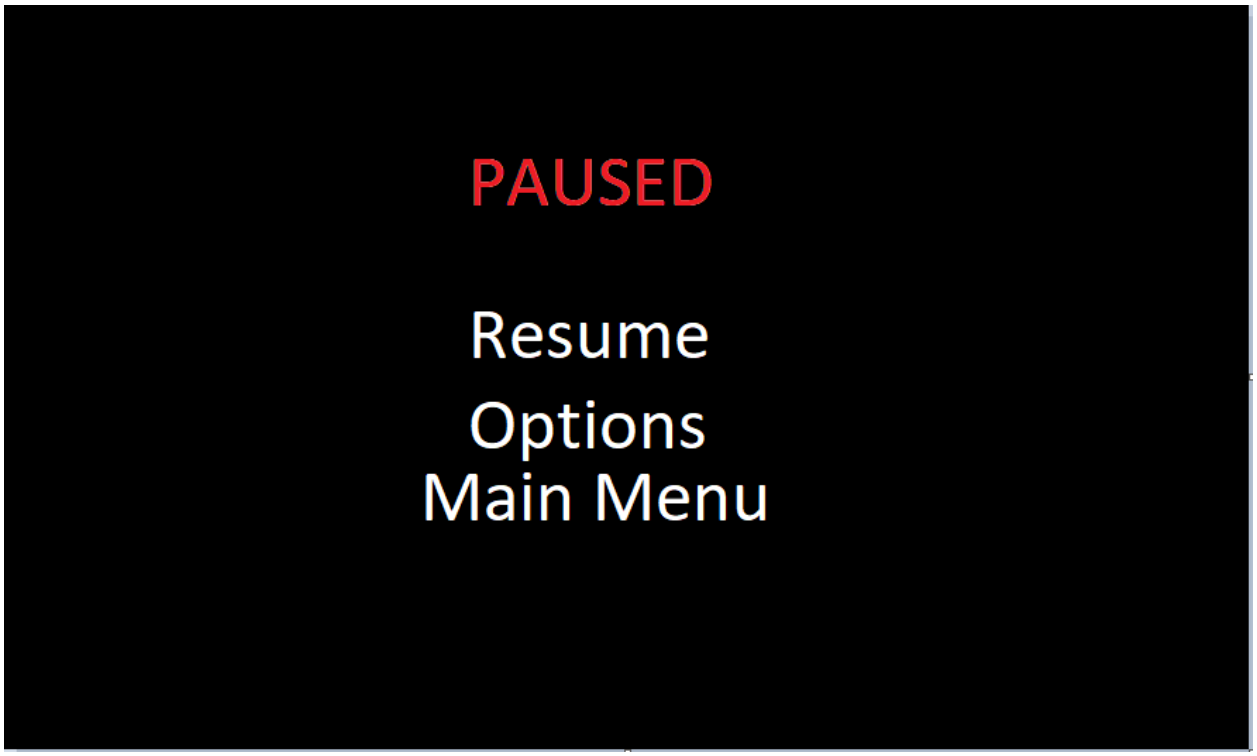
LEVEL SELECTOR

- This screen loads when the start button is pressed on the main menu
- The main menu is unloaded when this page is loaded
- This design will only have 5 levels to give an example of how the game may look when you click start
- There is a back button to allow the user to go back to the main menu if they want to
- There may be a leaderboard option I may add into the game on this page which allows the player to see all the high scores
- When a level is clicked, it will either glow or be enlarge



IN GAME

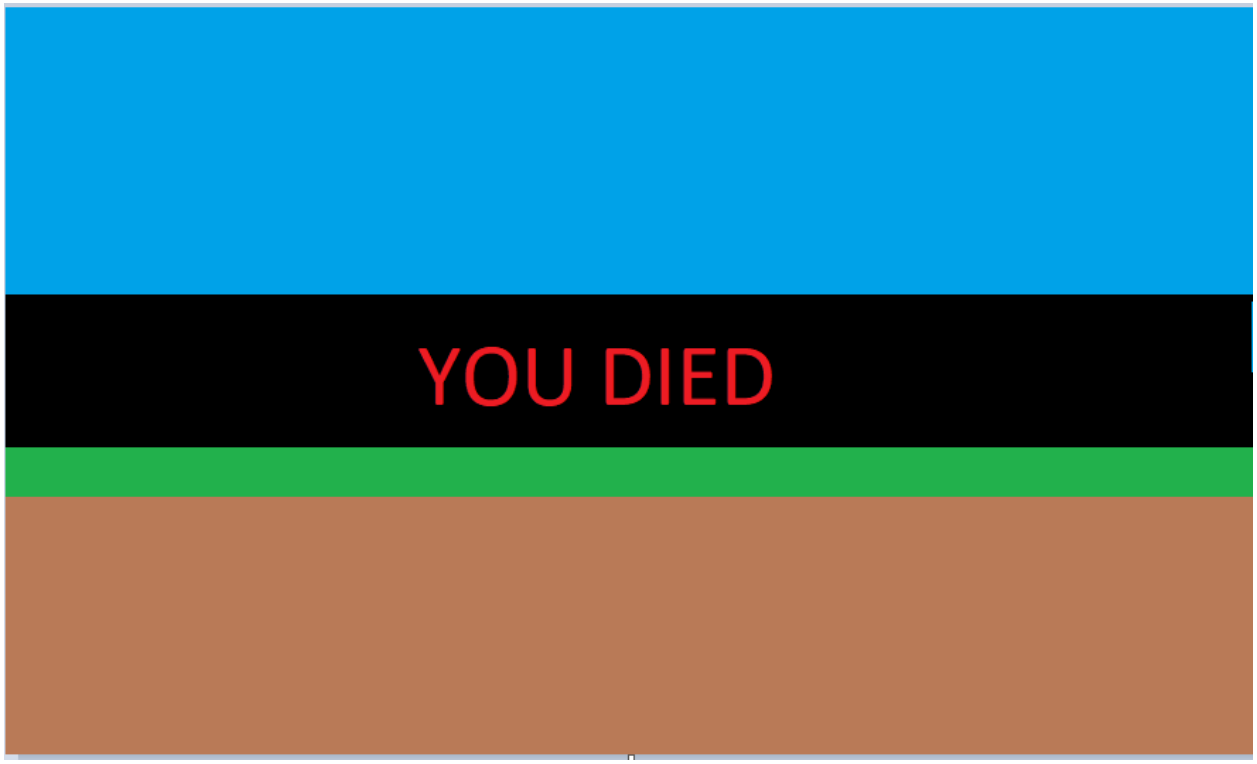
- This is a rough sketch of what I want the idea of my in game screen to be like and ive outlined what type of objects the player will interact with
- Gems will disappear when the player interacts with them and adds onto the gem counter
- Timer will tick down by 1 every second
- The powerup will disappear when its interacted with and will do what its designed to do
- Enemy will only disappear when its been killed by the player
- When player interacts with finish it will switch screens to the finish screen



PAUSE MENU

- My game will allow the user to pause the game whenever

- When the P key is pressed this screen is loaded
- Pause screen only works in game
- Displays 3 options which are: Resume, Options and Main Menu
- The resume option resumes the game and loads back the game screen
- The options button loads a screen where it shows the controls for the game and the audio button
- Main Menu button unloads the game screen and pause screen and take you back to the game main menu

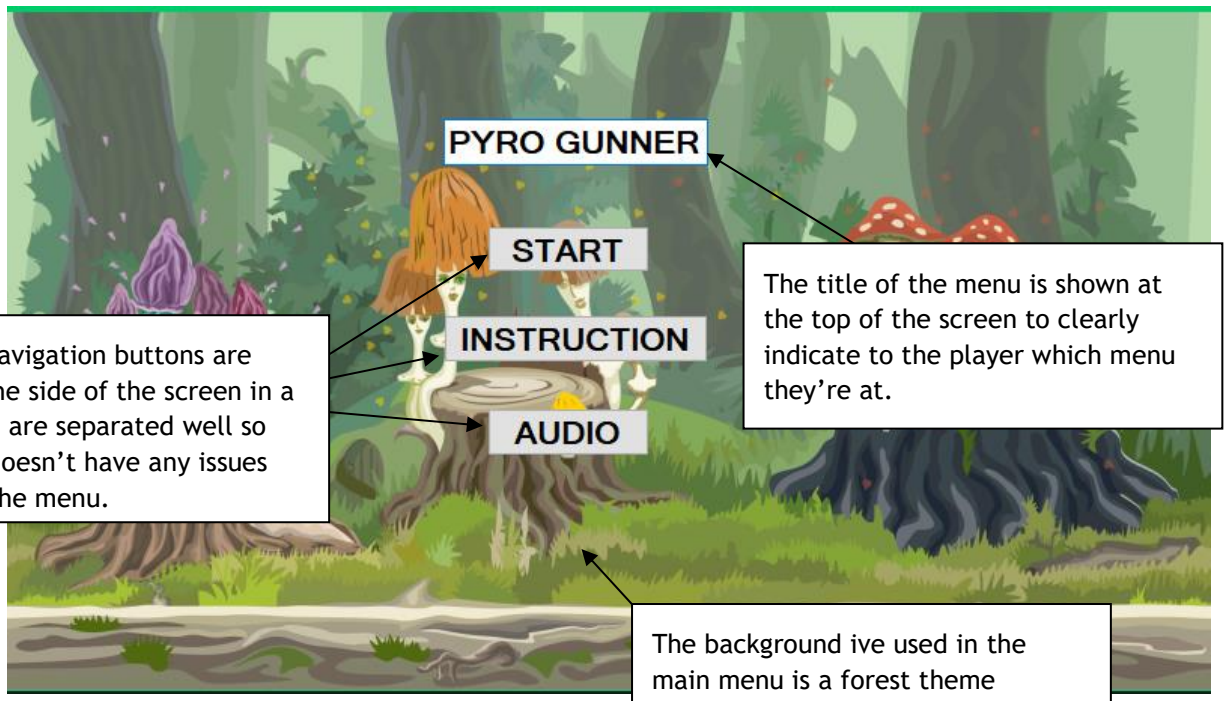


DEATH SCREEN

- This screen will show when the player dies in the game
- After this screen has loaded it will display another screen which will allow the user to retry the level or go back to the level selector

USABILITY FEATURES

MAIN MENU

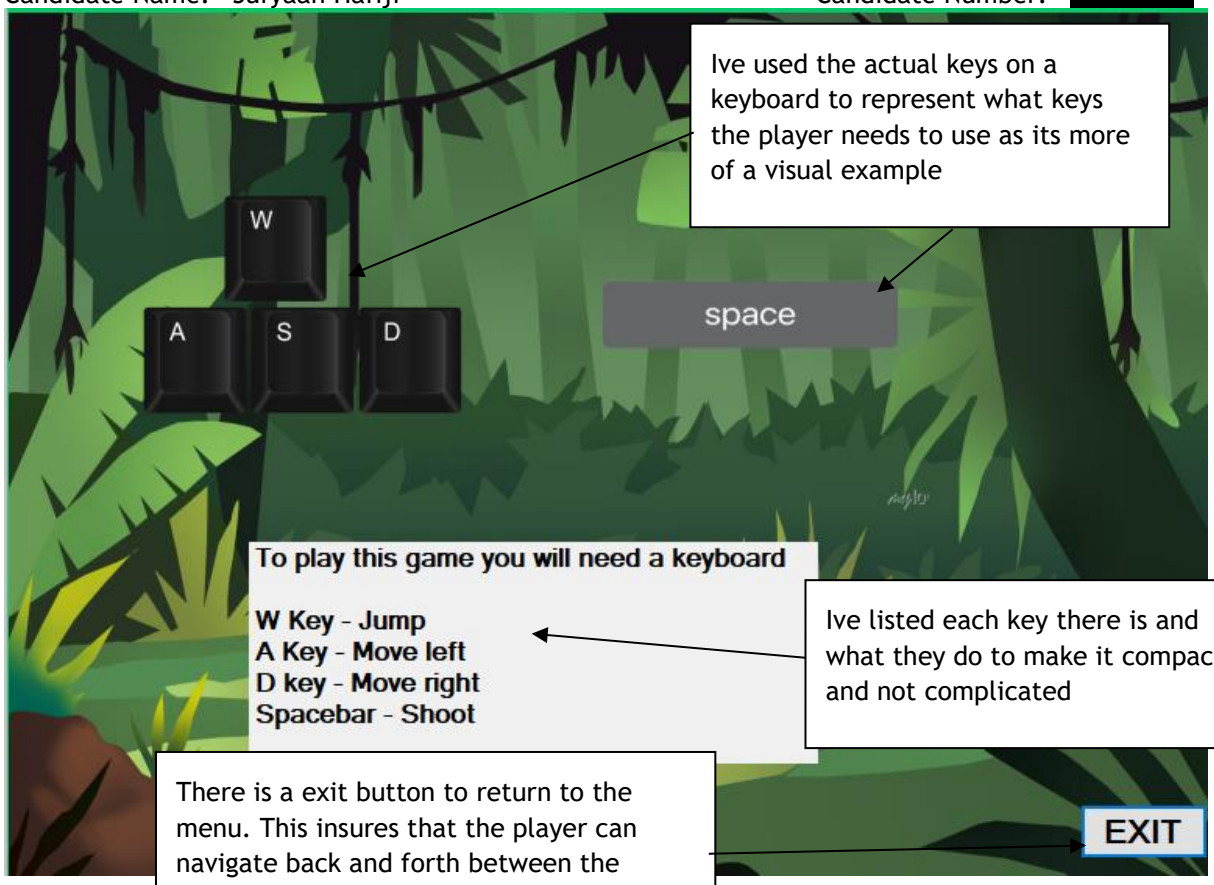


The menu navigation buttons are placed on the side of the screen in a big font and are separated well so the player doesn't have any issues navigating the menu.

The title of the menu is shown at the top of the screen to clearly indicate to the player which menu they're at.

The background ive used in the main menu is a forest theme background which lines up with the stakeholders requirements of it being cartoonish mysterious game

CONTROLS MENU

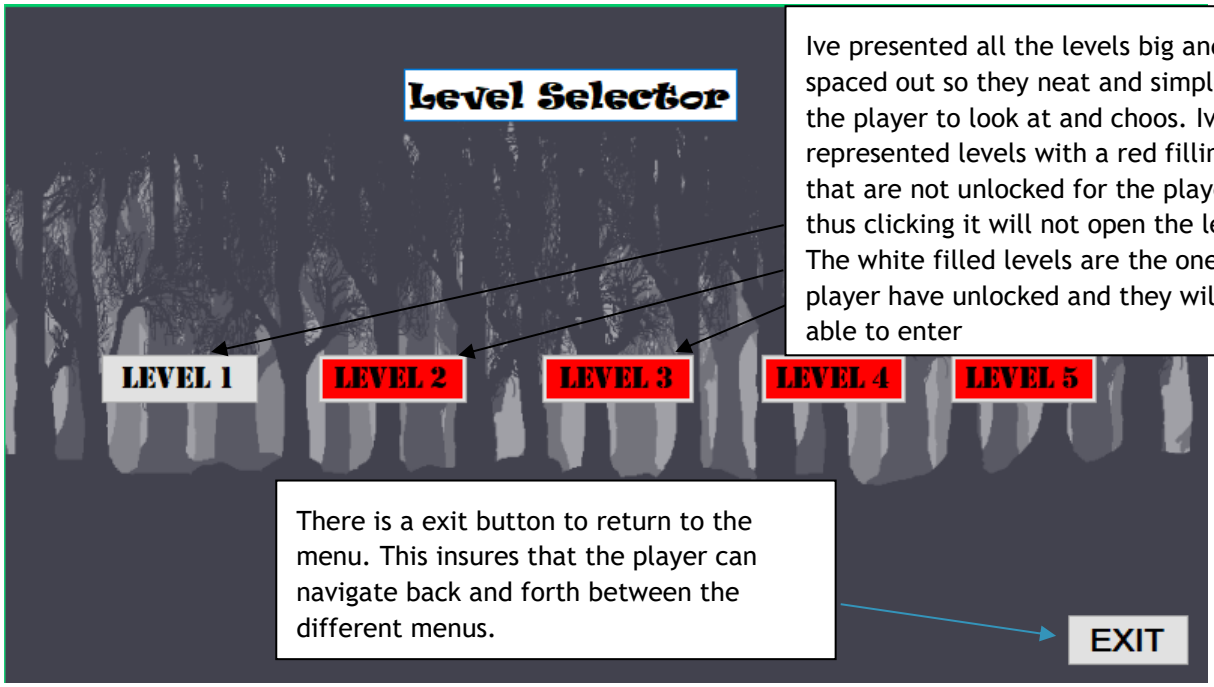


Ive used the actual keys on a keyboard to represent what keys the player needs to use as its more of a visual example

Ive listed each key there is and what they do to make it compact and not complicated

There is a exit button to return to the menu. This insures that the player can navigate back and forth between the different menus.

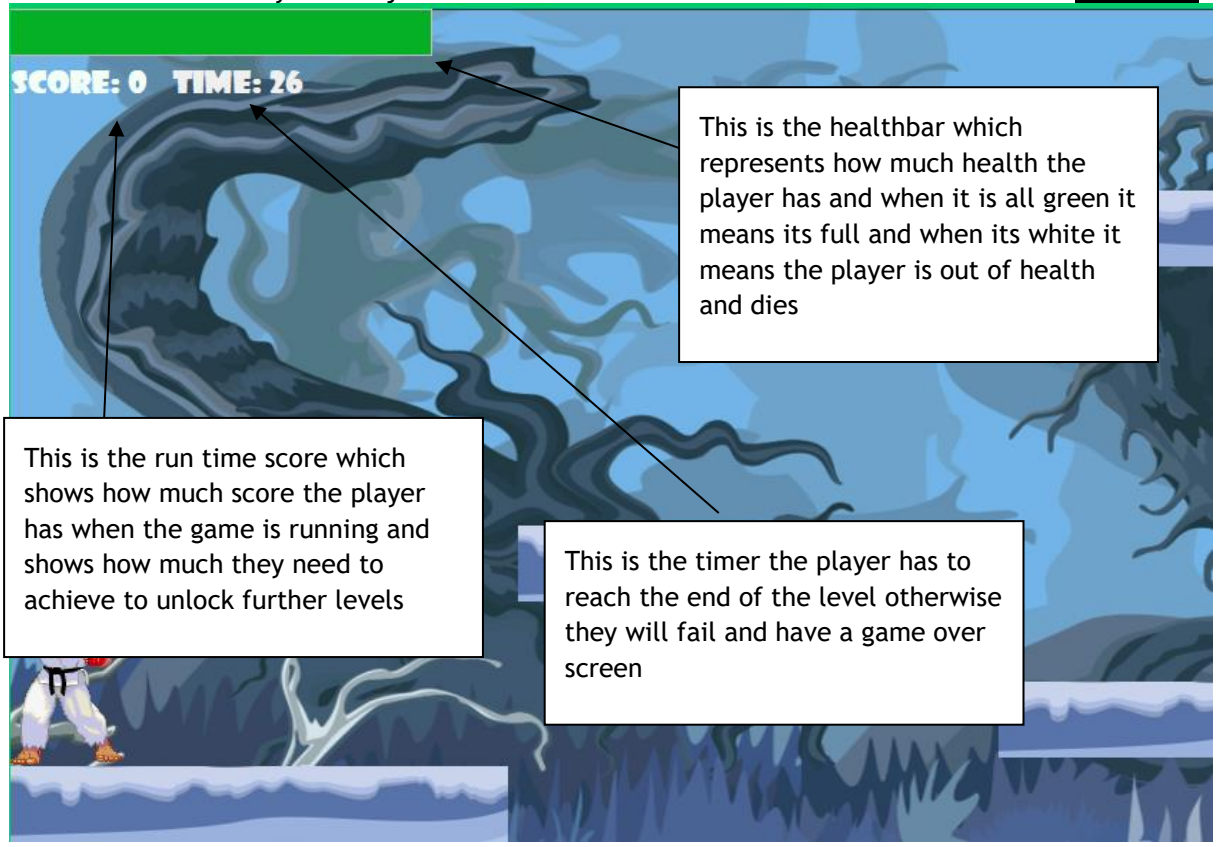
LEVEL SELECTOR MENU



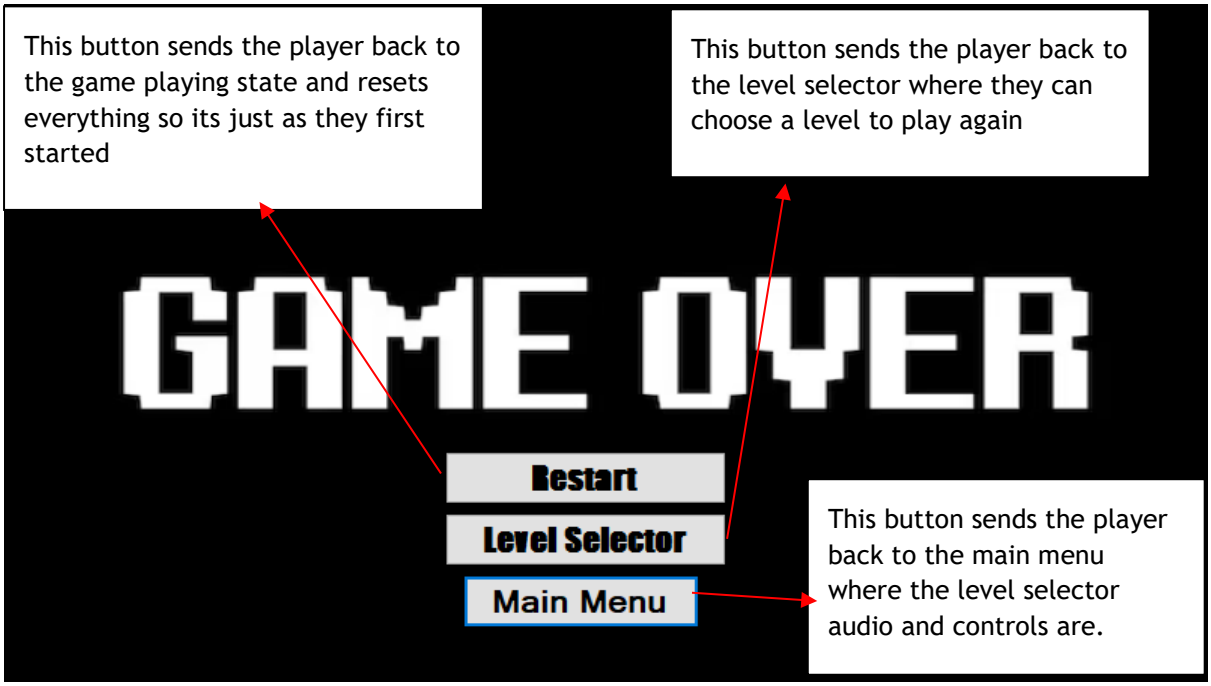
Ive presented all the levels big and spaced out so they neat and simple for the player to look at and choos. Ive represented levels with a red filling that are not unlocked for the player thus clicking it will not open the level. The white filled levels are the ones the player have unlocked and they will be able to enter

There is a exit button to return to the menu. This insures that the player can navigate back and forth between the different menus.

GAMEPLAY MENU

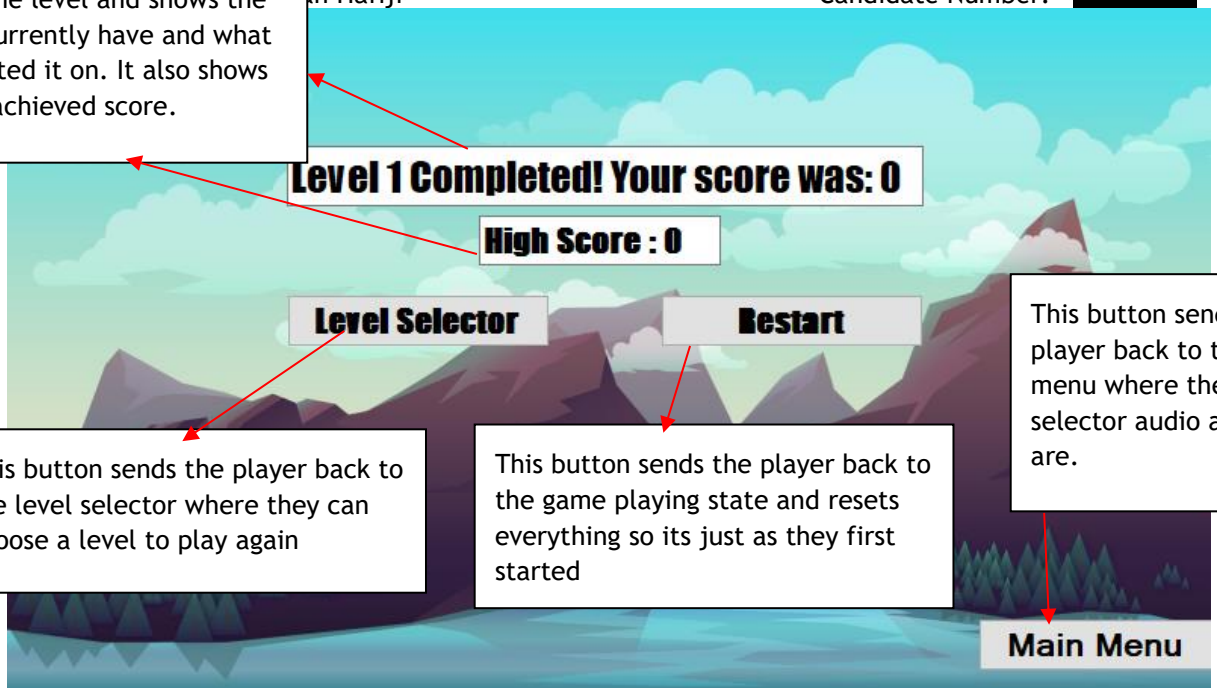


DEATH SCREEN MENU



LEVEL FINISHED MENU

This shows the player that they completed the level and shows the score they currently have and what they completed it on. It also shows the highest achieved score.

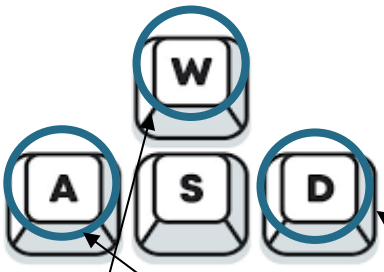


This button sends the player back to the level selector where they can choose a level to play again

This button sends the player back to the game playing state and resets everything so its just as they first started

This button sends the player back to the main menu where the level selector audio and controls are.

CONTROLS



W key is used for the player to jump

A key is used for the player to move left

D key is used for the player to move right

Spacebar key is used to attack/shoot

P key used for pausing game

KEY VARIABLES AND DATA STRUCTURE

Name	Instance	Data type	Explanation	Justification
goLeft	variable	Bool	It's a Boolean variable that checks if the left arrow key has been pressed or depressed and it sets it to true or false depending on that	The game needs to be constantly checking whether the player presses the arrow key to execute procedures that are linked to that key
goRight	variable	Bool	Checks if the right arrow key has been pressed or depressed and it sets it to true or false depending on that	The game needs to be constantly checking whether the player presses the arrow key to execute procedures that are linked to that key
Jumping	variable	Bool	Checks if the up arrow key has been pressed or depressed and it sets it to true or false depending on that	The game needs to be constantly checking whether the player presses the arrow key to execute procedures that are linked to that key
jumpSpeed	variable	Integer	Sets the speed on high fast the player jumps up	This is so I can control how fast the player jumps and adjust it
Force	variable	Integer	Sets the force on the player when they jump	This is so I can control how high the player jumps when they press spacebar
Score	variable	Integer	Saves the score as a number variable	This is to keep track of how the player is

				doing playing the game
Ammo	variable	Integer	Saves the amount of ammo the player has as a variable	This is to keep track on how much ammo the player has and to adjust it
Gameover	variable	Bool	A variable which checks if the player has died or timer has ran out	This is to check if the player has died or the timer has ran out to execute procedures that link to it
Shotfireball	variable	bool	Variable to check if the user has pressed the spacebar button	Checks if the player has pressed the button to execute code that shoots fireballs
Pause	variable	bool	Checks if the player has pressed the P button	This is to keep checking if the player has paused the game to execute code to pause the game
Unlocked	variable	Bool	Checks if the player has completed the first level	If its set to true the 2nd level is unlocked to play
Scoretimer	variable	Integer	Sets the score timer to 30 every time its ran	Keeps track of the timer and sets it to 30 every time
playerspeed	variable	integer	Sets how fast the player moves left and right	This is to set the player how fast they move and to adjust it
Playerhealth	variable	Integer	Sets how much health the player has when they start	This is set to a certain amount at the start and can be adjusted
backgroundSpeed	variable	Integer	Sets how quickly the background scrolls	Is a set amount of speed that the background scrolls

Enemy1health	variable	Integer	Sets how much health the enemy has	Allows me to adjust how much health it has and how hard it is to kill
Fireballspeed	variable	Integer	Sets how quick the fireball moves when its shoots	Lets me adjust how fast it goes
Highscore	variable	Integer	Saves the highest score the player reaches	Is a integer and not float because cant have half a score
isgrounded	variable	bool	Check if the player is in the air or on a platform	Executes code if the variable is true or false

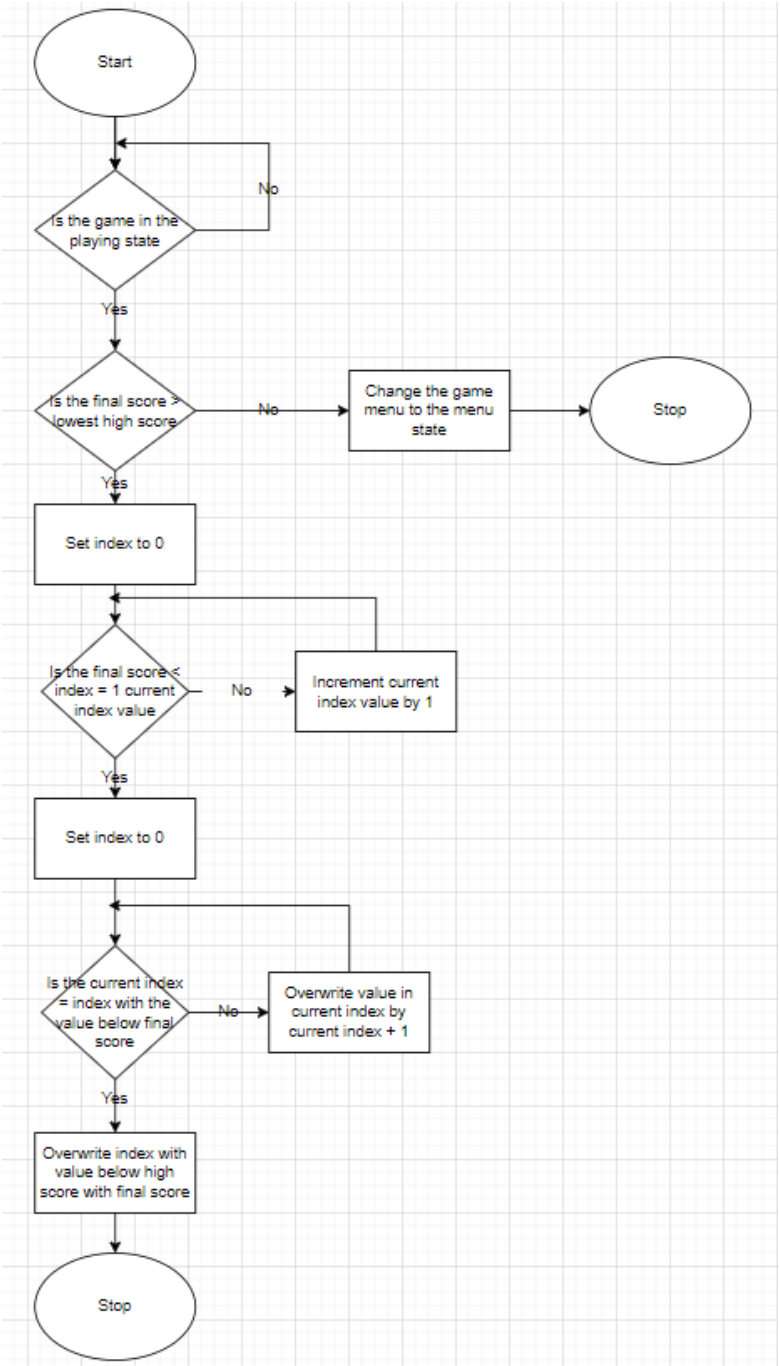
Method	Name	Data type	Explanation	Justification
Scoreboard				
Game state	Scoreboard		The game is at a part where the high score and leaderboard can be added	This is where all the code related to the scoreboard is
Procedure	High score		A new high score is added to the score board	This will give the game a competitive aspect as players getting better new high scores will be created
Variable	Score	Integer	This is the users score	This will allow the value of the users score to be used for game logic and the leaderboard
Array	High score array		This is the structure used to organize the scoreboard data	This will allow the new high scores to be rearranged
Text file	High score file	String / .txt	This is to the score the high scores of the player	This will allow the high scores to be saved and be changed
Paused				
Game state	Paused	Boolean	The game is at a part where it is paused	This is where all the code related to the pausing is
Procedure	Un-paused	Boolean	The game state changes from playing to paused	This will allow the user to

				resume playing the game
Menu				
Game state	Menu		The game is at a part where the controls are displayed to the user	This is where all the code related to the games controls is
Procedure	Controls		The controls are loaded as an image with a description	This informs the user what the controls are
Game over				
Game state	Game over		When the player dies it enters the game over state	This is where all the code related to the game over is
Procedure	Gamer over display		A game over image is loaded up	This shows the user that the game has ended
Start up				
Game state	Start up		The game is at a part where the game is initialized and the title of the game is displayed	The code is where all the code related to the games start up is
Procedure	Title		Game title is loaded up as an image	This tells the user what game they are playing
Playing				
Game state	Playing		The game is at a part where the actual game is being played	This is where all the code related to the playing of the actual game is
Procedure	Initialize		The game is initialized	This is to set all the variable to there
Procedure	Pause		The game state is changed from the playing state to the menu state	This makes the game more convenient as the players can start and stop the game whenever without losing progress
Procedure	Back menu		The game state changes from the playing state to the menu state	This will allow the user to switch between levels and looking at controls
Variable	Lives	Integer	A variable with the value of 3	This gives the user a finite amount of lives making the game more challenging and intensive

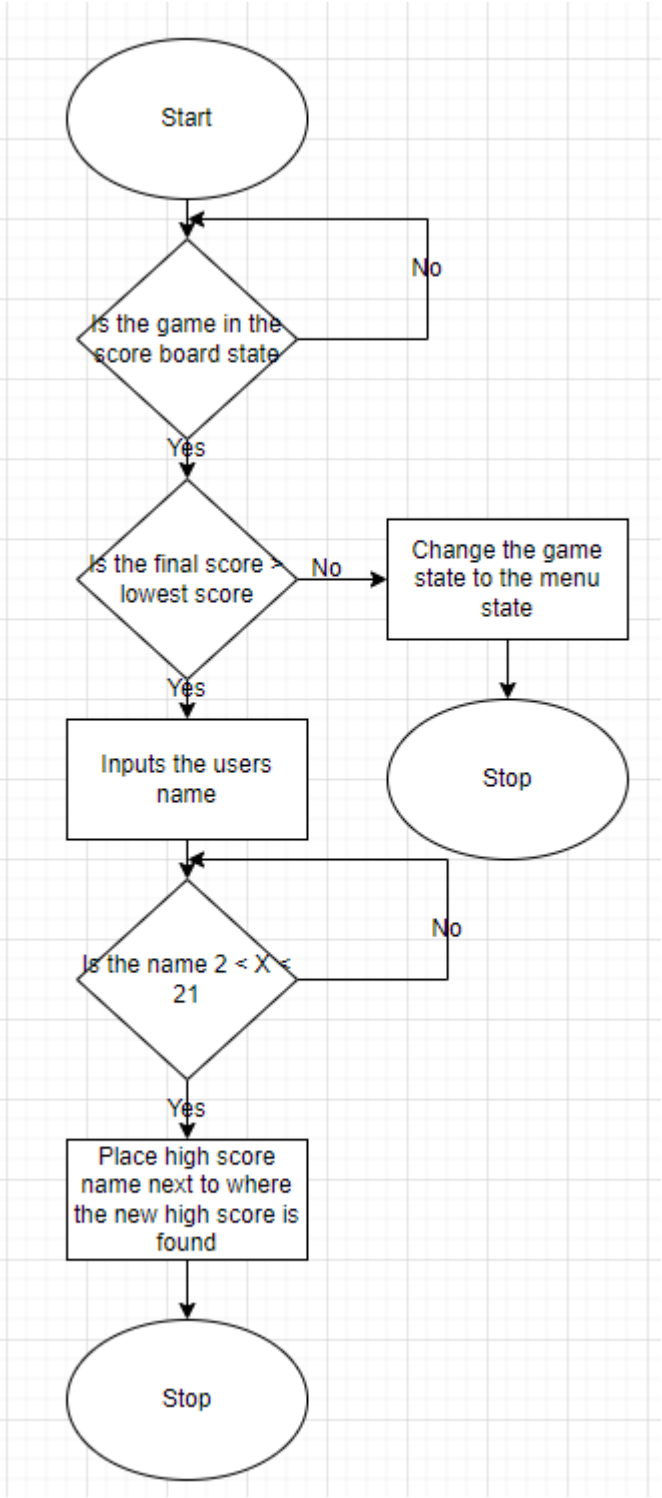
Variable	Bullets	Integer	A variable with the original value of 30	This gives the user a finite amount of lives making the game more challenging
Procedure	Bullet removal		1 bullet is removed from the bullet variable and one of the bullet sprites are removed	This makes the game more challenging as every time the user misses the target a bullet is removed
Procedure	Live removal		1 life is removed if a enemy comes into contact with the player sprite and a heart sprite is removed	This makes the game more challenging as the user comes into contact with enemies they will reduce their health and an intensity to it
Sound file	Shooting	.wav	Sound effect of a gunshot sound is played	This will help as an indicator if the player misses the visual of it
Sound file	Dying	.wav	Sound of a	This will help as an indicator for the player that they have died
Field sprite	Bullet	.jpeg	Pixel are of a bullet	This is an indicator for the user
Field sprite	Ammo	.jpeg	Pixel art of a box with ‘AMMO’ on it	This is an indicator for the user
Field sprite	Score	.jpeg	Pixel art of a box with ‘Score’ on it	This is an indicator for the user
Field sprite	Player standing	.jpeg	Pixel art of the player standing still	This makes the game unique and help as an indicator for the users
Field sprite	Player running left	.jpeg	Pixel art of the player running left	This makes the game unique and helps as an indicator for the users
Field sprite	Player running right	.jpeg	Pixel art of the player running right	This makes the game unique and helps as an indicator for the users
Field sprite	Player jumping	.jpeg	Pixel art of the playing jumping up	This makes the game unique and helps as an indicator for the users
Field sprite	Player shooting	.jpeg	Pixel art of the player shooting	This makes the game unique and

				helps as an indicator for the users
Field sprite	**OBJECT (Change after game made)**	.jpeg	Pixel art of **OBJECT** to stop the user from going through it making it an obstacle	This makes the game more challenging as the player will have to jump over the object while also making sure they don't die
Field sprite	**OBJECT (Change after game made)**	.jpeg	Pixel art of **OBJECT** to stop the user from going through it making it an obstacle	This makes the game more challenging as the player will have to jump over the object while also making sure they don't die
Field sprite	Enemy 1	.gif	Pixel art of an enemy to attack the player making the player have to defeat them	This makes the game more challenging as it makes the player more careful how they play so they don't die
Field sprite	Enemy 2	.gif	Pixel art of an enemy to attack the player making the player have to defeat them	This makes the game more challenging as it makes the player more careful how they play so they don't die
Field sprite	Enemy 3	.gif	Pixel art of an enemy to attack the player making the player have to defeat them	This makes the game more challenging as it makes the player more careful how they play so they don't die
Field sprite	Background	.jpeg	Pixel art of the background	This makes the game unique and adds a visual graphic to not make it look plain

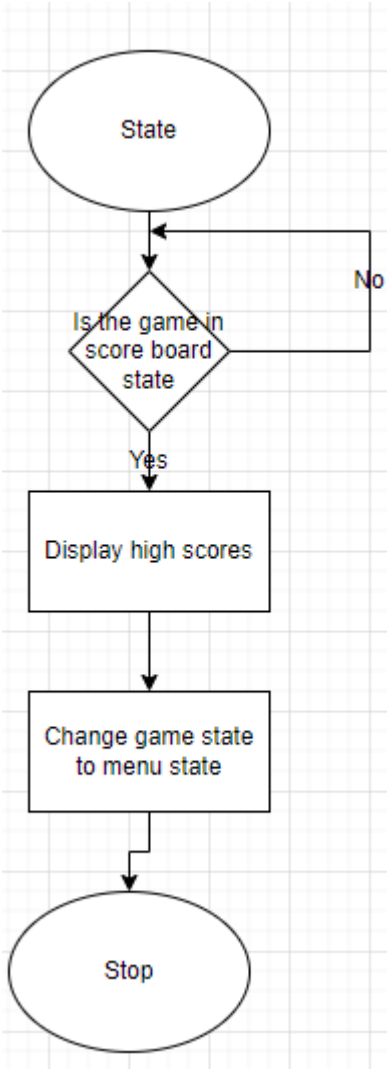
SCOREBOARD


Start
Is the game in the playing state
No
Yes
Is the final score > lowest high score
No
Change the game menu to the menu state
Stop
Yes
Set index to 0
Is the final score < index = 1 current index value
No
Increment current index value by 1
Yes
Set index to 0
Is the current index = index with the value below final score
No
Overwrite value in current index by current index + 1
Yes
Overwrite index with value below high score with final score
Stop

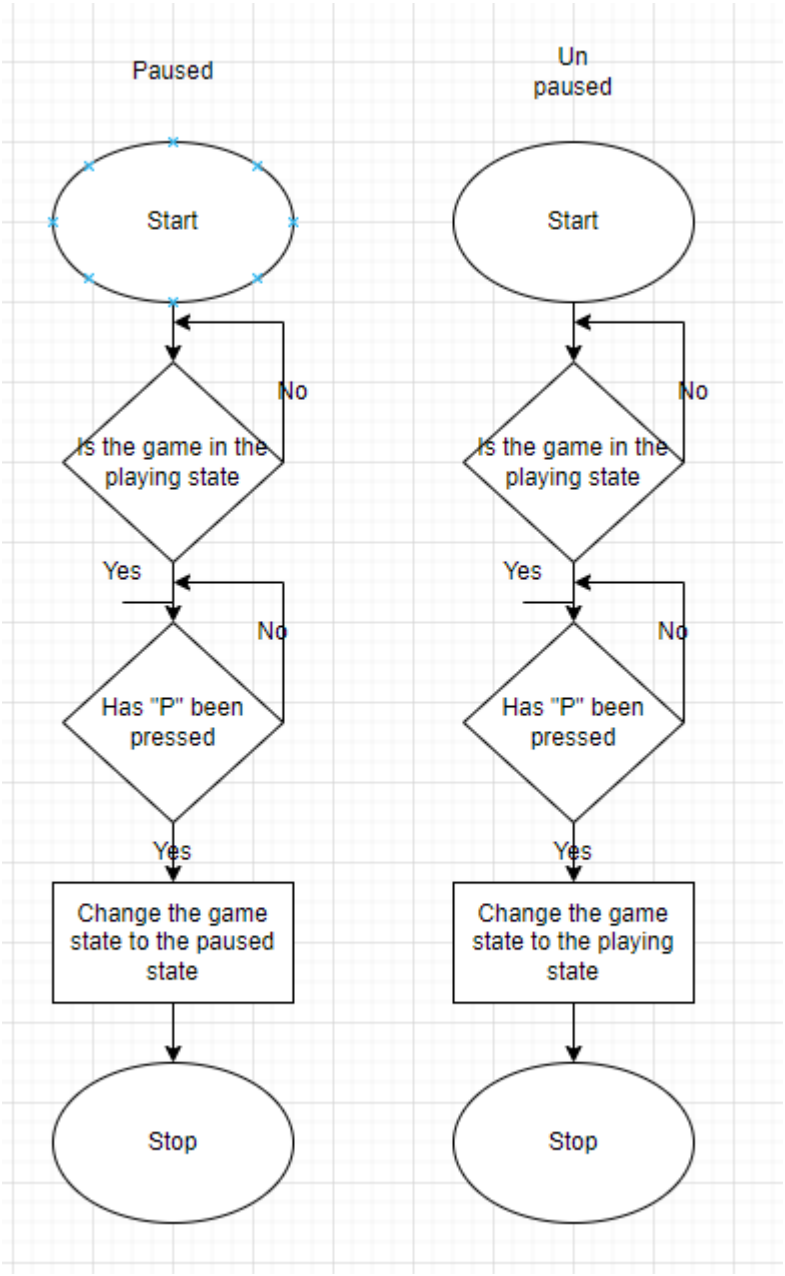
HIGH SCORE NAME AND VALIDATION



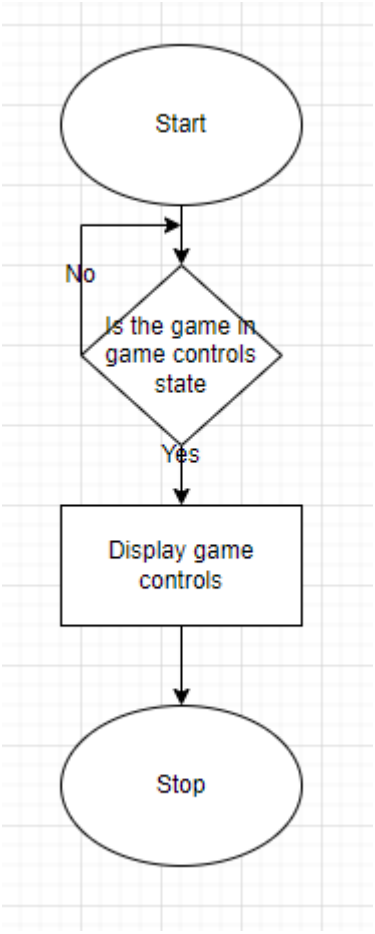
DISPLAY HIGH SCORE



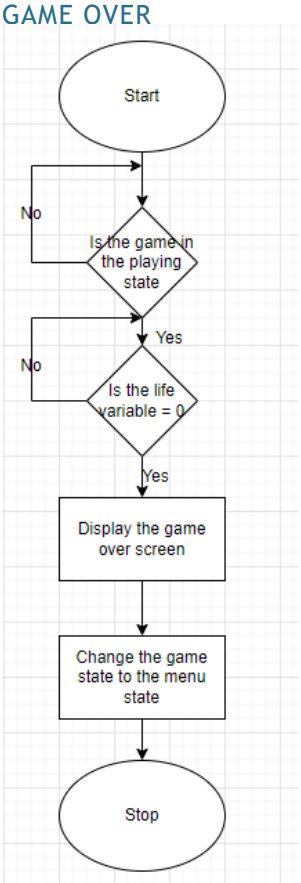
PAUSED AND UN-PAUSED



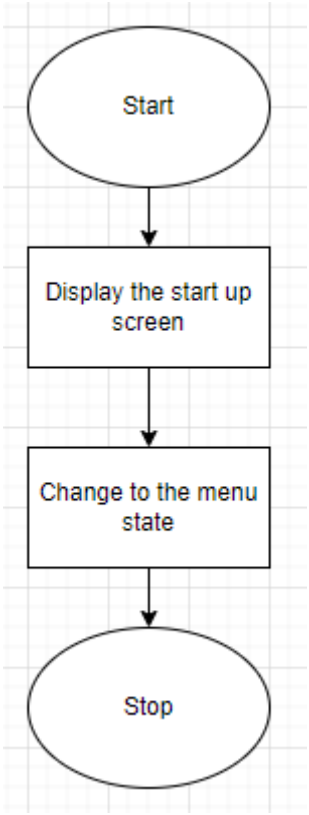
GAME CONTROLS



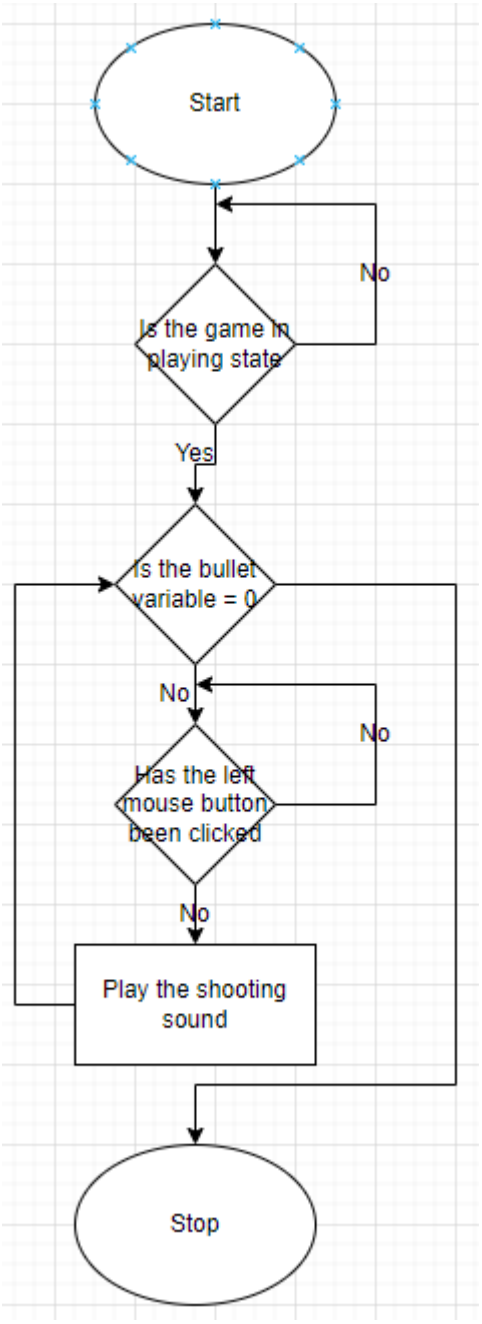
GAME OVER



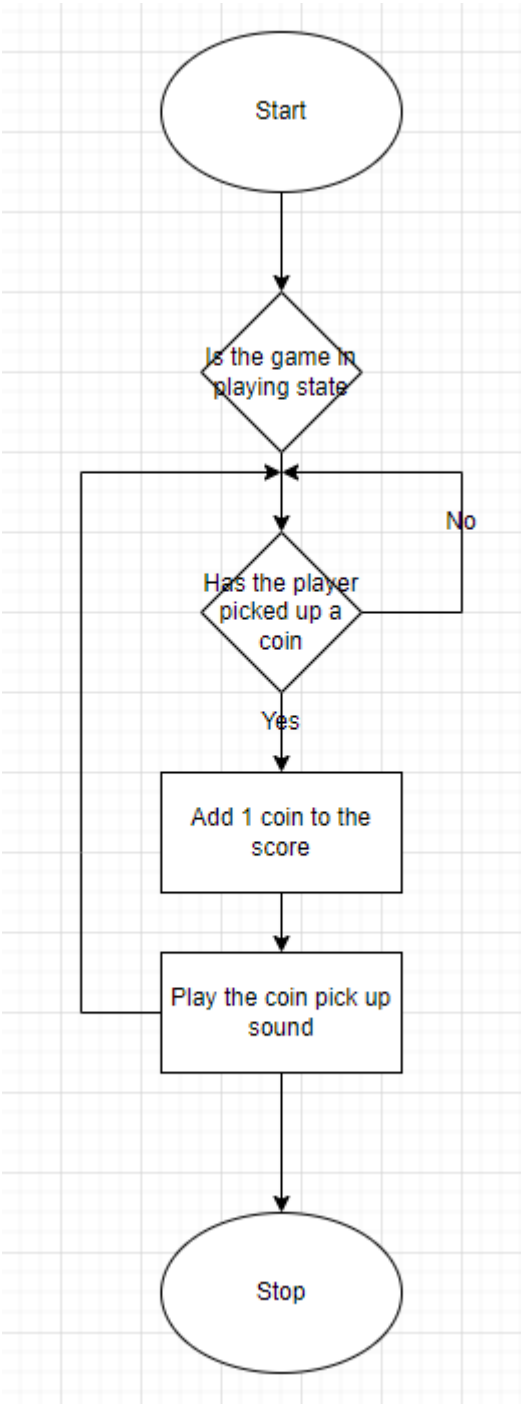
START UP


Start
Display the start up screen
Change to the menu state
Stop

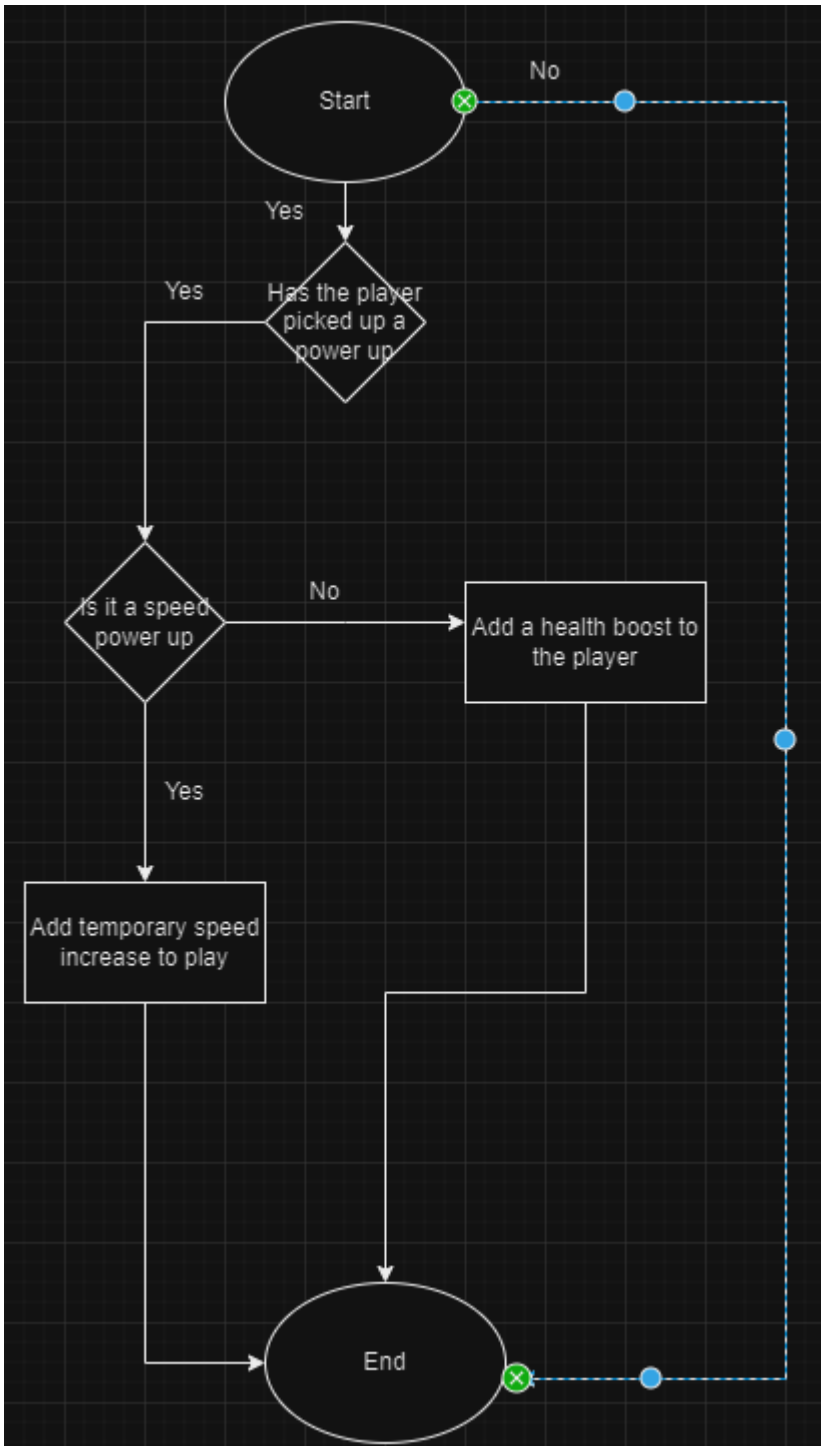
SHOOTING CHECK AND SOUND



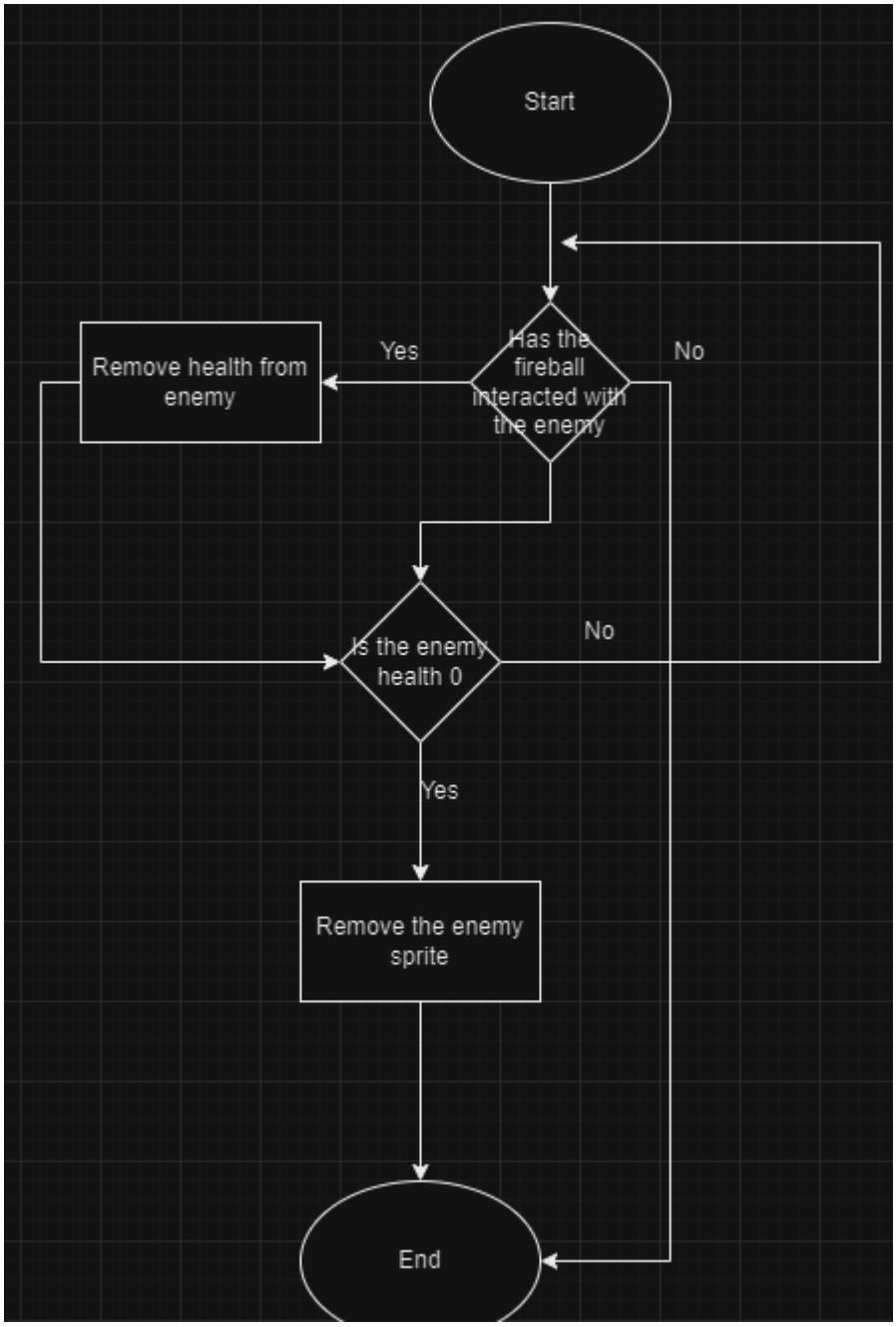
COIN PICK UP + SOUND



POWER UP PICKUP

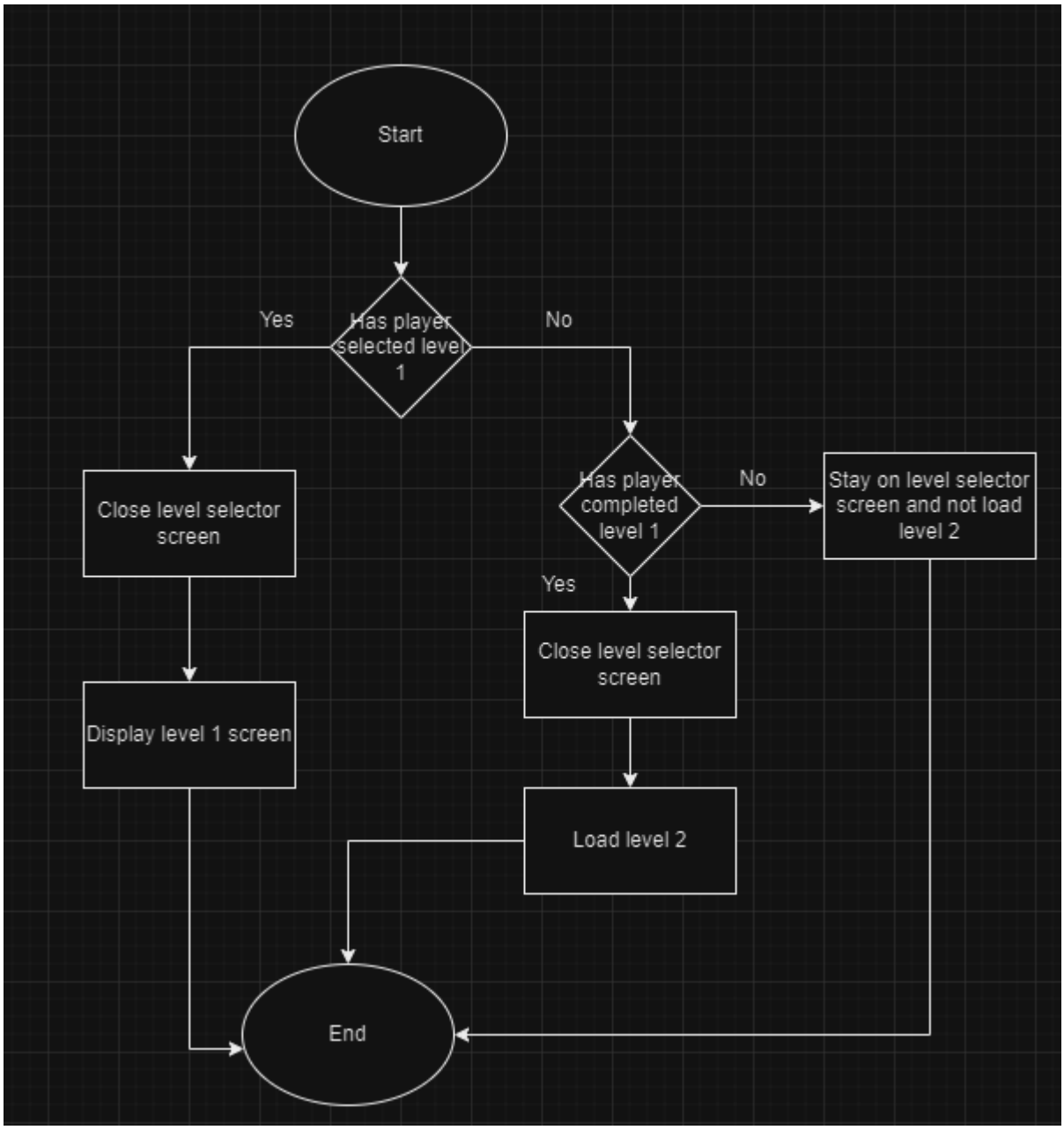


ENEMY HEALTH REMOVAL/DEATH



PLAYER DEATH

LEVEL SELECTOR



ALGORITHM JUSTIFICATION

SCOREBOARD

This is essential for the game because it needs to have a sense of difficulty for the player to try beat the highest score in the game

HIGH SCORE NAME AND VALIDATION

This is essential because it's a requirement for my stakeholder to not get bored of the game and they have a goal to keep beating making it less boring after they finish the game first try

DISPLAY HIGH SCORE

This is essential as I need a way to display the high score so the player can tell what the highest score is for the level

PAUSED AND UN PAUSED

This is essential for my game because the stakeholder made this a requirement to pause the game and unpause and also keep the game state the same like the score and timer

GAME CONTROLS

This is essential for the game as the player needs to be controlled by the user and to complete levels

GAME OVER

The justification for this is that the game needs to be constantly be checking if the player health is 0 or the player has finished the level and change the game screen

SHOOTING CHECK AND SOUND

The justification for this is that the game needs to be checking if the player has shot the fireball and to execute a procedure that corresponds with that control and also plays a sound.

COIN PICK UP + SOUND

The justification for this is that the game needs to be checking if the player has interacted with a coin and if they had it removes the coin and adds one onto the score and plays a sound

POWER UP PICKUP

The justification for this is that the game needs to check if the player has interacted with a power up and it needs to give the according power up and remove the powerup from the screen

ENEMY HEALTH REMOVAL/DEATH

The justification for this is that the game needs to check if the player has shot a fireball and if the fireball has interacted with the enemy it has to remove the enemies health and if its at 0 it needs to remove the enemy sprite

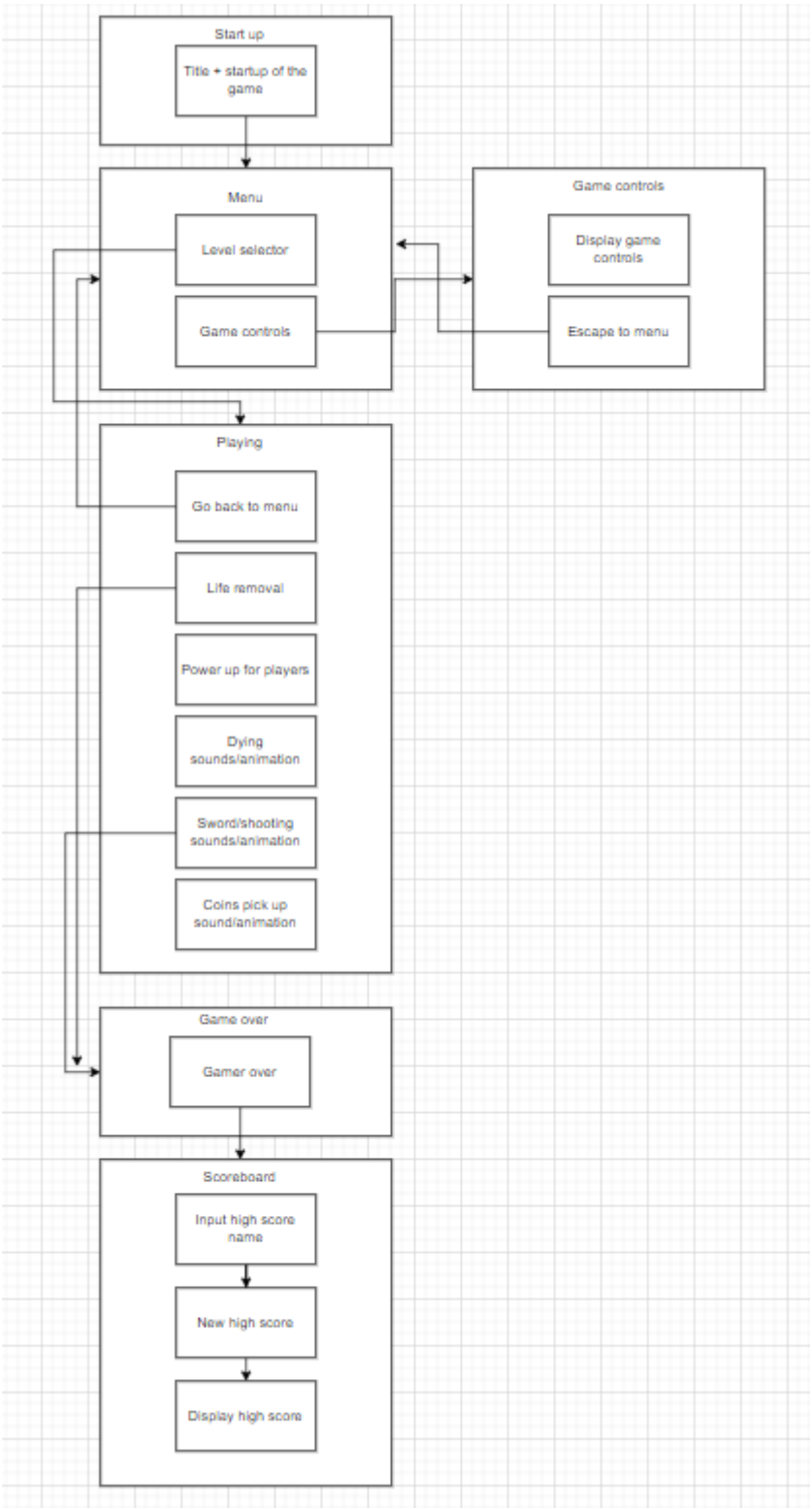
PLAYER DEATH

If the player reaches 0 health the player sprite has to be removed and the game state has to be changed to game over screen

LEVEL SELECTOR

This is essential as it’s a requirement from my stakeholder as he requested for multiple level and difficulties so a level selector is needed to meet this requirement

LINKING THE ALGORITHMS TOGETHER



Scoreboard

New high score

Test data	Type	Justification
New score > scores on scoreboard	Valid	The game needs to display the new score that is higher than the highscore
New score < scores on scoreboard	Invalid	The game doesn't display the new score that is higher than the highscore

High score name and validation

Test data	Type	Justification
Pixelgamer29	Valid	The game only allows 15 character names with letters and numbers
luwfuiwsoweit498347hic3ujnur	Invalid	The game doesn't allow for names bigger than 15 characters but the name is valid in terms of characters
#1	Invalid	The game doesn't allow for special characters and names that are too short

Controls

Test data	Type	Justification
'W'	Valid	The game needs to validate which characters are being pressed in the game and W is a valid character that has a piece of code that executes when pressed
'A'	Valid	The game needs to validate which characters are being pressed in the game and A is a valid character that has a piece of code that executes when pressed
'S'	Invalid	I need to make sure to check if the mechanic of movement is working but only for buttons I've linked it to
'D'	Valid	The game needs to validate which characters are being pressed in the game and D is a valid character that has a piece of code that executes when pressed
'G'	Invalid	I need to make sure to check if the mechanic of movement is working but only for buttons I've linked it to
'C'	Invalid	I need to make sure to check if the mechanic of movement

		is working but only for buttons I've linked it to
'Q'	Invalid	I need to make sure to check if the mechanic of movement is working but only for buttons I've linked it to

Sound effects

Test data	Type	Justification
'Spacebar'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player dies	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
'L'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player shoots	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player picks up coins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player wins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test

Life removal

Test data	Type	Justification
Player sprite comes into contact with ai sprite	Valid	When the sprite interact with the enemy the life of the player should be decreassed
Counter value is 0	Valid	When the timer reaches 0, the lives of the player should be removed and end the game
'Left mouse click'	Invalid	Clicking the left mouse should not execute the life removal procedure
'S'	Invalid	Pressing the letter S should not execute the life removal procedure

Bullet removal

Test data	Type	Justification

'Esc'	**Invalid**	**When the player presses escape the bullet amount should not be removed**
Counter value is 0	**Valid**	**When the counter value is 0 the bullets should be removed**
'Spacebar'	**Valid**	**When the spacebar button is pressed a bullet should be removed**

Pause

Test data	Type	Justification
'Esc'	Invalid	When the player presses escape the game shouldn't pause
'P'	Valid	When the player presses P it should execute the pause procedure as it meets the conditions of it
']'	Invalid	When the player presses] the game shouldn't pause because it isn't meeting the conditions of the procedure

Un-paused

Test data	Type	Justification
'Tab'	Invalid	When the player presses tab the un paused procedure should not execute
'P'	Valid	When the player presses P it should execute the pause procedure again as it meets the conditions
'A'	Invalid	When the player presses A it should not

Dying animation

Test data	Type	Justification
'Spacebar' on the enemy	**Valid**	**When the player presses the spacebar, it should be the only valid keyboard character that executes corresponding code**
'Delete' on the enemy	**Invalid**	**This is just a random key that should not work as the game should be checking what keys are pressed**
'1'	**Invalid**	**This is just a random key that should not work as the game should be checking what keys are pressed**

Gunshot

Test data	Type	Justification
'Left mouse click'	**Invalid**	**This is just a random key that should not work as**

		the game should be checking what keys are pressed
'Spacebar'	**Valid**	**To enable the audio sound, the game should only play it when the spacebar character is pressed**
Coming into contact with objects	**Invalid**	**This is just a random key that should not work as the game should be checking what keys are pressed**

Dying animation

Test data	Type	Justification
Counter value is 2	**Invalid**	**The play health value should not be executing code that is meant for the dying animation if the value is greater than 0**
Counter value is -2	**Invalid**	**The player dying animation should not be executed as it's a erroneous test that doesn't allow for negative health value for players**
Counter value is 0	**Valid**	**This is a valid test that should play the dying animation as its counter value is equal to the condition of the code**

Score value

Test data	Type	Justification
Player comes into contact with coin	**Valid**	**This is a valid test as the player interacts with the coin it should add onto the score value**
Timer is > 0	**Valid**	**This is a valid test because the timer impacts the score and if the timer value is less than 0 the score is invalid**
'Enter'	**Invalid**	**This is a invalid test because the character or condition of the score value being added to hasn't been met**

Power-ups

Test data	Type	Justification
Player comes into contact with enemy	**Invalid**	**This is a invalid test as the condition of a power up is different to interacting with a enemy so it should not give a power up to the player**
Player comes into contact with coins	**Invalid**	**This is a invalid test as the condition of a power up is different to interacting**

		with a coin so it should not give a power up to the player
Player comes into contact with power-up	**Valid**	**This is a valid test as the conditions have been met for the corresponding code to execute to add a power up**

GIF changing

Test data	Type	Justification
Player presses 'A'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player presses 'W'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player presses 'S'	**Invalid**	**This is a invalid test as it's a random character that should not be executing any code**
Player presses 'P'	**Invalid**	**This is the pause state button so it should be changing game states and not the gifs of the character**
Player presses 'Spacebar'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player comes into contact with enemy	**Invalid**	**This is a invalid test because I haven't added any animations for the character to change gifs while interacting with it**
Life counter value = 0	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Bullet counter value = 0	**Invalid**	**This is a invalid test as its another piece of code that runs in another time separate from the gif changing**

POST DEVELOPMENT TESTING

Test data	Type	Justification
'Spacebar'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player dies	Valid	My sound effects need to play when conditions are met in the game and this

		test is a example of it being a valid test
'L'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player shoots	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player picks up coins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player wins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test

Test data	Type	Justification
Player sprite comes into contact with ai sprite	Valid	When the sprite interact with the enemy the life of the player should be decreassed
Counter value is 0	Valid	When the timer reaches 0, the lives of the player should be removed and end the game
'Left mouse click'	Invalid	Clicking the left mouse should not execute the life removal procedure
'S'	Invalid	Pressing the letter S should not execute the life removal procedure

Test data	Type	Justification
'Esc'	**Invalid**	**When the player presses escape the bullet amount should not be removed**
Counter value is 0	**Valid**	**When the counter value is 0 the bullets should be removed**
'Spacebar'	**Valid**	**When the spacebar button is pressed a bullet should be removed**

Test data	Type	Justification
'Left mouse click'	**Invalid**	**This is just a random key that should not work as the game should be**

		checking what keys are pressed
‘Spacebar’	Valid	To enable the audio sound, the game should only play it when the spacebar character is pressed
Coming into contact with objects	Invalid	This is just a random key that should not work as the game should be checking what keys are pressed

Test data	Type	Justification
Counter value is 2	Invalid	The play health value should not be executing code that is meant for the dying animation if the value is greater than 0
Counter value is -2	Invalid	The player dying animation should not be executed as it’s a erroneous test that doesn’t allow for negative health value for players
Counter value is 0	Valid	This is a valid test that should play the dying animation as its counter value is equal to the condition of the code

Test data	Type	Justification
Player comes into contact with coin	Valid	This is a valid test as the player interacts with the coin it should add onto the score value
Timer is > 0	Valid	This is a valid test because the timer impacts the score and if the timer value is less than 0 the score is invalid
‘Enter’	Invalid	This is a invalid test because the character or condition of the score value being added to hasn’t been met

Test data	Type	Justification
Player comes into contact with enemy	Invalid	This is a invalid test as the condition of a power up is different to interacting with a enemy so it should not give a power up to the player
Player comes into contact with coins	Invalid	This is a invalid test as the condition of a power

		up is different to interacting with a coin so it should not give a power up to the player
Player comes into contact with power-up	**Valid**	**This is a valid test as the conditions have been met for the corresponding code to execute to add a power up**

Test data	Type	Justification
Player presses 'A'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player presses 'W'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player presses 'D'	**Invalid**	**This is a invalid test as it's a random character that should not be executing any code**
Player presses 'P'	**Invalid**	**This is the pause state button so it should be changing game states and not the gifs of the character**
Player presses 'Spacebar'	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Player comes into contact with enemy	**Invalid**	**This is a invalid test because I haven't added any animations for the character to change gifs while interacting with it**
Life counter value = 0	**Valid**	**This is a valid test as it's a character for a condition that changes gif for the character**
Bullet counter value = 0	**Invalid**	**This is a invalid test as its another piece of code that runs in another time separate from the gif changing**

Test data	Type	Justification
'Esc'	**Invalid**	**When the player presses escape the bullet amount should not be removed**
Counter value is 0	**Valid**	**When the counter value is 0 the bullets should be removed**
'Spacebar'	**Valid**	**When the spacebar button is pressed a bullet should be removed**

Test data	Type	Justification
'Spacebar'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player dies	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
'L'	Invalid	I need to make sure that the sound effects play when a certain condition is met and not when pressing a character
Player shoots	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player picks up coins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test
Player wins	Valid	My sound effects need to play when conditions are met in the game and this test is a example of it being a valid test

ACCEPTANCE TESTING

Number	Requirements	Input	Expected output	Justification
1	The users final score is added to the scoreboard if the score is great than the scoreboards lowest score	**Valid:** A final score of 300 **Invalid:** The string'Pixelgamer49'	**Valid:** The final score of the player will be placed on the leaderboard accordingly next to the users name **Invalid:** The string 'Pixelgamer49' is displayed as the score and the name on the leaderboard	**The justification for this is that the game should not display the username as the score so its incorrect if it does**

2	The users name can be used for the leaderboard if the name is > 2 and < 20 characters	**Valid:** The string 'Pixelgamer49' **Invalid:** The string 'A'	**Valid:** The string 'Pixelgamer49' is used on the leaderboard and placed next to the players final score **Invalid:** The string 'A' is placed on the leaderboard and placed next to the players final score3	**The justification for this is that a username cannot be less that 2 characters or longer than 20 so the game looks more compact and immersive for the player if its realistic names**
3	The player gains 1 point when a coin is picked up	**Valid:** When the player sprite comes into contact with a coin sprite, a point is added to the score **Invalid:** When the player sprite comes into contact with a coin sprite, a point isn't added	**Valid:** 1 point is added to the players score **Invalid:** No points is added to the players score	**The justification for this is that when the coin is picked up a point should be added to the score and this test checks if it works as it's a requirement from the stakeholder**
4	Score box at the top left of the screen	**N/A**	The scorebox is in the top left of the screen	**The justification for this is that the stakeholder made this a requirement for the player as it needs a interactable health bar to keep track of how much health they have**
5	Ammo box at the bottom left	**N/A**	The scorebox is in the bottom left of the screen	**The justification for this is that the player should be able to see how much ammo they have to make it easier so they don't have to guess how much they have**
6	100 health	**Valid:** The life variable will be set to 100 **Invalid:** The life variable will be set to another number than 100	**Valid:** When the level is selected, the life variable will be reset to 100 **Invalid:** When the level is selected, the life variable will be reset	**The justification for this is that the player requested for 100 health in the design success criteria and requirements so I adding this meets their requirements**

			to a number other than 100	
7	All sprites and backgrounds will be in a pixel art style	**N/A**	All sprites and backgrounds will be in a pixel art style	**The justification for this is that a cartoon design theme is a requirement from the success criteria as the stakeholder specifically requested this**
8	Pressing the 'P' key to pause the game	**Valid:** When in the playing state of the game, the character 'P' is pressed **Invalid:** When in the playing state of the game, the character 'W' is pressed	**Valid:** The game state will change from the playing state from to paused state **Invalid:** The player's character will jump	**The justification for this is that the stakeholder requested that I create a pause function for the game**
9	Pressing the 'P' key to un-pause the game	**Valid:** When in the paused state of the game, the character 'P' is pressed **Invalid:** When in the playing state of the game, the character 'W' is pressed	**Valid:** The game state will change from paused to the playing state `+state **Invalid:** The player's character will jump	**The justification for this is that the player requested that I created a pause function for the game that works**
10	Dying animation for the player	**Valid:** Life counter for the player reaches 0 **Invalid:** Life counter for the player is > 0	**Valid:** The dying gif will play when the player life reaches 0 **Invalid:** The dying gif will not play	**The justification for this is that the player should have a dying animation when their health reaches 0 to indicate that they have died**
11	Player picks up power up	**Valid:** When the player sprite comes into contact with a power up it is removed **Invalid:** When the player sprite comes into contact with a power up it isn't removed	**Valid:** Player will receive a power up according to its sprite picture e.g. more speed and the power up sprite is removed **Invalid:** Player will not receive a power up according to its sprite picture and the power up	**The justification for this is that powerups are a requirement from the stakeholder and testing this and making it valid meets the stakeholders request**

			sprite is not removed	
12	GIF changes according to the players controls and movements	**Valid:** When in the playing state the player presses a control for example 'W', 'A', 'D', 'Spacebar' **Invalid:** When in the playing state the player presses a control for examples 'O', 'Y', 'T', 'Delete'	**Valid:** The gif of the character changes according to the control that is pressed **Invalid:** The gif of the character does not change according to the control that is pressed	**The justification for this is that GIF changes makes the game more immersive for the player and achieves the stakeholders requirements**
13	Ammo box is 3	**Valid:** The ammo box variable is 3 as default **Invalid:** The ammo box variable is another number than 3 as default	**Valid:** When the level is selected, the ammo box variable is reset to 3 **Invalid:** When the level is selected, The ammo box variable is reset to a number that's not 3	**The justification for this is that I am meeting the stakeholders requirements of having 3 ammo when the game starts**
14	Life removal	**Valid:** Player sprite comes into contact with an enemy sprite an animation will play **Invalid:** Player sprite comes into contact with an enemy sprite an animation will not play	**Valid:** 1 life is removed when player sprite comes into contact with enemy sprite **Invalid:** No lives or more than 1 life is removed when the player sprite comes into contact with enemy sprite	**The justification for this is that the game needs to be removing health when the player comes into contact with a enemy sprite**
15	Fireball removal	**Valid:** Player presses spacebar as an attack **Invalid:** Player presses P as an attack	**Valid:** Fireball animation is played and a bullet is removed from the ammo box **Invalid:** No bullets are removed and the game state goes from playing state to paused state	**The justification for this is that the fireballs need to be remove when the player presses the spacebar button and not any other button**

16	Sound effects	**Valid:** Player sprite comes into contact with a coin sprite **Invalid:** Player sprite comes into contact with a coin sprite	**Valid:** A coin pickup sound will play **Invalid:** No sound effects will be played	**The justification for this is that there needs to be sound effects when the player interacts with something like a coin or presses spacebar**

PROPOSPAL SIGN OFF

Score board summary

What is successful	Improvements to be made
• Custom tag next to the scoreboard • Scoreboard background changes color depending on the score	• Needs to be more aesthetically pleasing

End page screen summary

What is successful	Improvements to be made
Win page • Shows players score • Allows you to select the next level	• Needs to be aesthetically pleasing
Dying page • Aesthetically pleasing • Shows players score	N/A

Main menu summary

What is successful	Improvements to be made
• Simple and easy menu navigation • Shows the player what they can do	• Needs to be an exit to desktop button

Instruction screen proposal

What is successful	Improvements to be made
• Easy to understand and follow	• Need to design it better

Initializing game screen summary

What is successful	Improvements to be made
• Design provides clears instructions	• Make a variety of background

Pause screen summary

What is successful	Improvements to be made
• Layout of the pause menu • Allows the player to see controls and change audio	N/A

SIGNED ……

C. DEVELOPING THE CODED SOLUTION ("THE DEVELOPMENT STORY")

DEVELOPMENT PLAN

This will be a section that has a brief breakdown on what I will be adding and in what order to create sprints that allow me to make the game efficiently within the time I have got.

ITERATION 1

What aspects of the program will be developed?

This will have the designing of the form and the user interface for the player to try and meet the requirements of the design section from the stakeholder.

Which tests will be required to test these parts of the game?

The tests that will be required is testing what inputs the player can give to the game and if the game crashes when a certain input is given.

What functionality will the expected prototype have?

At the end of the iteration I should be able to ask the stakeholder for feedback that they can evaluate and if the user interface is up to their standards and the design of the forms.

ITERATION 2

What aspects of the program will be developed?

This will be the character movement for my game and I will be trying to meet the stakeholder requirements from the users interview

Which tests will be required to test these parts of the game?

The tests that will be required are the boundary valid and invalid test to check if the game breaks if a boundary input is given, a valid input is given and a invalid input is given

What functionality will the expected prototype have?

The functionality should have:

- **Forwards**
- **Backwards**
- **Jump**

ITERATION 3

What aspects of the program will be developed?

This iteration will have:

- **Gem pickup**
- **Powerup pickup**

Which tests will be required to test these parts of the game?

If the player can interact with the pickups and whether they will execute the right code when the conditions are met

What functionality will the expected prototype have?

It should have a working pickup system of:

- **Gem pickup - Should add onto the score**
- **Powerup pickup - Should change the variables of the player**

ITERATION 4

What aspects of the program will be developed?

This iteration will contain the shooting aspect of the game that has been put in the requirement section of development

Which tests will be required to test these parts of the game?

The tests required should:

- **Should shoot a fireball**
- **Take away 1 ammo from ammo variable**

What functionality will the expected prototype have?

It should be able to shoot a fireball at a vector and it should take away 1 from the ammo variable and also the textbox on the user interface.

ITERATION 5

What aspects of the program will be developed?

This iteration will be the enemy AI and it should:

- **Be a moving AI**
- **Take damage away from the player**
- **Should be able to die**

Which tests will be required to test these parts of the game?

The tests required will be:

- **Does enemy AI die**
- **Does enemy AI do damage**
- **Does enemy AI move**

What functionality will the expected prototype have?

The enemy AI should have a health variable that is taken away from when the fireball interacts with the enemy and it should be able to take health away from the player.

ITERATION 6

What aspects of the program will be developed?

This iteration will be the scrolling background to give more immersion to the player as said in the design section

Which tests will be required to test these parts of the game?

- **Character movement**
- **Background movement**

What functionality will the expected prototype have?

- **Background moves with the player**

ITERATION 7

What aspects of the program will be developed?

- **Interrupt screens**

Which tests will be required to test these parts of the game?

- **Player dies**
- **Timer runs out**
- **Character P is pressed**
- **Player interacts with finishing flag**

What functionality will the expected prototype have?

- **Pause screen**
- **Death screen**
- **Finish screen**

ITERATION 8

What aspects of the program will be developed?

- **Level 2 of the game**
- **Moving floors**
- **Advanced enemy AI**

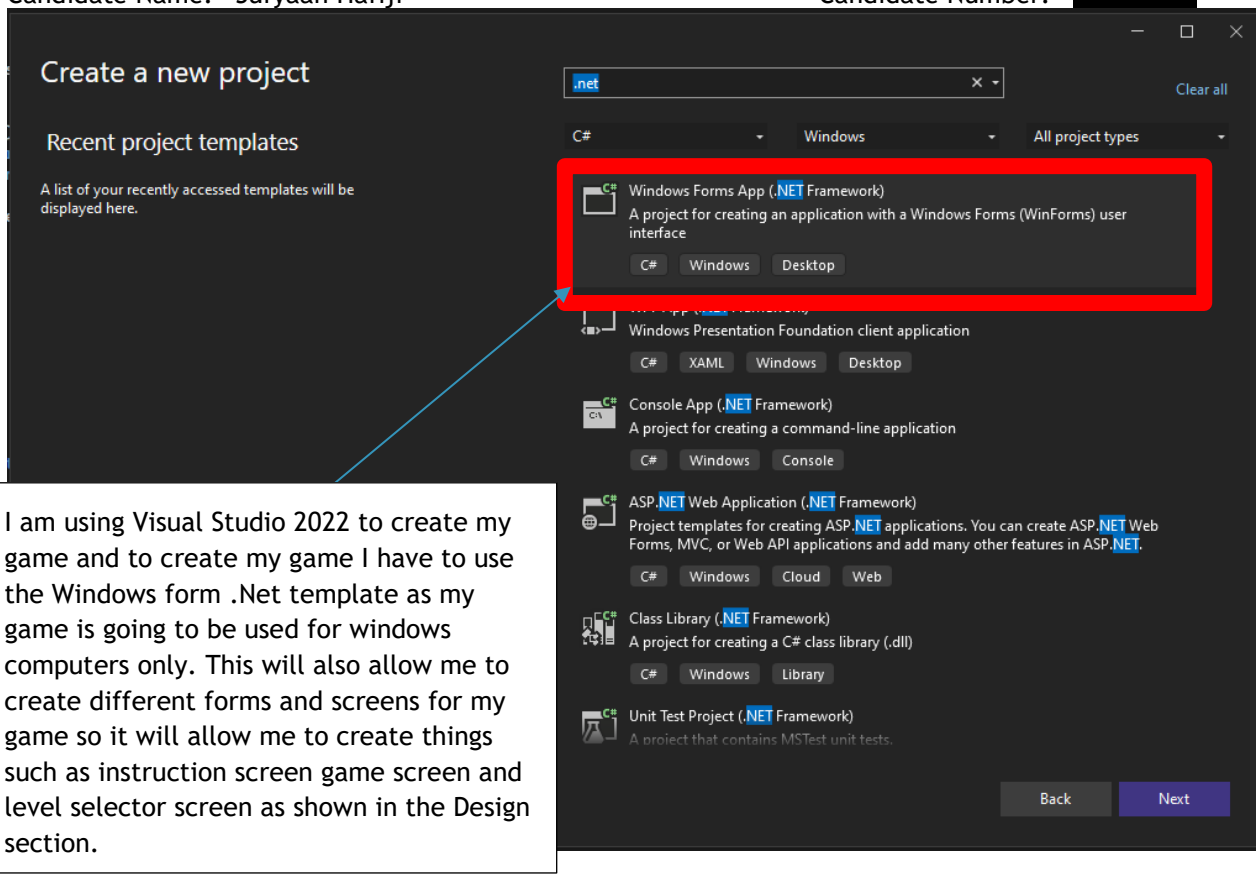
Which tests will be required to test these parts of the game?

- **Jumping on platform**
- **Damage from enemy AI**

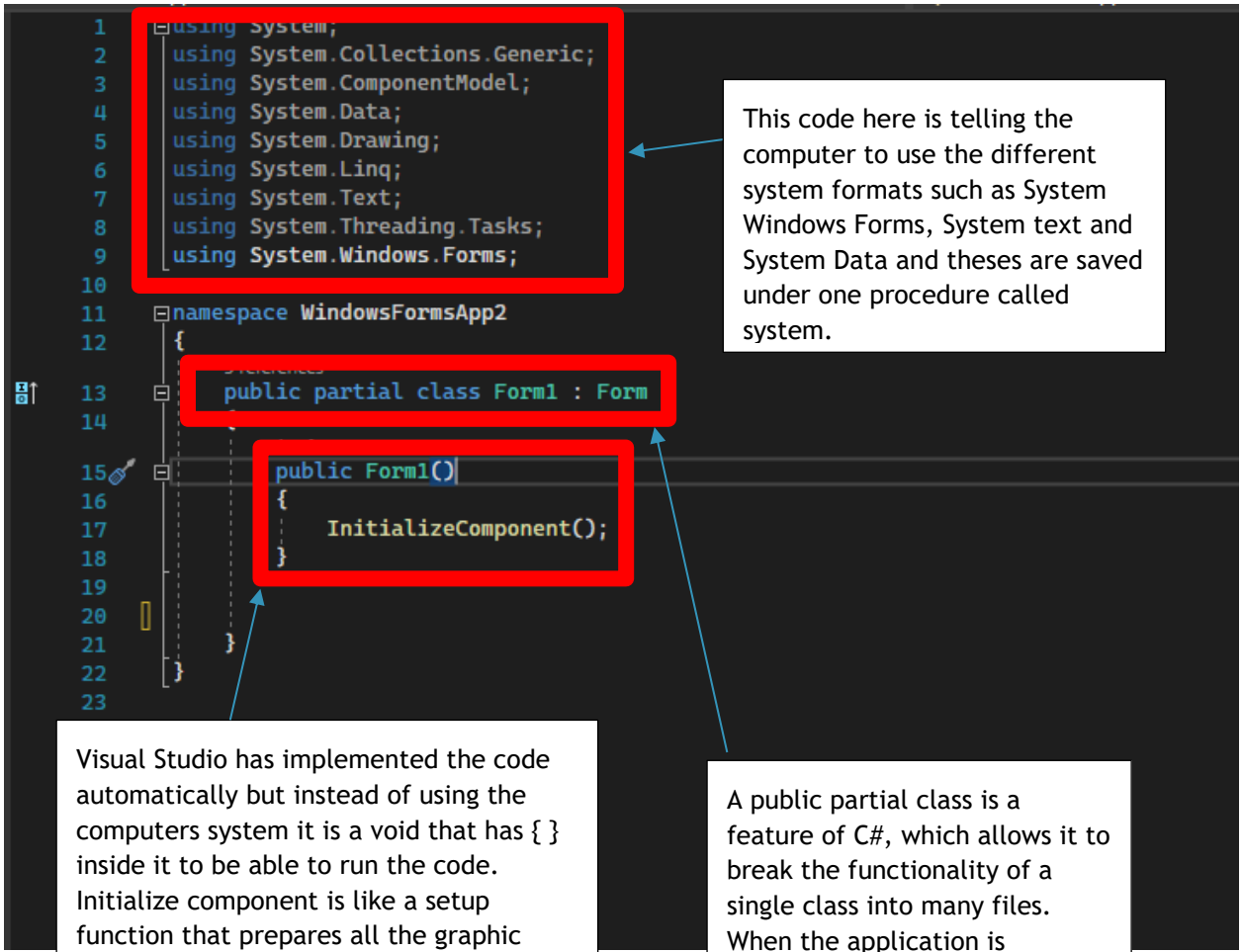
What functionality will the expected prototype have?

- **If player can stay on a moving platform**
- **If enemy AI work properly**
- **Same as previous iterations**

PROJECT CREATION



I am using Visual Studio 2022 to create my game and to create my game I have to use the Windows form .Net template as my game is going to be used for windows computers only. This will also allow me to create different forms and screens for my game so it will allow me to create things such as instruction screen game screen and level selector screen as shown in the Design section.

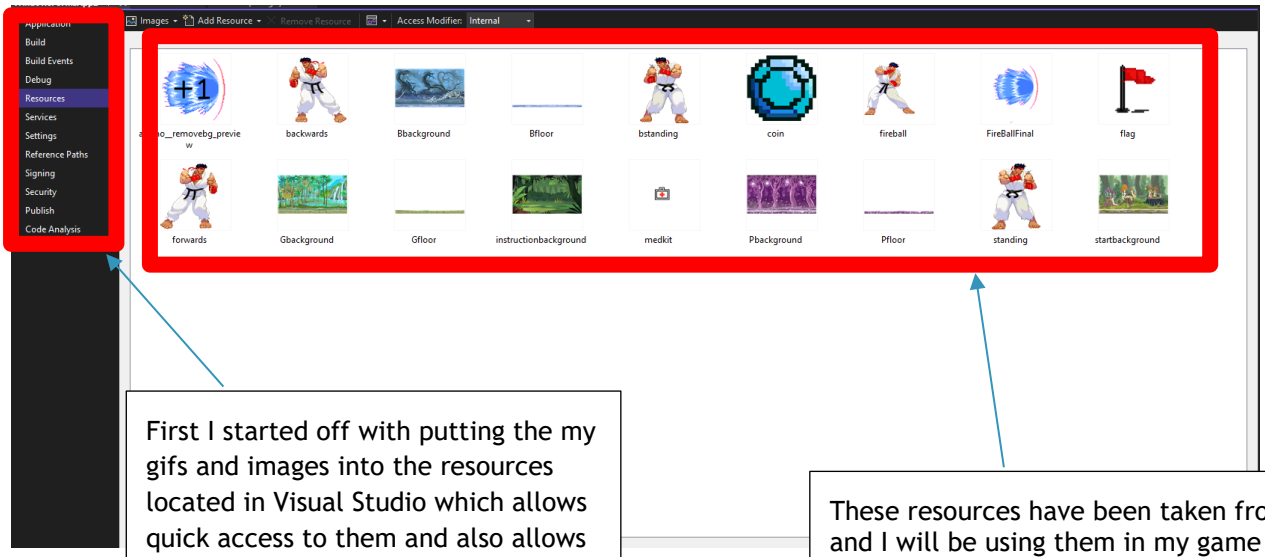


This code here is telling the computer to use the different system formats such as System Windows Forms, System text and System Data and theses are saved under one procedure called system.

Visual Studio has implemented the code automatically but instead of using the computers system it is a void that has { } inside it to be able to run the code. Initialize component is like a setup function that prepares all the graphic elements such as the background, picture boxes etc. and it is all based on the design I have created visually using the Visual Studio designer.

A public partial class is a feature of C#, which allows it to break the functionality of a single class into many files. When the application is compiled, these files are then reassembled into a single class file.

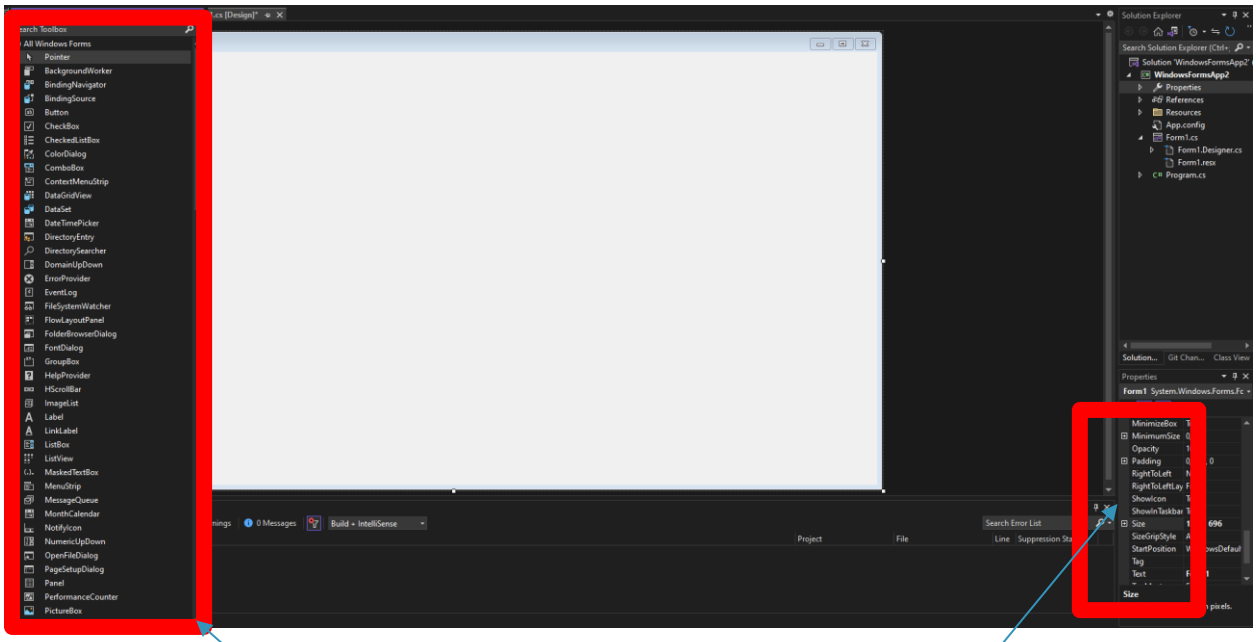
Candidate Name: <Sufyaan Hafiji> Candidate Number: <[redacted]>



First I started off with putting the my gifs and images into the resources located in Visual Studio which allows quick access to them and also allows me to change images or gifs in the code while the game is running

These resources have been taken from online and I will be using them in my game to save time instead of making my own. My player gifs have been taken from Moo ICT's street fighter game and the backgrounds have been taken from https://craftpix.net/freebies/free-fairy-tale-game-backgrounds/. The flag, medkit and coin have been taken from google images.

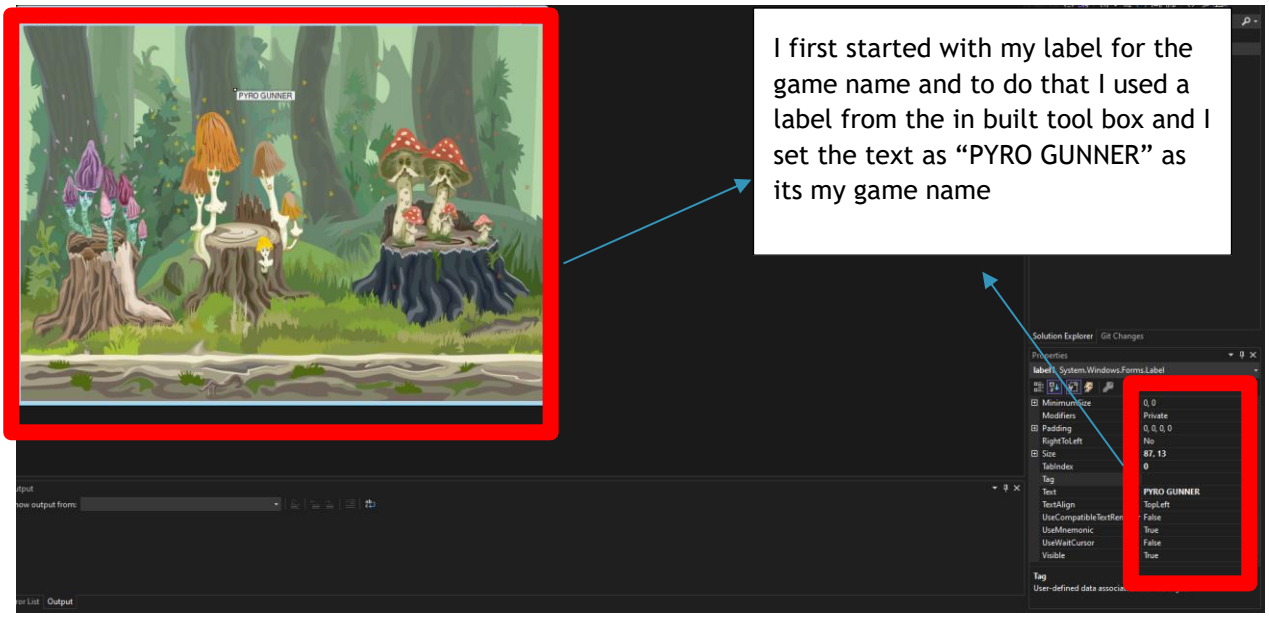
PROTOTYPE 1 (DESIGNING FORMS AND UI)

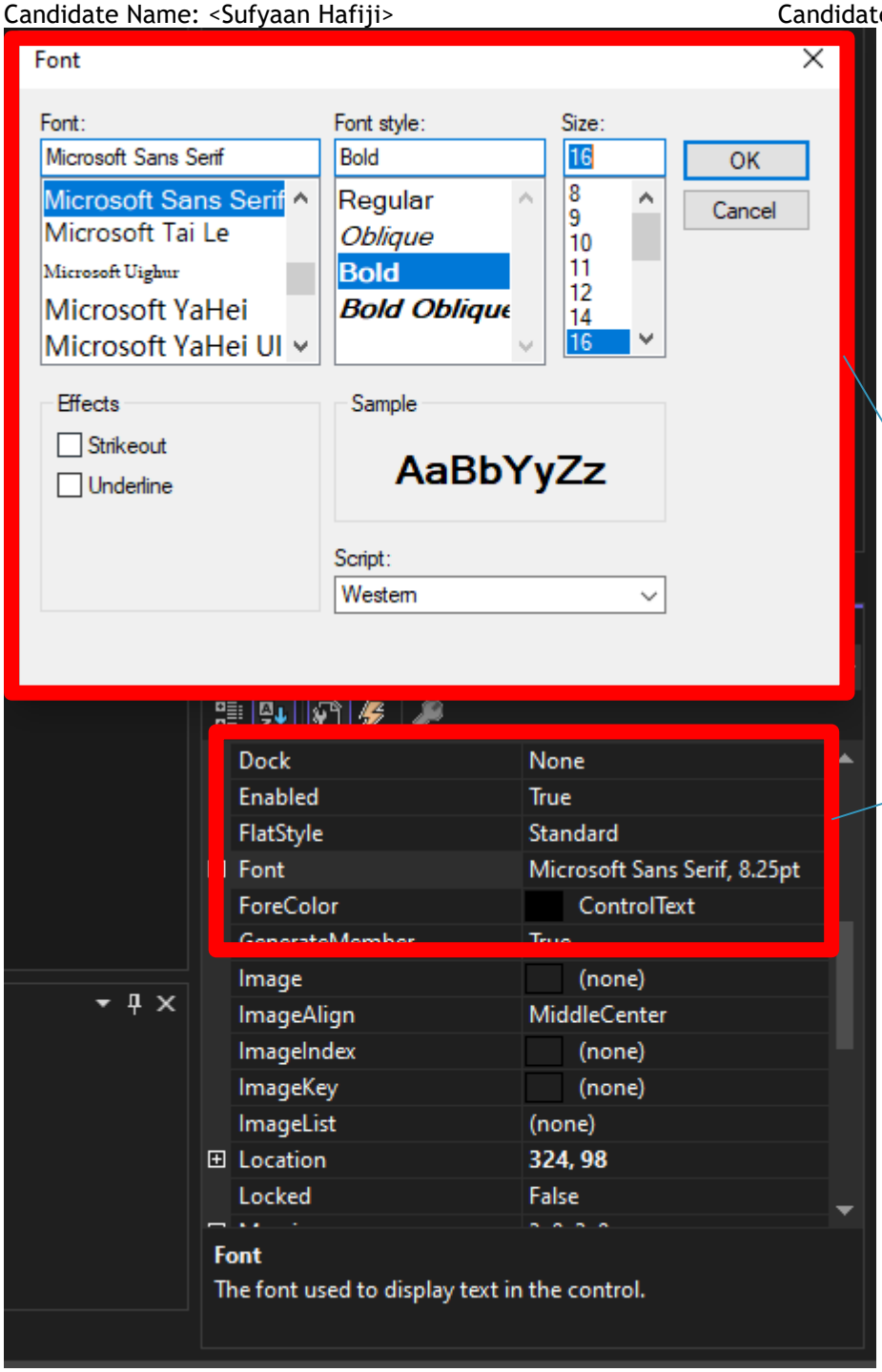


This toolbox comes installed with Visual Studios .net framework, which allows you to add graphical elements to your game and also things like timers which are the tick intervals/ frames your game runs at. I will also be using pictureboxes for the characters and platform and labels/buttons

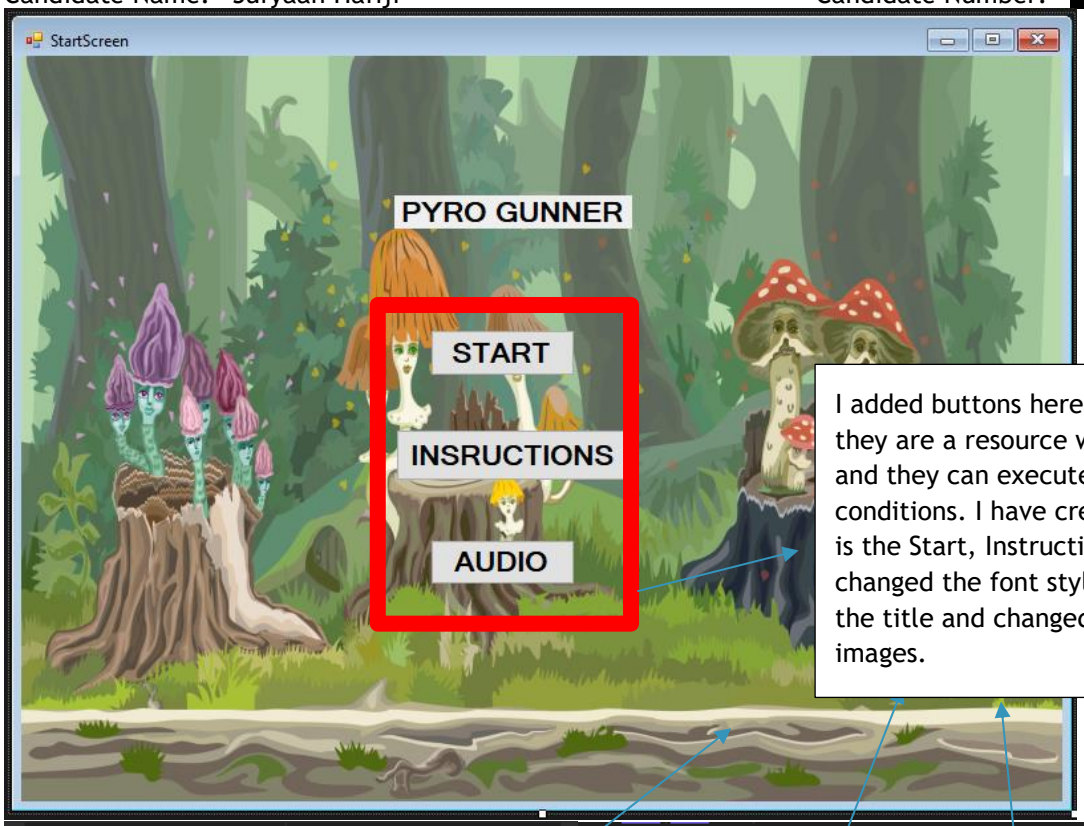
My form size will be set to 800 by 600 as my stakeholder asked for it to be simple and not too complicated for the first level and the menus. I will be changing this size when I do my 2nd level screen

After doing this, I decided to use my first design picture as a influence to my design as I wanted to keep it similar to what I first thought of. I did decide to change the background as it was a simple sketch of what I wanted the layout to be and not the design

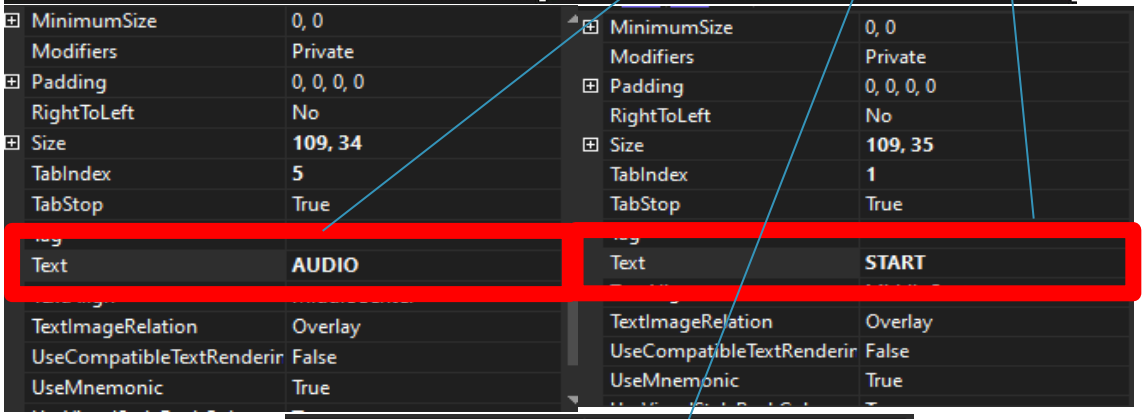


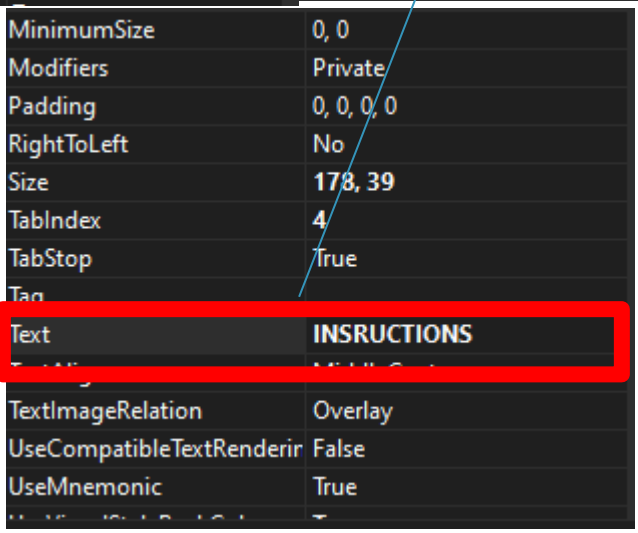


I then changed the font style to bold, as I want it to appeal to the player and changed the size of it to a bigger font. I have chosen Microsoft Sans Serif as my font for my start screen as I want a variety of fonts on my game

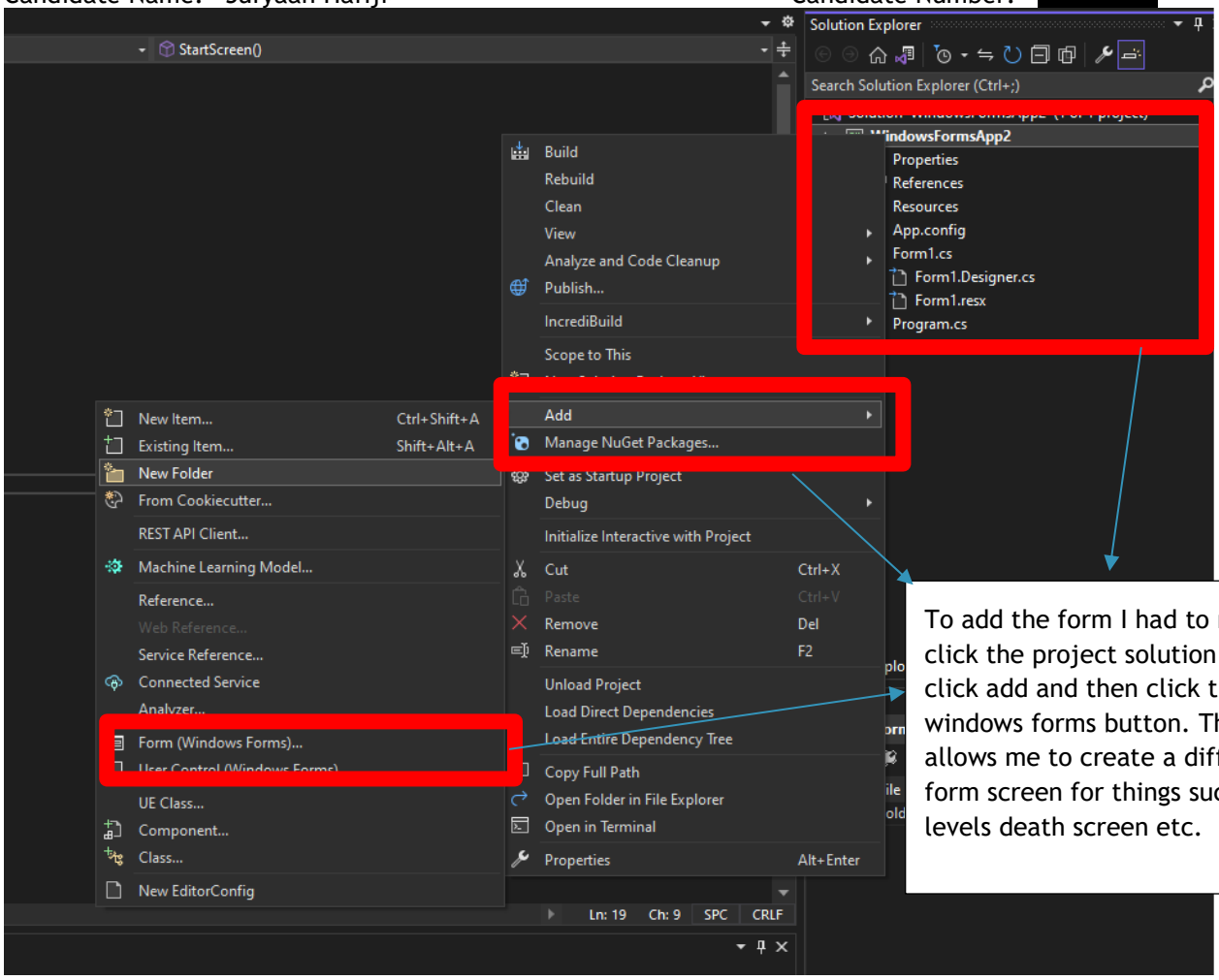


I added buttons here from the toolbox and they are a resource where they are intractable and they can execute code under certain conditions. I have created three buttons and it is the Start, Instructions and Audio buttons. I changed the font style and size of it to match the title and changed the text as shown in the images.

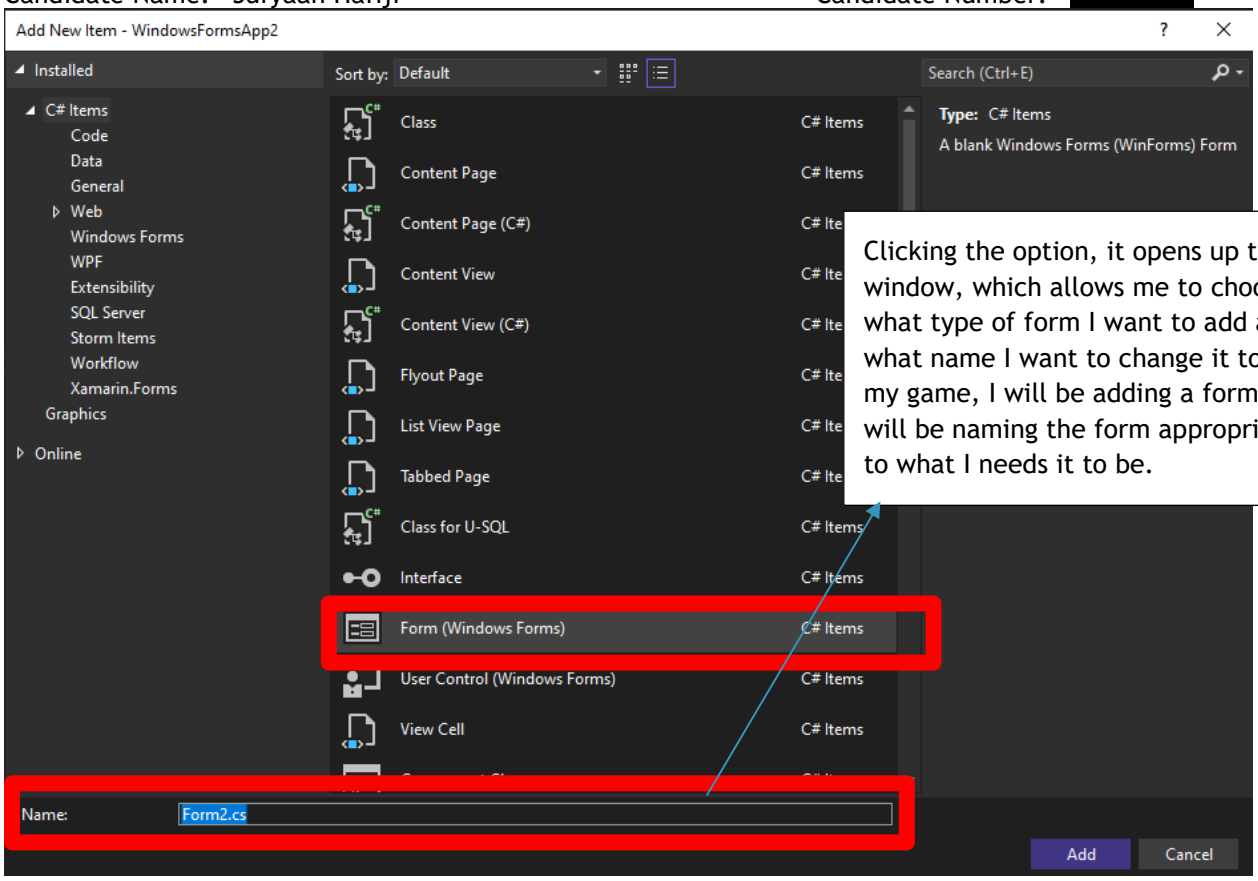




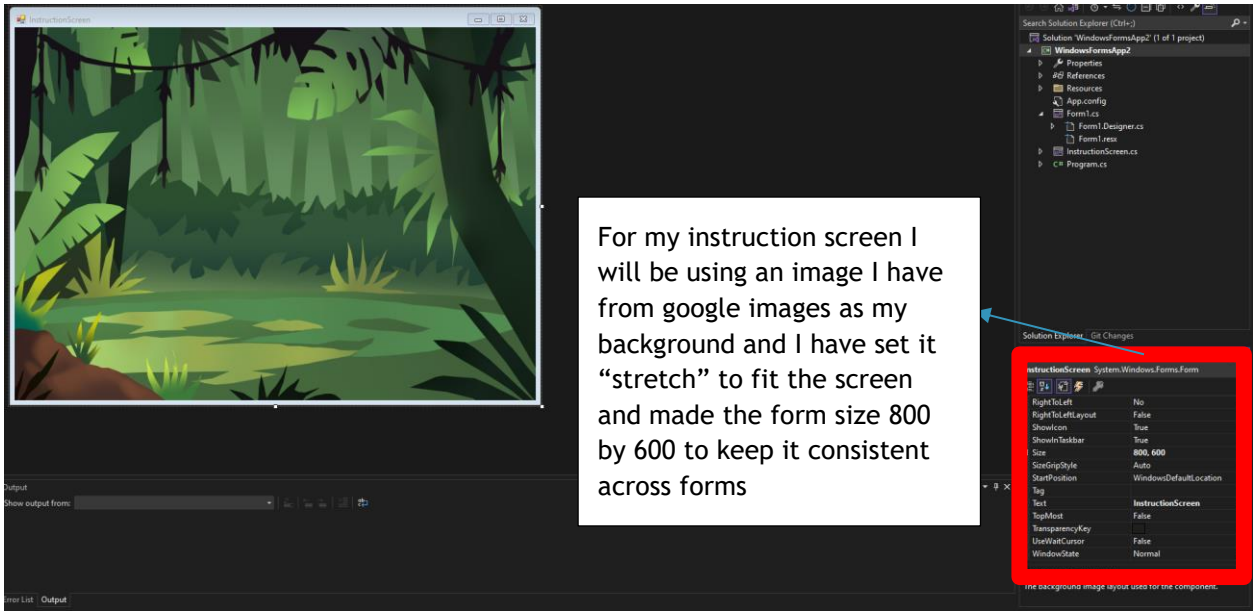
To be able to use the buttons for the intended uses, I had to create 2 more forms which are the Level Selector and the instruction screen. This will allow me to put the code in the buttons to redirect me to the appropriate screen.



To add the form I had to right click the project solution name, click add and then click the windows forms button. This allows me to create a different form screen for things such as levels death screen etc.

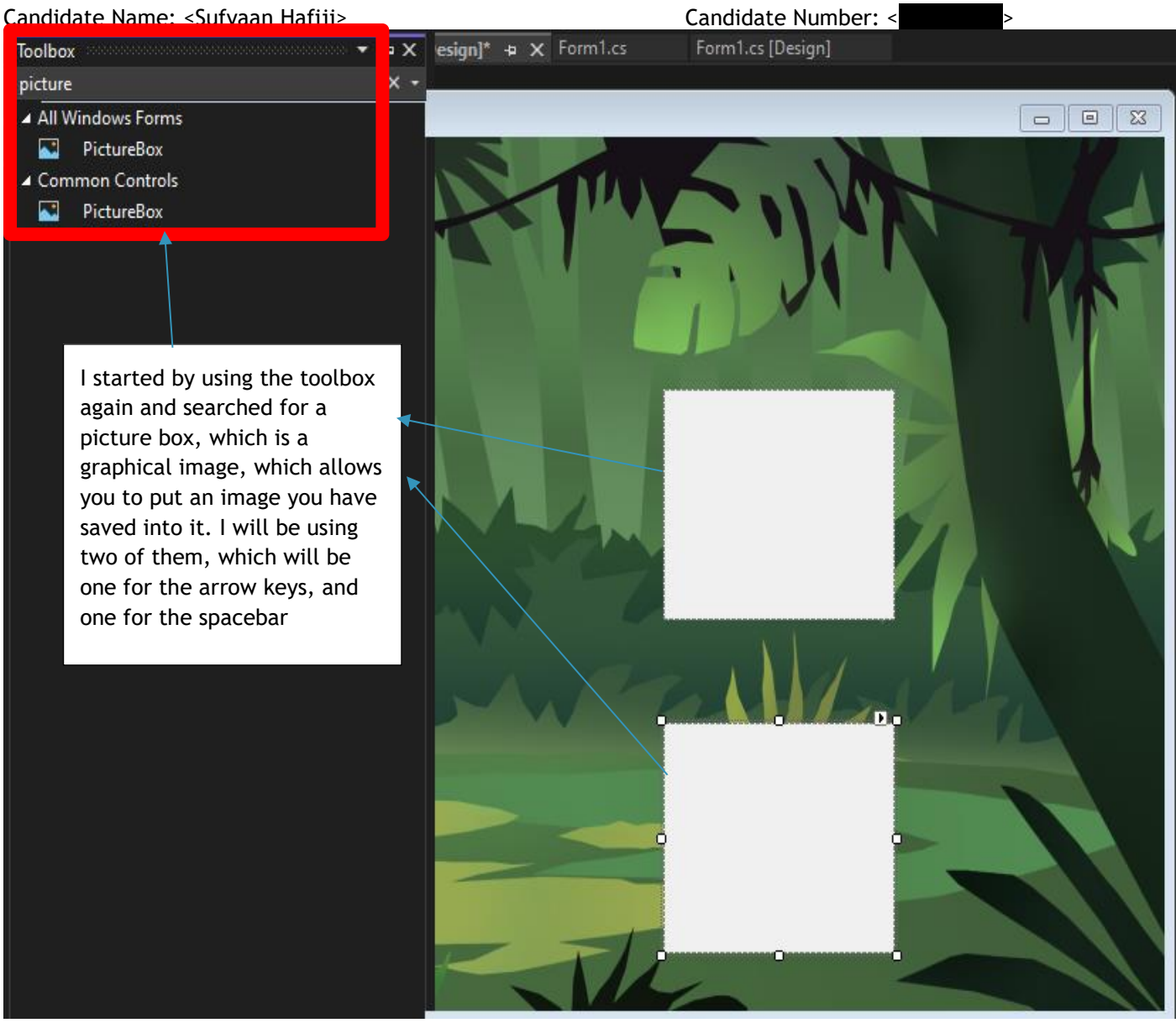


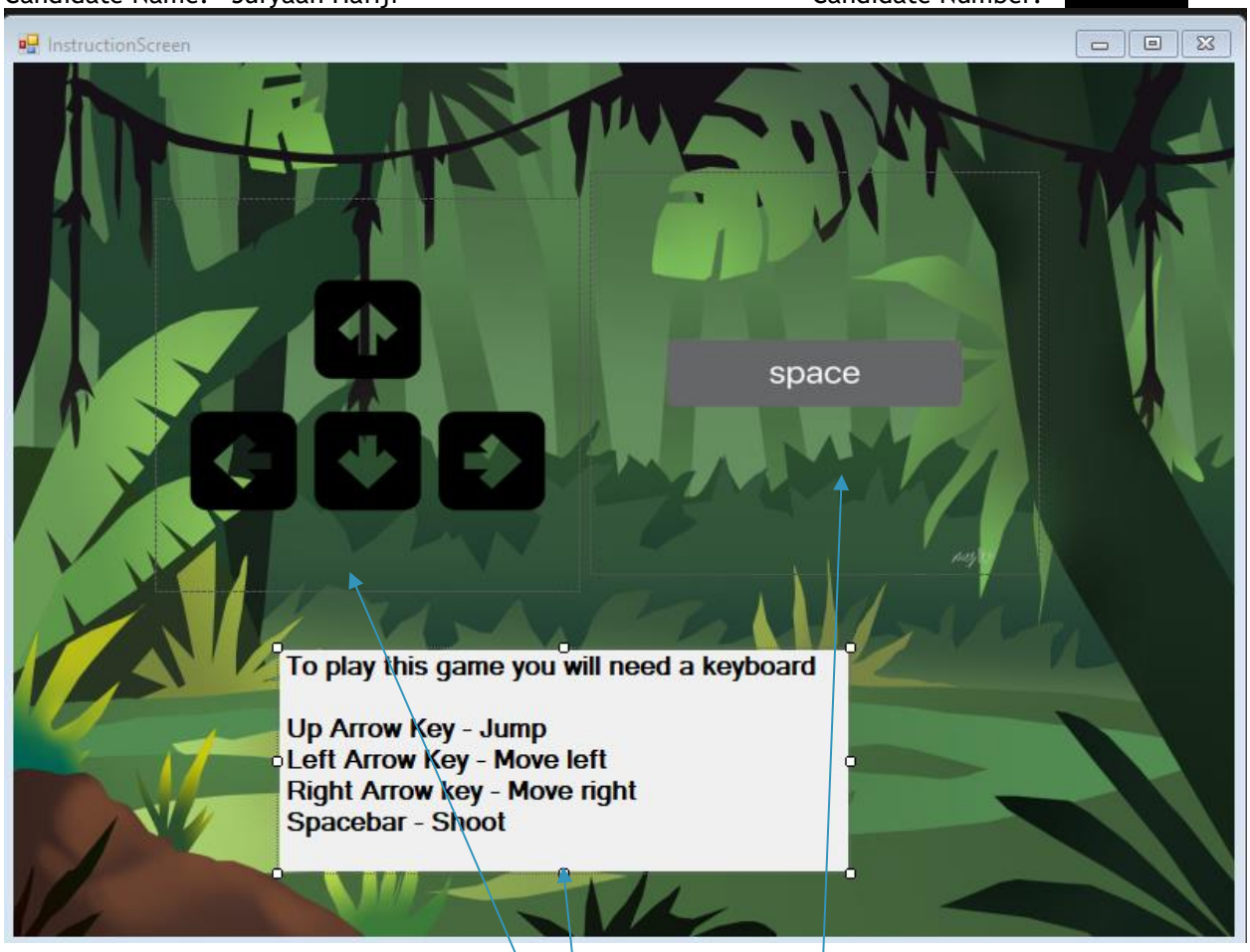
Clicking the option, it opens up this window, which allows me to choose what type of form I want to add and what name I want to change it to. For my game, I will be adding a form and will be naming the form appropriately, to what I needs it to be.



For my instruction screen I will be using an image I have from google images as my background and I have set it "stretch" to fit the screen and made the form size 800 by 600 to keep it consistent across forms

For my instruction screen, I will be using my design from the design section of having the keys you need to play the game and the explanation of what they will do and how to use them


Toolbox
picture
All Windows Forms
PictureBox
Common Controls
PictureBox
esign]*
Form1.cs
Form1.cs [Design]
I started by using the toolbox again and searched for a picture box, which is a graphical image, which allows you to put an image you have saved into it. I will be using two of them, which will be one for the arrow keys, and one for the spacebar



For the images, I used google images for them and saved them and put them into my resources folder inside visual studios for easy access to them. This also prevents me from losing any data because it is saved into the game. I also added a label, which allows me to write an explanation of the keys and how to use them

Now that I have added another form in my game, this allows me to use my button on the start screen to switch forms and make the old form invisible/removed.

```
private void StartButton_Click(object sender, EventArgs e)
{

}

1 reference
private void Instructions_Click(object sender, EventArgs e)
{

}
```

By double clicking the buttons on the form screen, Visual Studio automatically creates a private void, which identifies this block of code or procedure as method, or it will not return any values. The { } are needed to contain the code

```
private void Instructions_Click(object sender, EventArgs e)
{
    InstructionScreen gameWindow = new InstructionScreen();
    gameWindow.Show();
    this.Hide();
}
```

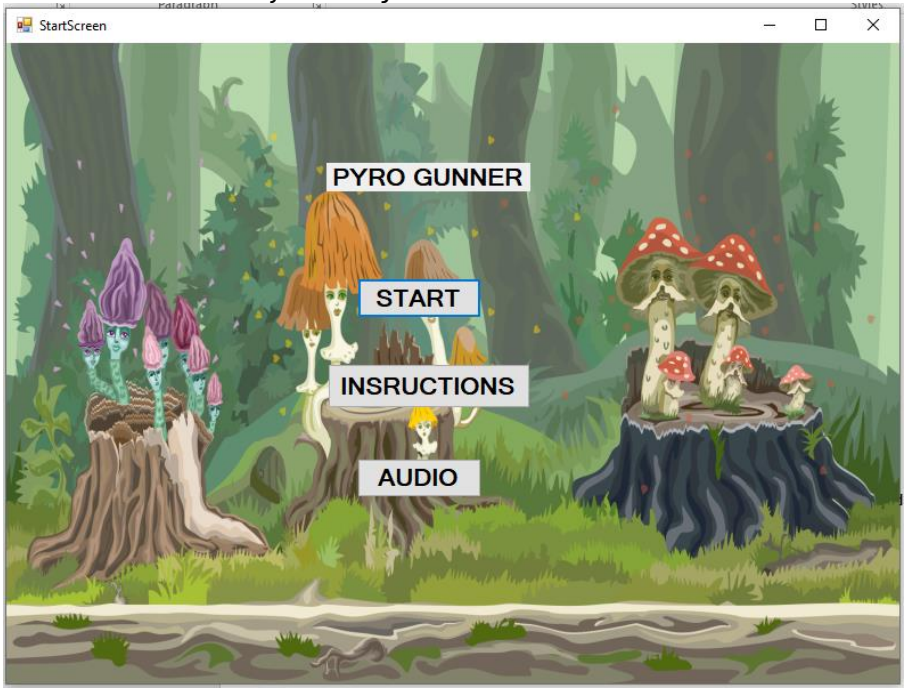
InstructionScreen is the name of the form I want to switch to so declaring this in the code will allow the computer to recognize which form you want to switch to

gameWindow is a class and you have set the variable to that class to InstructionScreen which would open the game window when the code is running

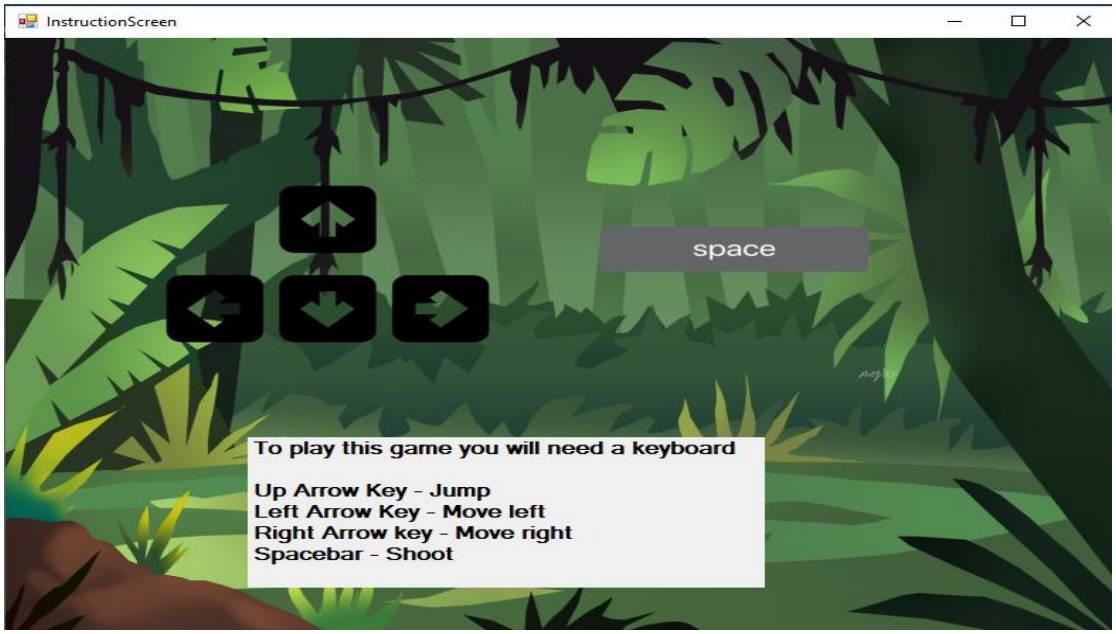
This.Hide and .Show are ways of making a form visible or invisible and in the code I have written it for the gamewindow of Instruction screen to be shown and the start screen to be made hidden. This will allow the player to see the instruction screen while not having multiple forms open which will look better visually

After this I tested the instruction button to see what would happen if I clicked it and it was successful

BEFORE

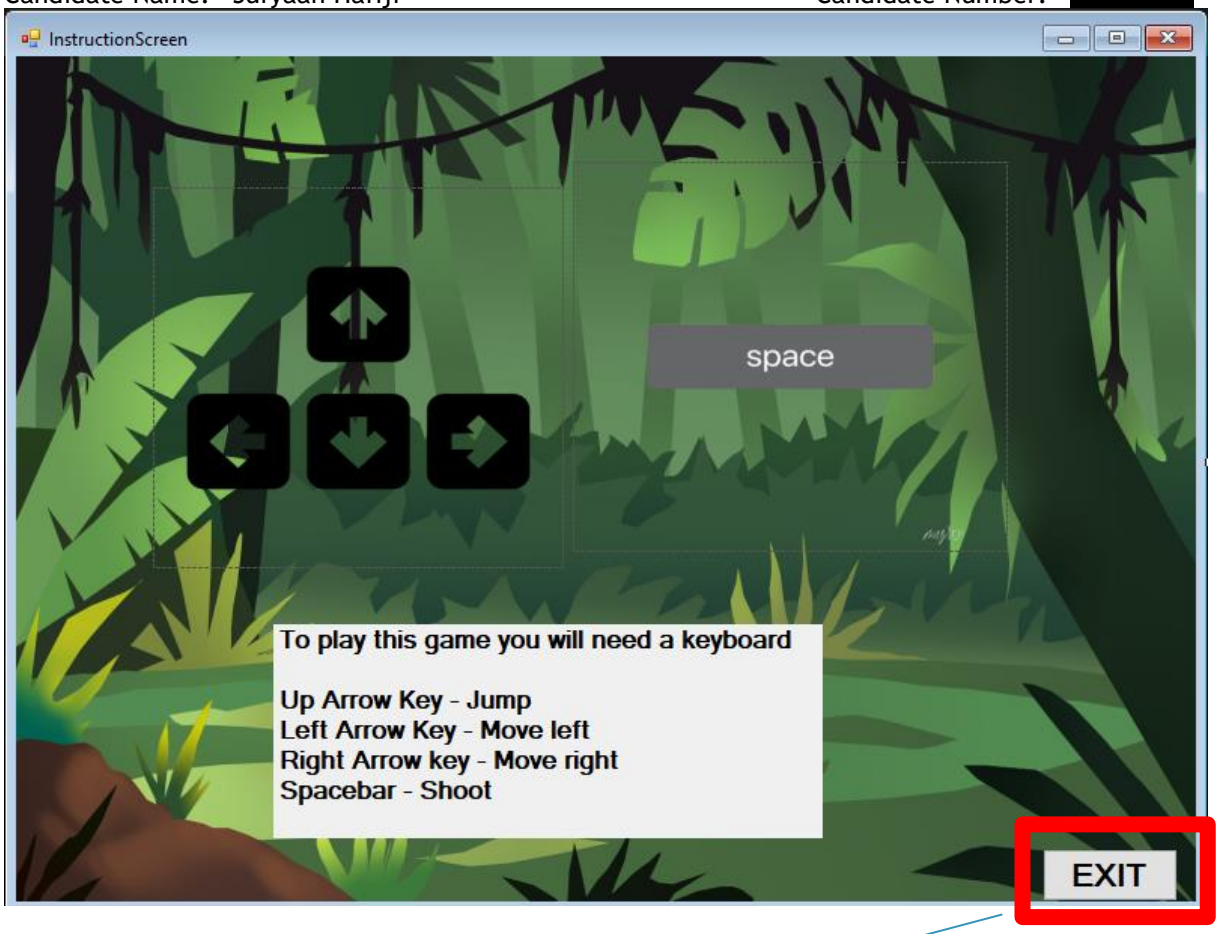


AFTER



This test was successful as the InstructionScreen was shown and the StartScreen visibility was set to false.

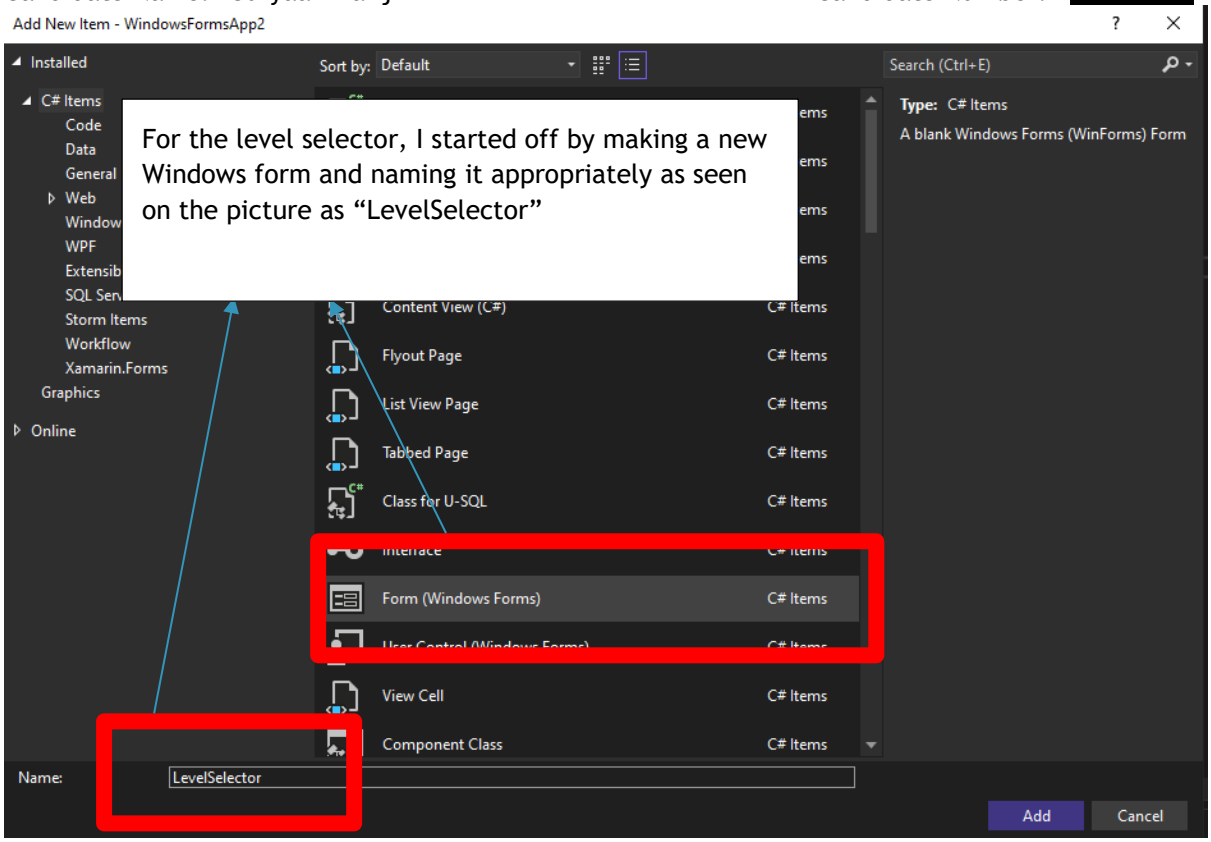
After my first button was a successful test I decided to add a exit button on the instruction screen as a way for the player to back out and not just stay on the same screen. This will allow to exit and enter the instruction screen when they want.



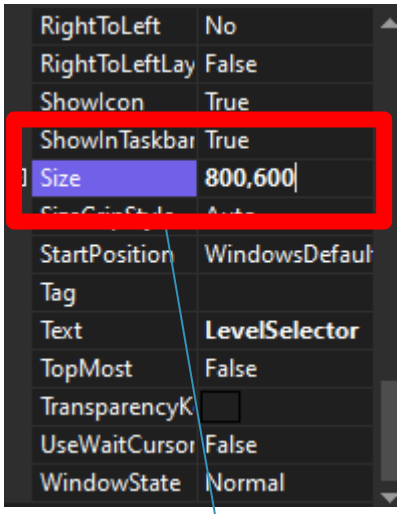
The exit button allows the user to enter the StartScreen form with a click and it removes the instruction screen window as you can see from the line “this.Hide();”. This hides the form on the button click and shows a new gamewindow

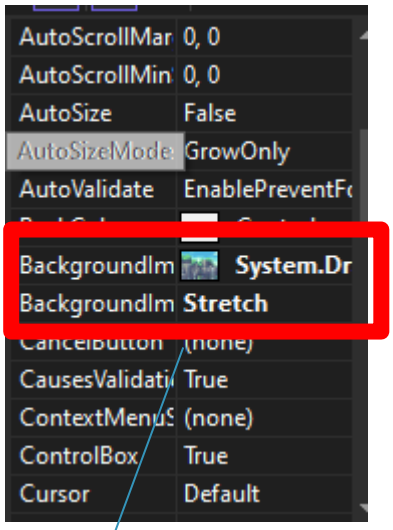
```
10
11
12
13      tionScreen : Form
14
15      )
16      {
17          InitializeComponent();
18      }
19
20      private void Exitbutton_Click(object sender, EventArgs e)
21      {
22          StartScreen gameWindow = new StartScreen();
23          gameWindow.Show();
24          this.Hide();
25      }
26  }
27 }
28
```

After implementing the exit button, I decided to test it to see the results and it was successful showing the startscreen when clicking the button. After the instruction screen was made and fully working, I decided to move onto a level selector with the similar design to the sketch I made on the design section.

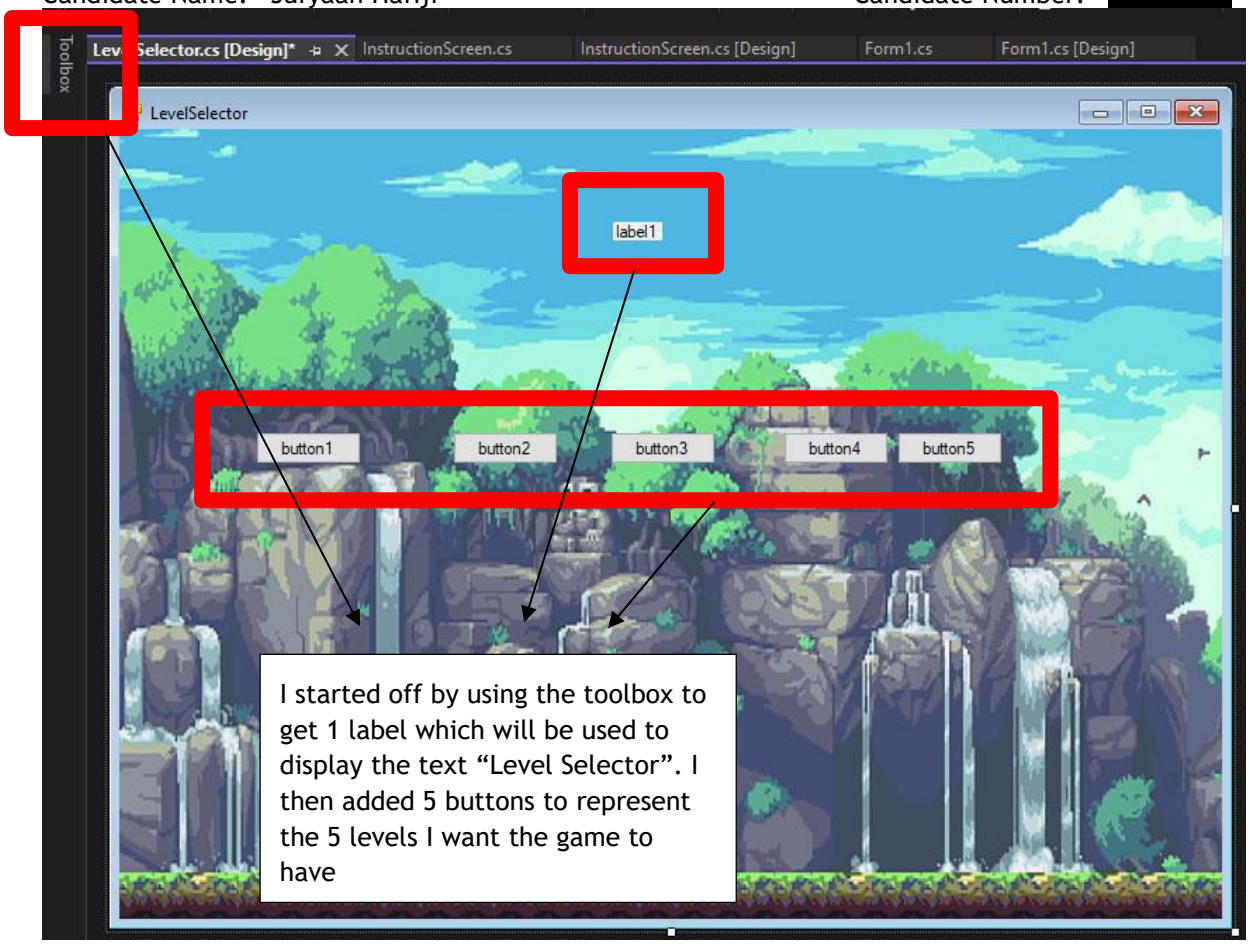


For the level selector, I started off by making a new Windows form and naming it appropriately as seen on the picture as “LevelSelector”

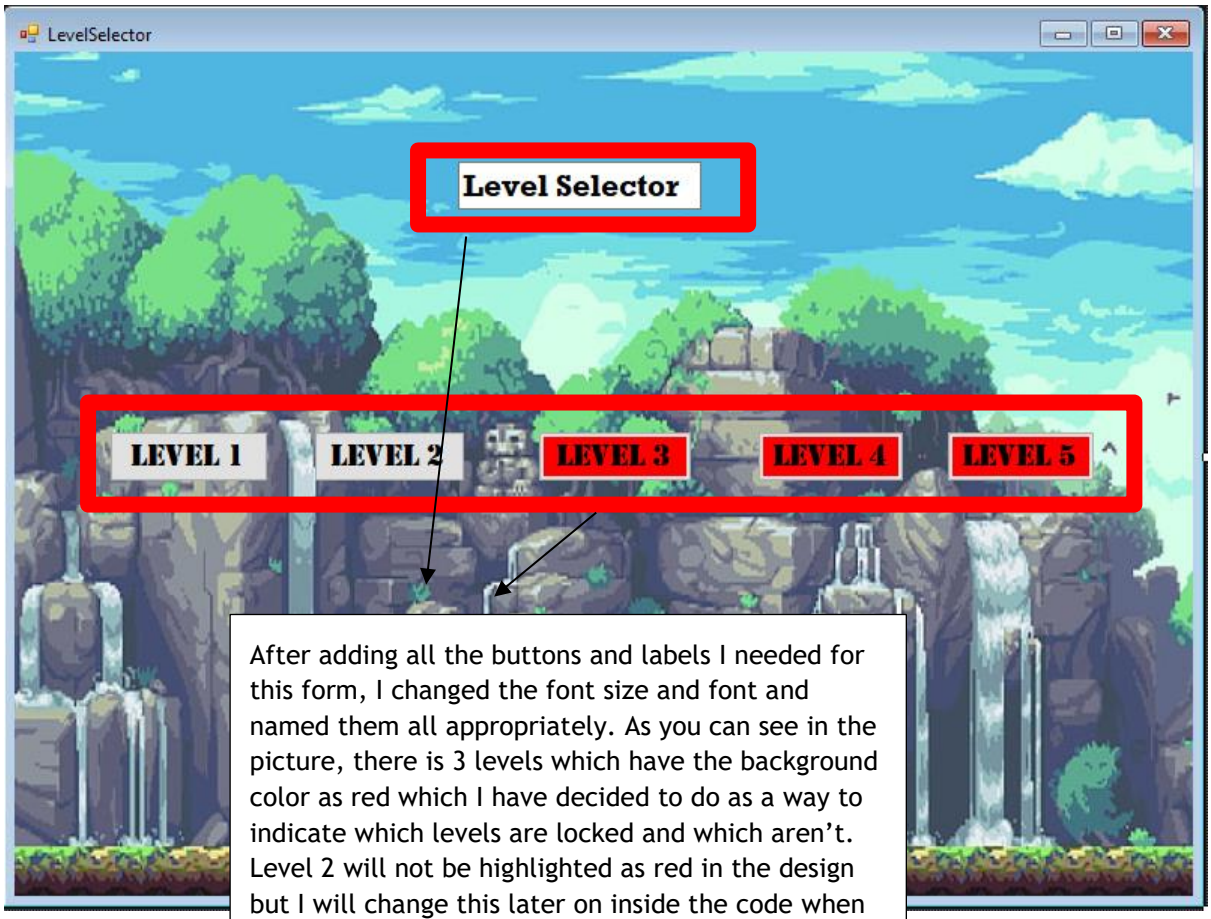




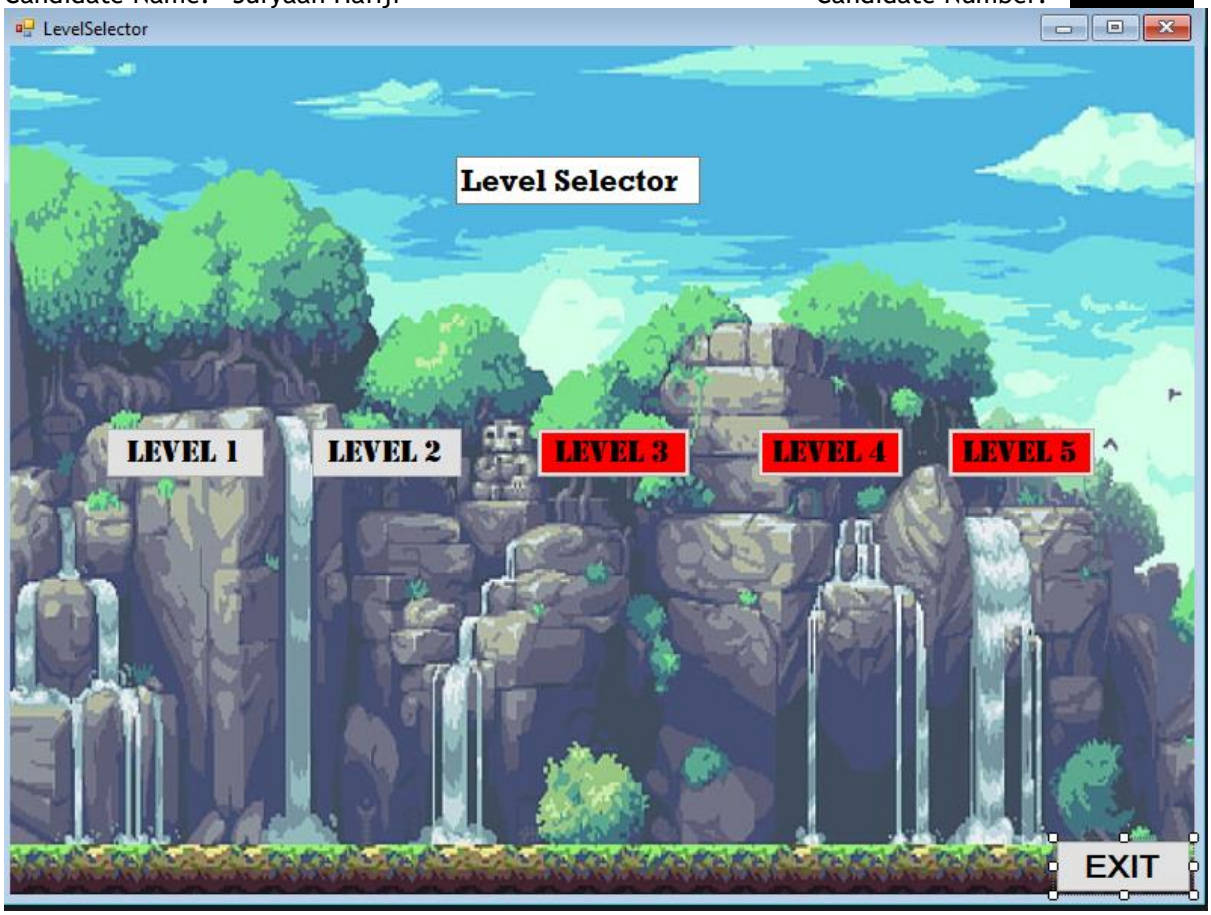
After creating the form for the LevelSelector, I changed the form size from the default to 800 by 600 which is consistent with all my other forms so far to keep it a similar size and design. I used a background image from the internet as the Level Selector background and I changed the background image style to stretch to fill the whole form from the centre.



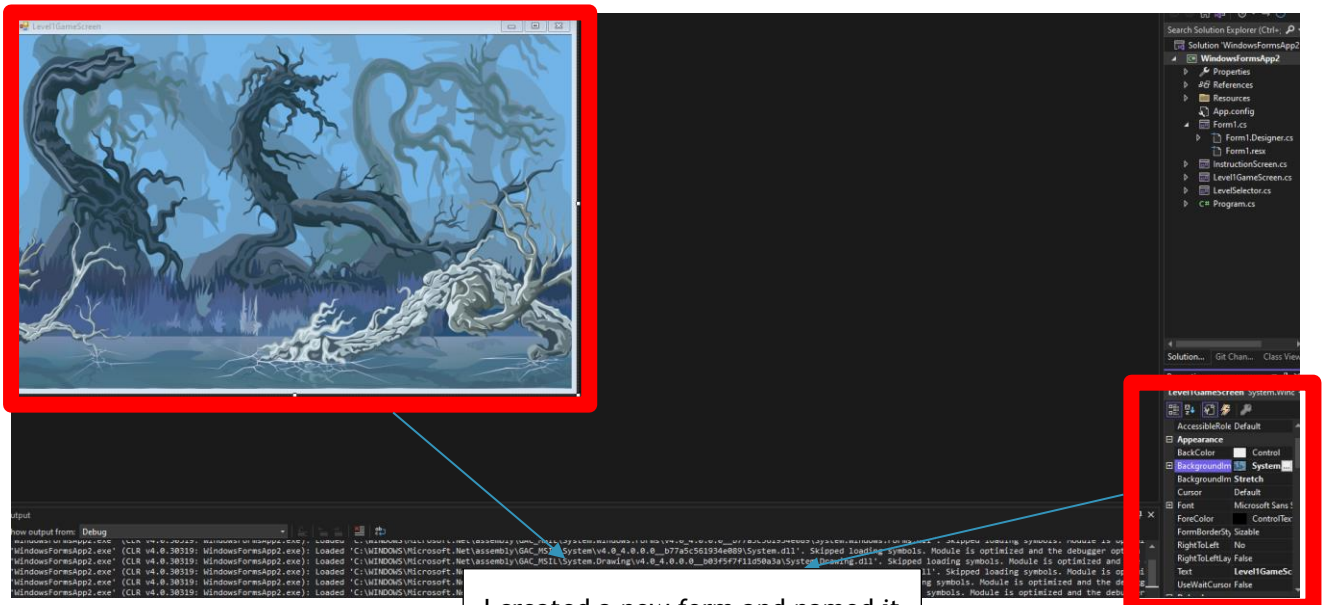
I started off by using the toolbox to get 1 label which will be used to display the text “Level Selector”. I then added 5 buttons to represent the 5 levels I want the game to have



After adding all the buttons and labels I needed for this form, I changed the font size and font and named them all appropriately. As you can see in the picture, there is 3 levels which have the background color as red which I have decided to do as a way to indicate which levels are locked and which aren’t. Level 2 will not be highlighted as red in the design but I will change this later on inside the code when the game runs



I also added a exit button for the players ease to get back to the start screen if they decide they need to look at the instruction screen. After this, I decided to make another form for my first level.



I created a new form and named it "Level1GameScreen" as this was going to be my first level. I changed the background and stretched it to fit the form and the size of it which is 800 by 600

```
private void L1_Click(object sender, EventArgs e)
{
    Level1GameScreen gameWindow = new Level1GameScreen();
    gameWindow.Show();
    this.Hide();
}

1 reference
private void Exitbutton_Click(object sender, EventArgs e)
{
    StartScreen gameWindow = new StartScreen();
    gameWindow.Show();
    this.Hide();
}
```

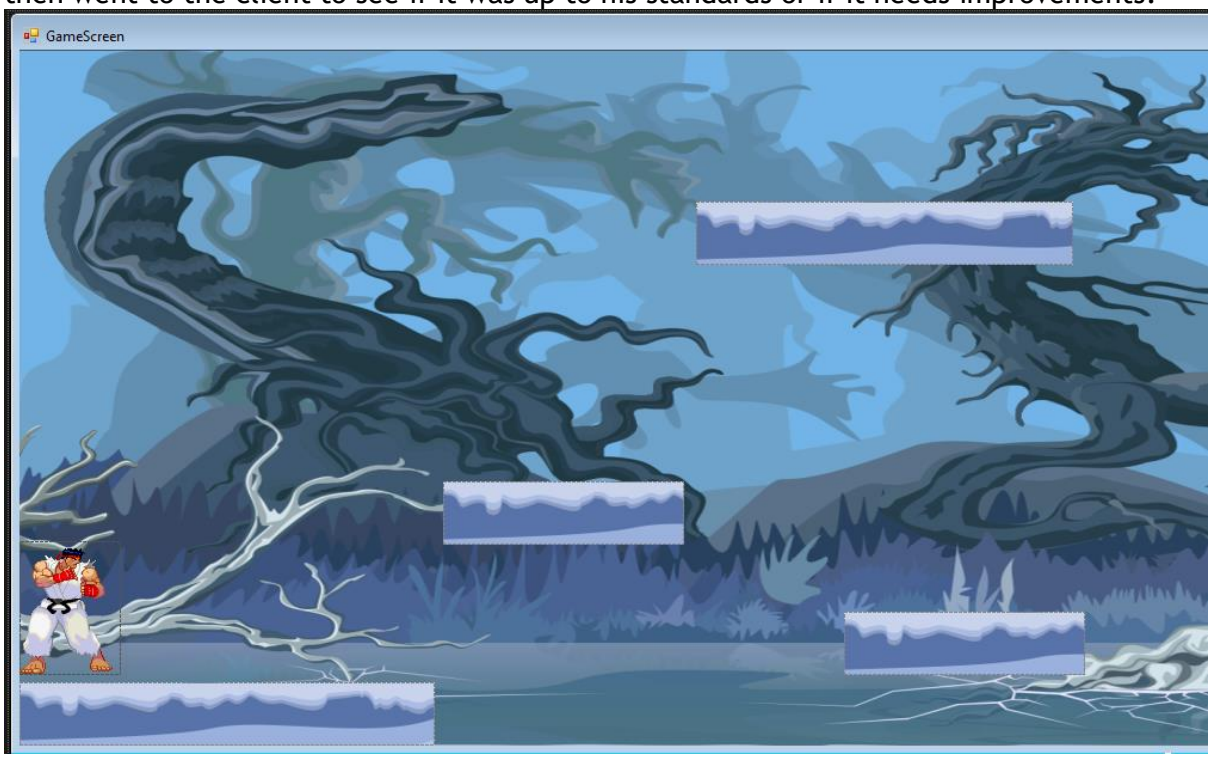
After creating the first level form, I was now able to link my level 1 button from the level selector the form which can be seen above and also the exit button to the start screen

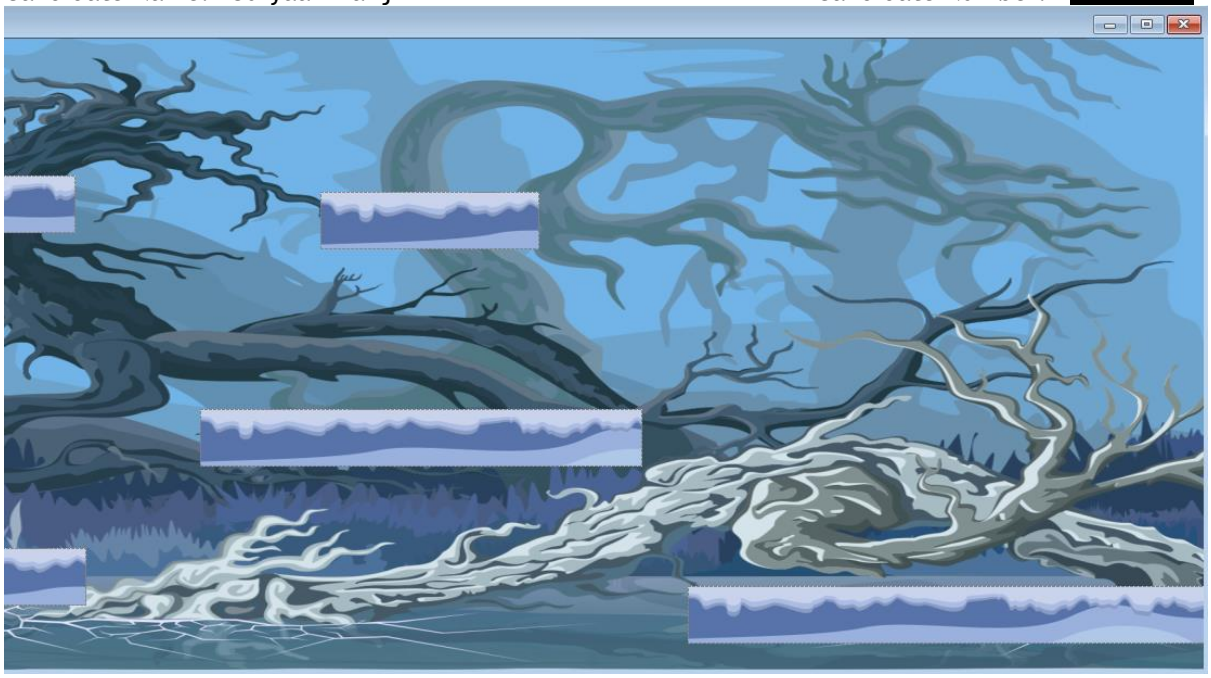
After successfully creating a level selector in my game, I left Level 2 without a colour so I can add into the code to make it red by default but then switch to white once the first level is cleared. As you can see in the picture below, I have set the button backcolor to red and created a variable in gamescreen that can switch it to white

```
1 reference
private void LevelSelecter_Load(object sender, EventArgs e)
{
    L2.BackColor = Color.Red;
    if (GameScreen.unlocked == true)
    {
        L2.BackColor = Color.White;
    }


}
```

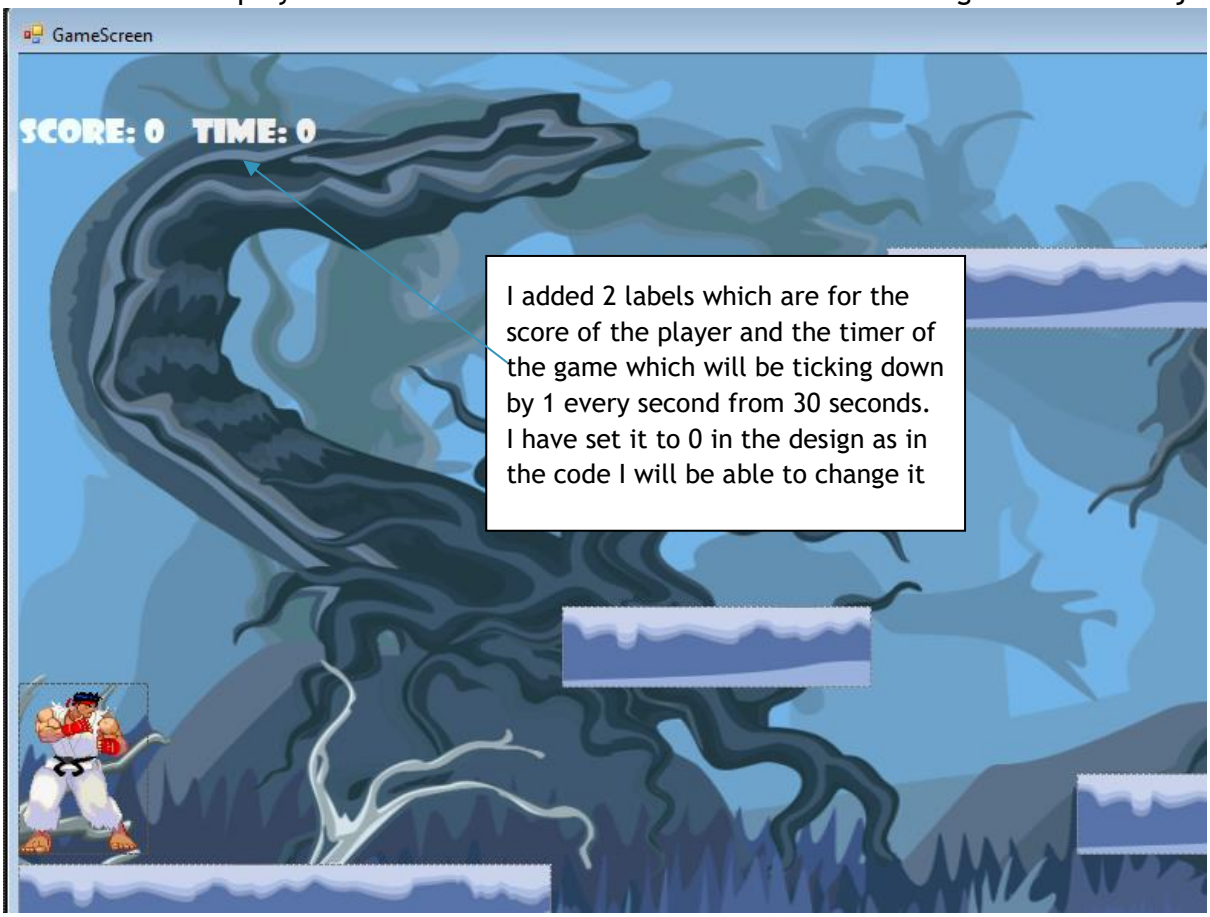
Creates a private void for the level selector screen and changes the backcolor of the level 2 button to red. Then a if statement occurs if the variable in gamescreen "unlocked" is true, it will change the backcolor of the button to white

Going in a linear fashion, I decided to now design my first level and how it should look and then went to the client to see if it was up to his standards or if it needs improvements.



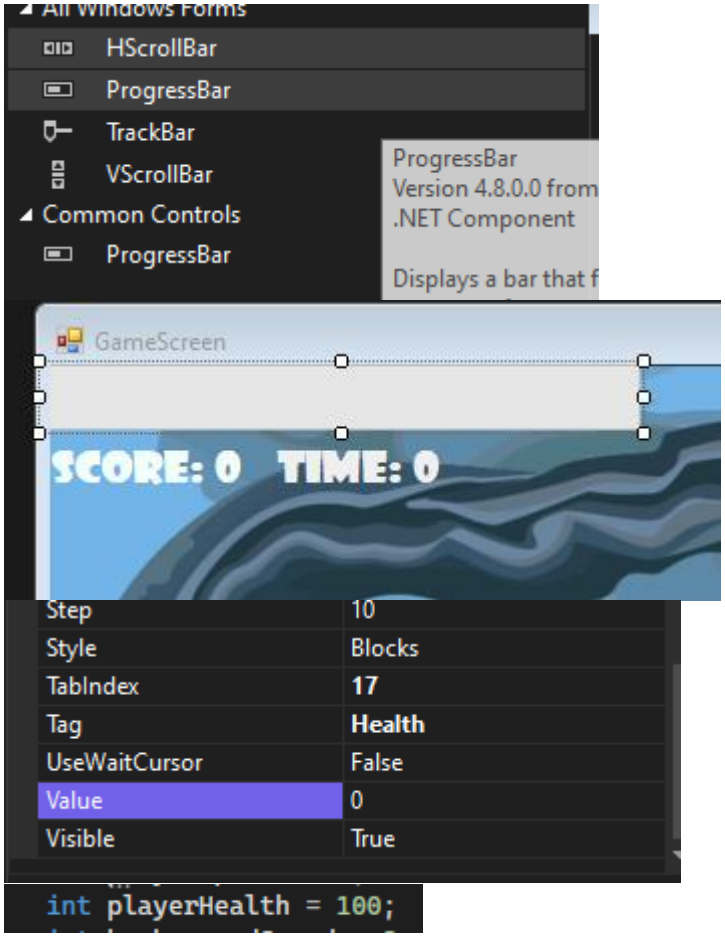


I have stretched the resolution of the game to fit more platforms so the game isn't too small and isn't interesting or fun. The platforms are going to be solid which means the player wont be able to fall through them and it will allow them to jump on them and walk on them. The player I have used will also be able to move left and right and fire and jump.



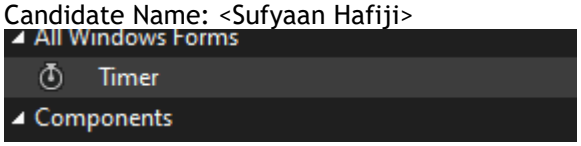
After doing this, I decided to finish the UI and the important information for the player which is now the healthbar. The client requested me to add a health bar but not make it have too much health or make the enemies do more damage so the game isn't too easy

Before: What I did to make it

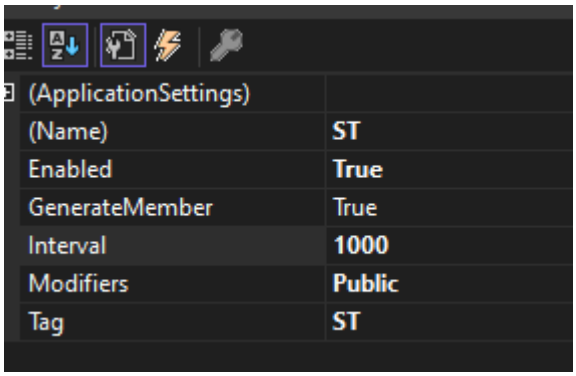


Testing it and seeing if it works

After this successful test it was now time to make my timer work and create a timer for it. This will allow the timer to be reduced by 1 by making a timer that will tick and every tick it does it will reduce a number on the timer



The timer is a component that raises an event at user defined intervals which allow me to make things like the frame per second of the game and what intervals the score goes down



After creating the timer, I changed the name to ST short for score timer and I set the interval at 1000 which is 1 second. This will allow me to code the label number being taken away every second

```
public static int scoretimer = 30;
```

Ive made a variable called score timer and made it public so I can access the timer/variable from other forms. I will be making the timer 30 seconds to make it not too easy or too hard

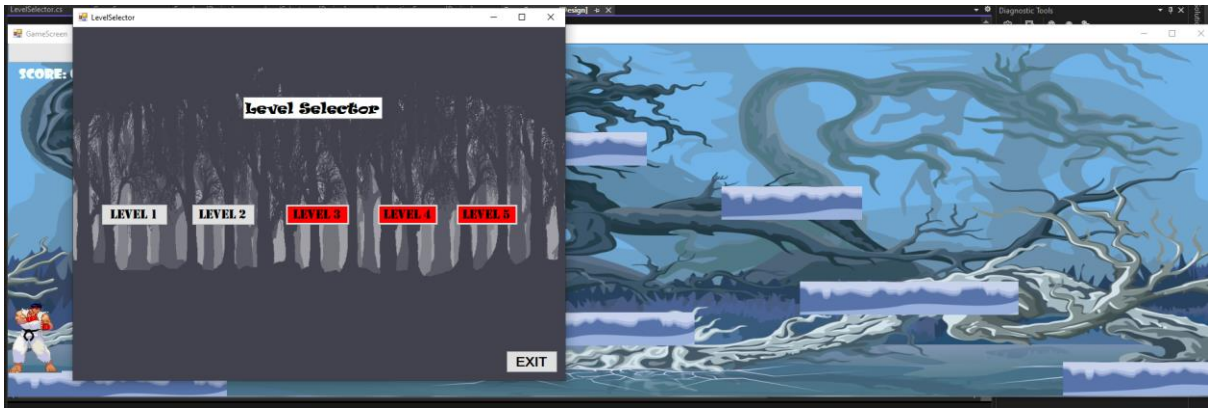
```
1 reference
private void ScoreTimer1(object sender, EventArgs e)
{
    //A score timer for the gamescreen which minuses 1 away every second from the set variable time and updates the time in the textbox
    scoretimer--;
    Time.Text = "TIME: " + scoretimer;
```

This allows it to minus it by 1 and it adds the number onto the label I have created which is the TIME

After creating the score timer I decided to test if it worked and ran the game but I ran into a bug.

TESTING FORM SCREENS CHANGING (ISSUE/BUG)

When I ran the game, the first screen would load as intended and then when I clicked the start button it loaded the level selector form and also the level form without me clicking it.



The correct way I intended this to work was when I clicked the first level, it would then hide the level selector form and then open up the level 1 form. To fix this I decided to look at my code for potential errors or things im missing to help me achieve this and fix this issue so it has a level of immersiveness that will not make my game look bad for my stakeholders

```
private void LevelSelector_Load(object sender, EventArgs e)
{
    GameScreen gameWindow = new GameScreen();
    gameWindow.Show();
    this.Hide();
    GameScreen.score = 0;
    GameScreen.scoretimer = 30;
}
```

This is the piece of code that is used to open up the first level window and its suppose to also hide the level selector form but this is not working properly. After looking at the code everything is correct and should work but it still wasn't working so I decided to look at the other code I used for the startscreen form.

```
private void start_Click(object sender, EventArgs e)
{
    LevelSelector gameWindow = new LevelSelector();
    gameWindow.Show();
    this.Hide();
}
```

This is the code for the start screen and comparing it to the previous screenshot the code it all looked correct. After testing this out more I realised that the function I wrote the code in was incorrect as it was the loading of the form which made my level load in the same time the level selector.

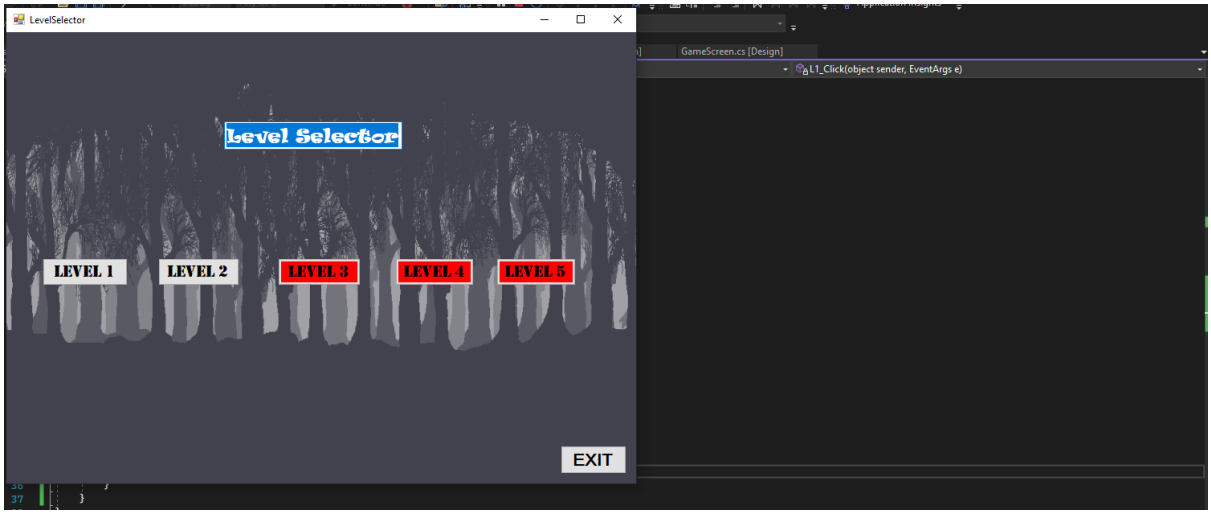
SOLUTON TO FORM CHANGINGE PROBLEM #1

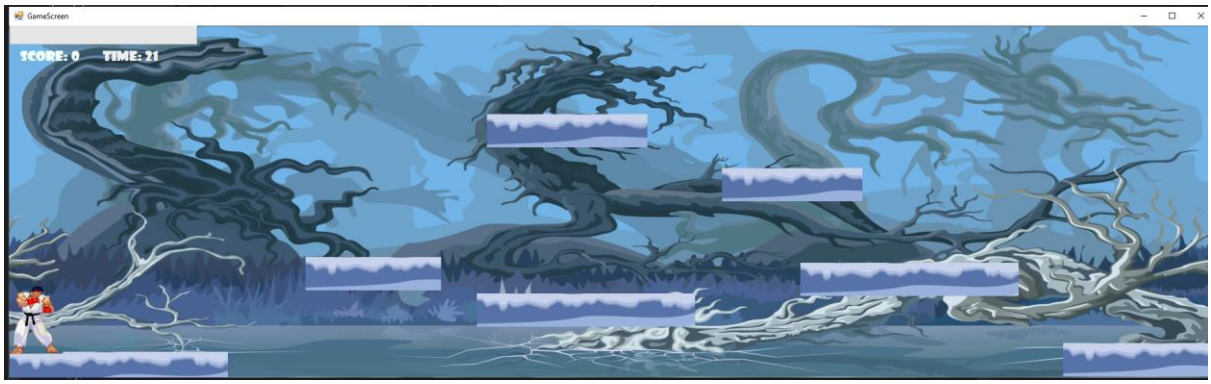
I created a new function for the button that is used to select level 1 and tried putting the code in there to test if it was going to work.

```
private void L1_Click(object sender, EventArgs e)
{
    GameScreen gameWindow = new GameScreen();
    gameWindow.Show();
    this.Hide();
}
```

This is the improved code that uses the same code I used on the load function but instead it is now in the button click function. After this I decided to test it again if it worked.

TESTING THE SOLUTION (SUCCESS) #1





This was a <u>successful</u> test and showed that my solution to the bug/error will not be in the final product of the game and gives the immersiveness that my stakeholders seek in a game.

After fixing my bug I then decided to test if my timer worked correctly and it was working as intended.

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Does timer work as intended	No input needed	No input is needed because it's a timer and nothing will change it from going down quicker or slower	Timer works as intended at 1 second interval	NA
Does main screen work as intended	Left click on START button. Left click on INSTRUCTION button. Left click on AUDIO button	I need all the button inputs to work as intended for the game to work and change screens	All the buttons work as intended except the AUDIO button as it does not do anything	The AUDIO button may be solved on a later date as its not a crucial part of my game so I will change it later
Does level selector work as intended	Left click on level selector button on main screen	As it's a button a left click should only work for it to execute the code inside	Level selector form loads and main screen form is hidden	NA
Does instruction screen work as intended	Left click on instruction button on main screen	As it's a button a left click should only work for it to execute the code inside	Instruction screen is loaded up and main screen form is hidden	NA
Does GameScreen work as intended	Left click on level 1 button on level selector screen	As it's a button a left click should only work for it to execute the code inside	Gamescreen form is loaded up and the main screen form is hidden	NA

REVIEW AND FEEDBACK (P1)

I have now completed the first prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.
A quick interview with REDACTED has been made (Available in Appendix R.1)
Summary of the conversation:

- No changes need to be made in the game.

As this is just form designs its expected to have no changes or improvements

REQUIREMENTS WORKED ON

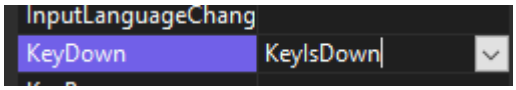
Requirements	Justification	Met?	Justification
Main menu	The game must have a main menu and a quick menu in-game to allow for navigation in the solution.	Yes	This requirement was fully met because it was a essential part of development of my game and without it my game wouldn't be efficient

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
Menus/Forms	**The menu will be simple and not too complicated. There may only be 2-3 options on the menu so its easy to navigate and easy to start the game.**	Yes	This is the same as the requirement above and my game would not be finished or able to work without having menus or forms
Different backgrounds	**Game must have a variety of background so it looks more professional and immerisve**	Yes	This was fully met as it was easy to use a variety of different backgrounds for my game

PROTOTYPE 2 (CHARACTER MOVEMENT)

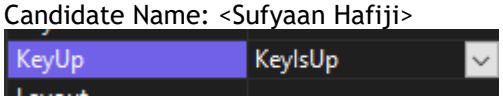
After setting up the basics of my game which I may come back to, to refine and do some final touches I have started to make my actual game that will the stakeholders will be playing. I will start off by writing code and doing iterative testing on the movement of the character.



This KeyDown function is used to register if the player is pressing a key on the keyboard which will allow me to manipulate and use code to check what keys are being pressed and writing corresponding code to the keys pressed which will also be the movement and shooting.

```
private void KeyIsDown(object sender, KeyEventArgs e)
{

}
```

Visual studios automatically creates a function when the properties of the game are changed or added to allow code to be written inside that will only execute if the conditions are met.



This function allows visual studio to register when the key is not pressed down and does the same thing as the KeyDown function

```
private void KeyIsUp(object sender, KeyEventArgs e)
{

}
```

The same function as KeyDown is made in the code but its for KeyUp instead

```
//movement
int playerSpeed = 10;
bool goLeft, goRight, jumping;
int jumpSpeed = 10;
int force = 8;
```

For the movement, I created variables as integers that allow me to go specific speeds such as how fast I will jump and what the height I will jump at. The playerSpeed will be the speed on how fast I move and I created Boolean integers for the going left and right and jumping that will allow me to use the previous keydown and keyup functions to do specific things

```
private void KeyIsDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Left)
    {
        goLeft = true;
    }
}
```

Ive used the Boolean variable goLeft and goRight from the previous picture to set it to true if the key left/right arrow is pressed to allow me to execute code corresponding code in another function

In the function KeyDown ive added a IF statement to set conditions if a key is pressed, I want the game to execute something. In this picture ive added a IF statement for the button left/right arrow on the keyboard to achieve my success criteria of controls

```
if (e.KeyCode == Keys.Right)
{
    goRight = true;
}
```

```
if (e.KeyCode == Keys.Up && jumping == false)
{
    jumping = true;
}
```

For the jumping mechanic, I added a IF statement and check if the arrow key UP was pressed and also checking if the Boolean variable jumping was false, it would set it to true. This would allow me to a mechanic of jumping into the game by using the timer to execute code that will move the picture box up whenever the keycode UP was pressed

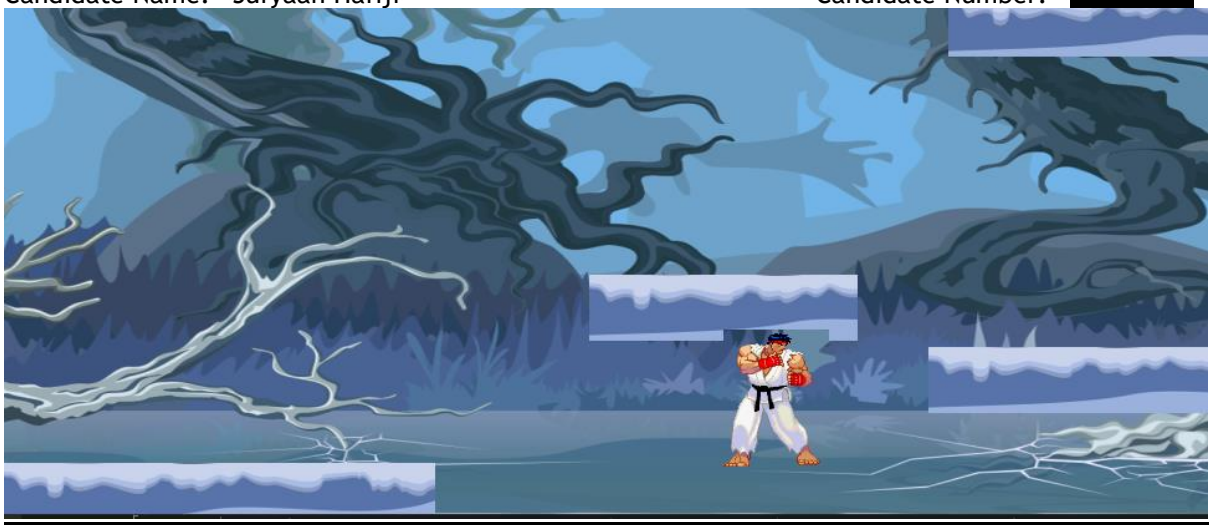
```
private void KeyIsUp(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Left)
    {
        goLeft = false;
    }
    if (e.KeyCode == Keys.Right)
    {
        goRight = false;
    }
    if (e.KeyCode == Keys.Up && jumping == false)
    {
        jumping = false;
    }
}
```

I did the opposite for the KeyUp Boolean variables which set the variables to false once the key is not pressed down. This will allow the character to stop jumping or moving when the key is not pressed allowing the player to control the character properly.

```
private void MainTimerEvent_Tick(object sender, EventArgs e)
{
    if (goLeft == true && Player.Left > 60)
    {
        Player.Left -= playerSpeed;
    }
    if (goRight == true && Player.Left + (Player.Width + 60) < this.ClientSize.Width)
    {
        Player.Left += playerSpeed;

    }
}
```

In this screenshot of code I have added an IF statement that executes code that make the character move right or left depending on which variable is being changed to true. If goLeft is true and the users character is more than x=60 on the form, it will make set the players speed to negative to allow the character move backwards which is also left on my form. If goRight is true it makes the playerspeed value a positive so it go forward which is also back. On the goRight if statement it also check if the players width is les than the forms size.

TESTING MOVEMENT (ISSUE/BUG)



This test was successful as I could move my character backwards and forwards but there is a issue with the game as the player is able to go through the platforms instead of the player being on top of it.

SOLUTION TO MOVEMENT PROBLEM #1

```
foreach (Control x in this.Controls)
{

}
```

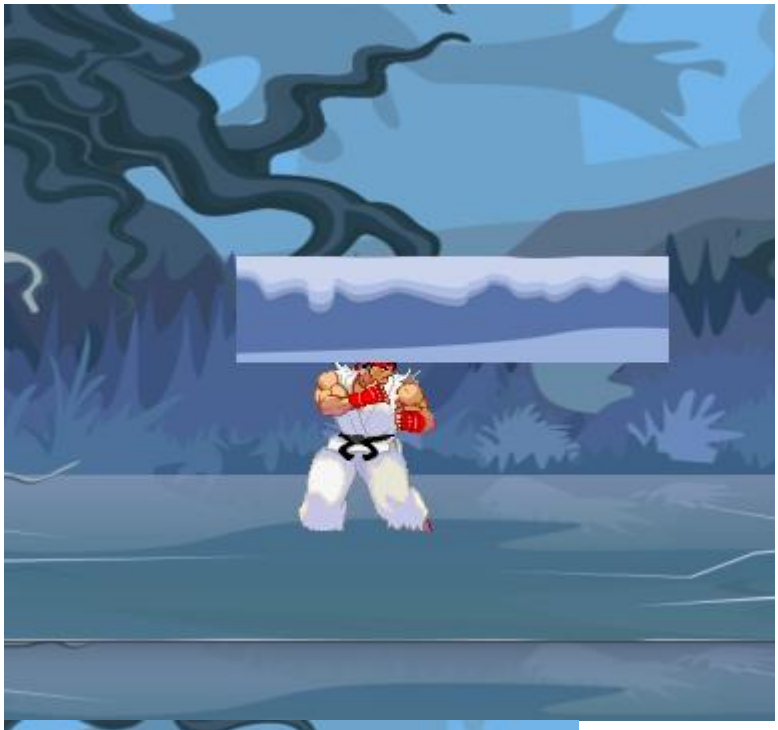
I started off creating this foreach loop inside the maintimerevent function so I don't need to write a statement for each piece of code inside that will be interacting with the player.

```
if (x is PictureBox && (string)x.Tag == "platform" || x is PictureBox && (string)x.Tag == "platform1")
{

}
```

Inside the function I added this if statement that makes it so if the picturebox interacts with the image name platform it will execute a certain piece of code

```
if (x is PictureBox && (string)x.Tag == "platform" || x is PictureBox && (string)x.Tag == "platform1")
{
    //This forces the players picturebox to move upwards and the top of the player is put above the platform
    //and the jumpspeed is reduced to 0. It also changes the boolean variable is grounded to true
    //which executes another function when the condition is met
    force = 8;
    Player.Top = x.Top - Player.Height;
    jumpSpeed = 0;
    isgrounded = true;
}
```

TESTING THE SOLUTION (FAILURE) #1



This was a unsuccessful test as the game is glitching and sending the character up and down and even creating 2 versions of the character and the player was also not falling down when its not interacting with a platform so I will have to fix the code or add something that will fix it.

SOLUTION TO MOVEMENT PROBLEM #2

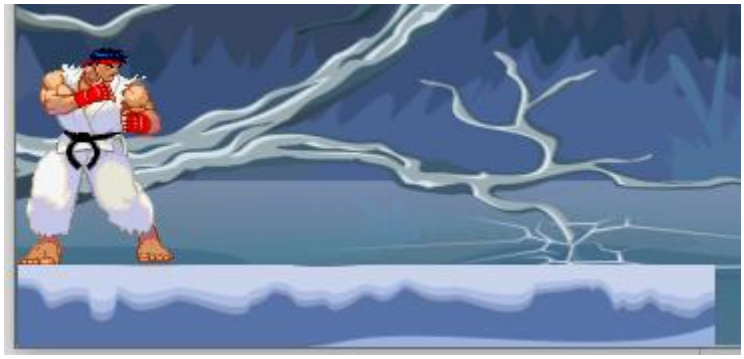
As my code was logically correct and I didn't have errors in the code, I had to look at what the code was executing for it to be doing this issue. The problem was not in writing the code, but that I was missing something that didn't cause the player to go up or down and jitter. To try fix this issue I added a if statement to see if it would work as its executing on the condition of something being true.

```
if (x is PictureBox && (string)x.Tag == "platform")
{
    if (Player.Bounds.IntersectsWith(x.Bounds) && jumping == false)
    {
        //This forces the players picturebox to move upwards and the top of the player is put above the platform
        //and the jumpspeed is reduced to 0. It also changes the boolean variable is grounded to true
        //which executes another function when the condition is met
        force = 8;
        Player.Top = x.Top - Player.Height;
        jumpSpeed = 0;
        isgrounded = true;
    }
```

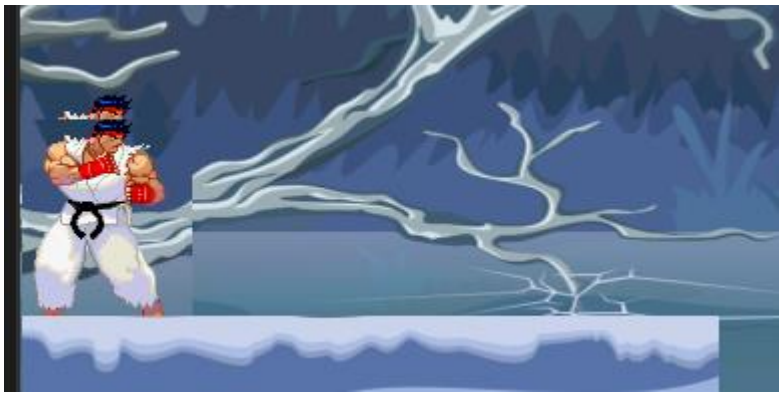
The if statement I added was to check if the players picturebox area was touching the platform bounds, it would execute the code that was inside it and set the jumping to false so it didn't make the player keep going up and down

TESTING THE SOLUTION (SUCCESS) #2

After I added this if statement to my code I decided to test my game again and see if its doing the same problem as last time.



I successfully fixed the problem my game had as the player was no longer jumping up and down so this solution to the problem was successful.



Although the test was successful I was having frame issues with my game as it would jitter slightly due to the background and the picturebox of the player so it gave a sort of blur to it but its only a visual bug and not a code error that drastically changes the way the game plays so I may come back to this at a later date to try fix the problem.

The player falling down was also a successful test and I have shown this through adding another platform below the player and they can drop down onto the lower platform without falling through it or the player bugging by going up and down.

I also tested the player jumping and that was now successful after fixing that code.

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Character moving left	Left arrow key	As the stakeholder wanted simple inputs I decided to use arrow keys	Character moves left on the game screen	NA

		for input to move the player		
Character moving right	Right arrow key	As the stakeholder wanted simple inputs I decided to use arrow keys for input to move the player	Character moves right on the game screen	NA
Character jumping	Up arrow key	As the stakeholder wanted simple inputs I decided to use arrow keys for input to move the player	Character moves upwards on the game screen	NA
Physics of the the character jumping	Up arrow key	As I don't want the player to just jump and stay in the air I needed to test the code I made that it will bring the character back down	Character comes back down after they jump	NA
Game platforms	All arrow keys	The player needs to stay on top of platforms to make the game playable	Player does not fall through the floor	NA

REVIEW AND FEEDBACK (P2)

I have now completed the second prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.

A quick interview with REDACTED has been made (Available in Appendix R.2)

Summary of the conversation:

- **There is a bug when you hold down spacebar which lets the character keep going up**

MAKING THE CHANGES REQUESTED BY REDACTED

To fix this issue I created a Boolean variable that is set to true so when the player presses spacebar, it is set to false and then the spacebar if statement in keydown will not execute if the bool is not set to true. This should fix it as it wont execute until the character is touching a platform.

```
bool isgrounded = true;
```

```
if (e.KeyCode == Keys.Up && jumping == false)
{
    jumping = true;

}
```

```
if (e.KeyCode == Keys.Up && jumping == false && isgrounded == true)
{
    jumping = true;
    isgrounded = false;
}
```

TESTING THE SOLUTION (SUCCESS)

This test was successful as holding down the spacebar no longer kept executing the condition statement of jumping.

REQUIREMENTS WORKED ON

Requirements	Justification	Met?	Justification
Easy controls	REDACTED mentioned that he didn't want complicated controls for the game so it must be simple to use	Yes	This requirement was fully met because I used a common key control that most people know how to use

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
Controls	**He has requested to use arrow keys for movement and spacebar to shoot**	Yes	I successfully met this and made my controls the arrow keys for the player

PROTOTYPE 3 (GEMS+POWERUPS PICKUP)

GEMS

As part of my requirements and success criteria set by my stakeholder, after having successfully made movement in my game I decided to add gems and powerups. The concept of the two are very similar but they execute different procedures and functions as one updates a score counter and the other increases player health or speed.

First I created pictureboxes and set the images to gifs of a gem so that it looks more animated making it more immersive for the player to play.

```
Tag                gem
```

I set the tag of the picturebox to gem so its accessible throughout my code instead of using individual names which would be more tedious and time consuming to do. After this i created a if statement inside the foreach loop so it can execute code when the playerbox interacts with the gem picturebox.

```
if(x is PictureBox && (string)x.Tag == "gem")
{
    if (Player.Bounds.IntersectsWith(x.Bounds) && x.Visible == true)
    {
        score++;
        x.Visible = false;
    }
}
```

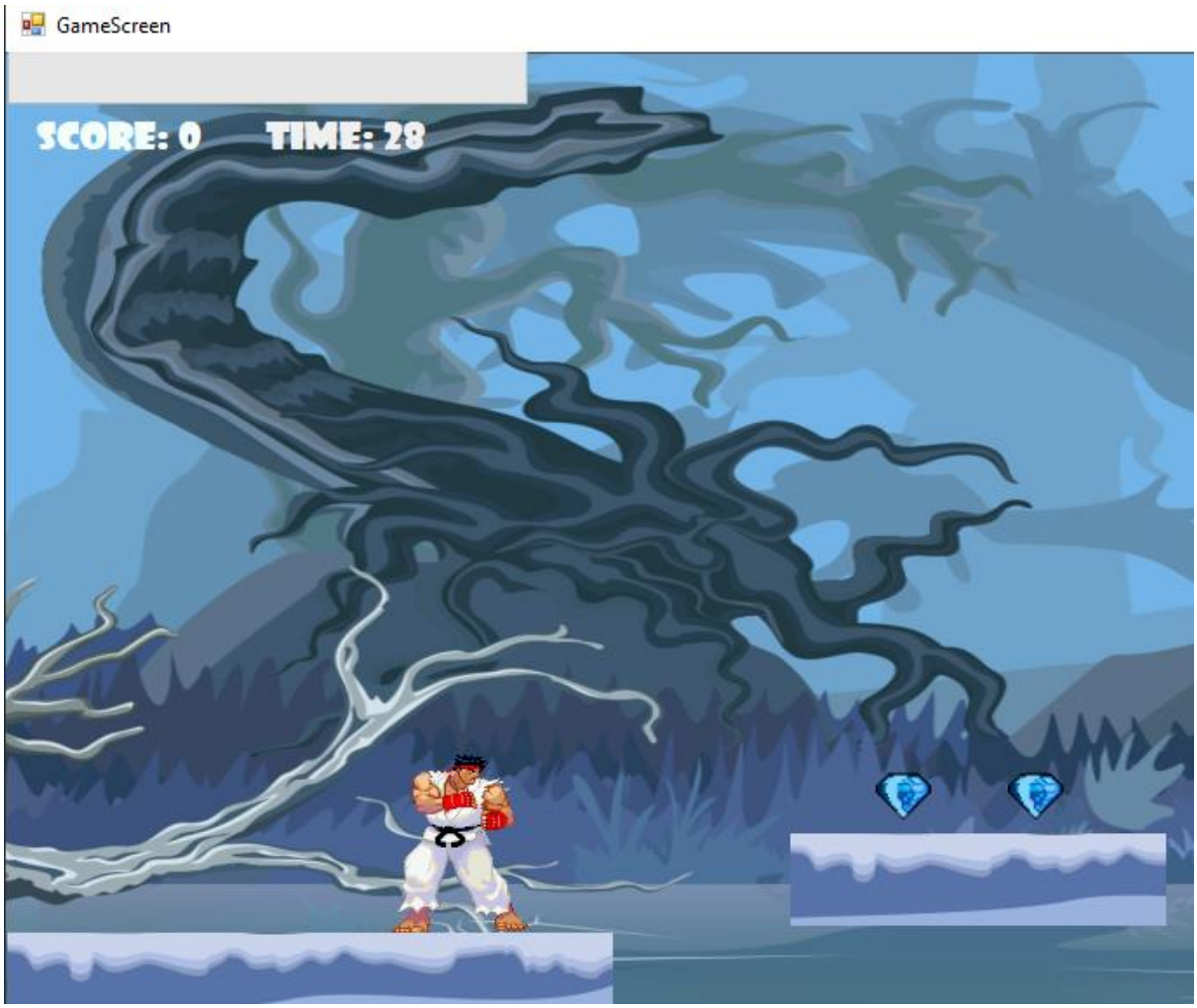
This line of code is a selection statement which checks if a condition has been met and then executes the code inside the brackets.

This makes the picturebox not visible to the player and removes it from the screen

This if statement checks if the players picturebox intersects with it and the gem picturebox is still visible

I also added a line of code that adds to the score everytime they interact with a gem. This helps the player keep track of their score and meets the requirements of the stakeholder of having a score.

TESTING THE GEM PICKUP #1 (SUCCESS+BUG)



GameScreen

This test was successful because the player interacted with the gem and it removed it and made it invisible but there is also a bug/issue. After collecting the gem it did not add to the score as I wrote in the code meaning it's a logic error as it runs as intended but not with updating the score.

SOLUTION TO THE PROBLEM

As it's a logic error, it means I have to either add or remove code that will allow the score to update when the gem is picked up by the player.

```
1 reference
private void MainTimerEvent_Tick(object sender, EventArgs e)
{

    Scorebox.Text = "Score " + score;
```

In the main timer event, I added a line of code that replaces the scoreboxes text with "Score: "+score which should allow the score to update when the gems are picked up.

TESTING THE SOLUTION #1 (SUCCESS)

After adding that line of code I decided to test the game again to make sure its working and picked up gems.

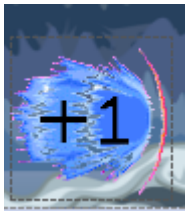
This test was successful and my solution was correct as the score updated when I picked up a gem. This now meets the stakeholders requirements of having a score to keep track of their progress in the game.

POWERUPS

AMMO ADDITION

As part of my stakeholders requirements, I need to add powerups in my game which allows the player to gain a boost or help to do something in the level. I will be currently implementing a healthboost and adding 1 to their ammo count.

I started off with making a picturebox of the fireball my player uses and added a +1 to make it obvious to the player that the ammo count will go up.



This is the image I will be using to show that it's a powerup. The code for the powerup will be similar to the gem pickup as it's the same concept but it changes different parts of the code.

```
if (Player.Bounds.IntersectsWith(ammo1.Bounds))
{
    ammo1.Visible = false;
}
```

I will be using this code for the removing the visibility of the picturebox when it comes into contact with the players picturebox.

TESTING IF THE AMMO IS REMOVED #1 (SUCCESS)

This test was successful as it removed the ammo powerup when the player interacted with it but now I have to add a ammo textbox and a variable to change when the if statement is executed.

SOLUTION TO AMMO ADDITION

I started off with making a visual representation of the ammo with a ammo box and I added a textbox that displays the current amount of ammo the player has. This will be set to 3 as the stakeholder REDACTED has requested at the start of the game and it will be able to reach maximum 4 in level 1.

```
int ammo = 3;
```

I created a integer variable and set it as 3 because you cant have a fraction of a ammo in my game. This will allow the game to set the ammo count as 3 everytime the level is started.

```
ammotext.Text = "Ammo= " + ammo;
```

After creating the integer variable, it allowed me to display the variable in the textbox which updates everytime the fireball is shot or a powerup is picked up

```
if (Player.Bounds.IntersectsWith(ammo1.Bounds))
{
    ammo1.Visible = false;
    ammo++;
}
```

Then I added a line of code inside the if statement inside the foreach loop to add ammo when the player comes into contact with the powerup

TESTING AMMO COUNT (ISSUE/BUG)

The ammo pickup worked but there was a bug in my code that kept increasing the ammo after its turned invisible so it cant be picked up. To fix this I need to add a piece of code that will correct this issue or alter the code to make it work properly.

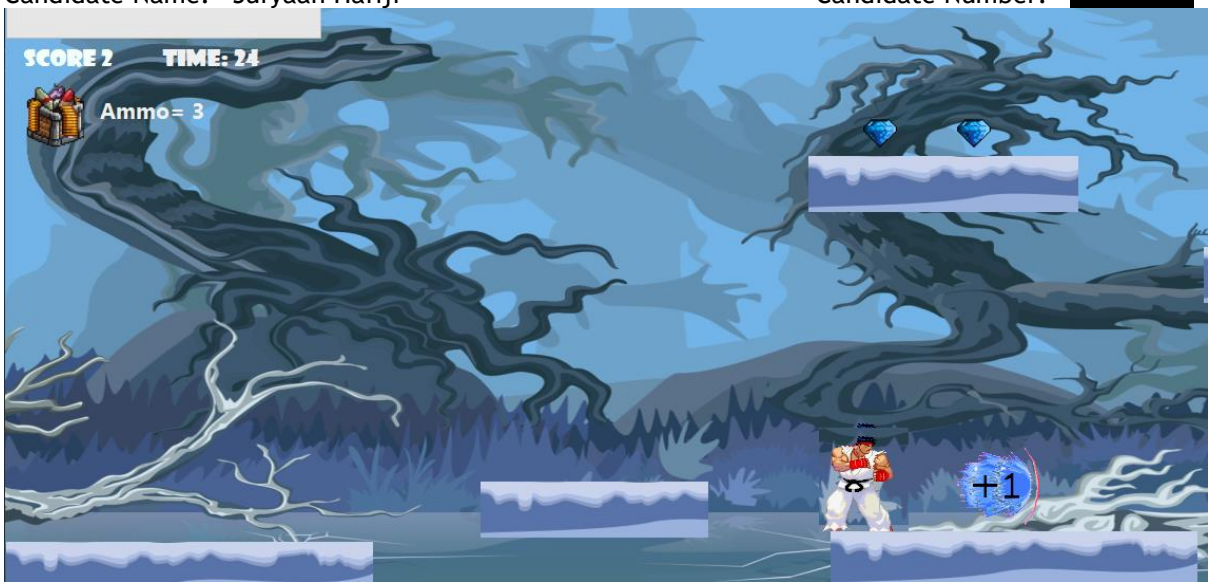
SOLUTION TO AMMO BUG

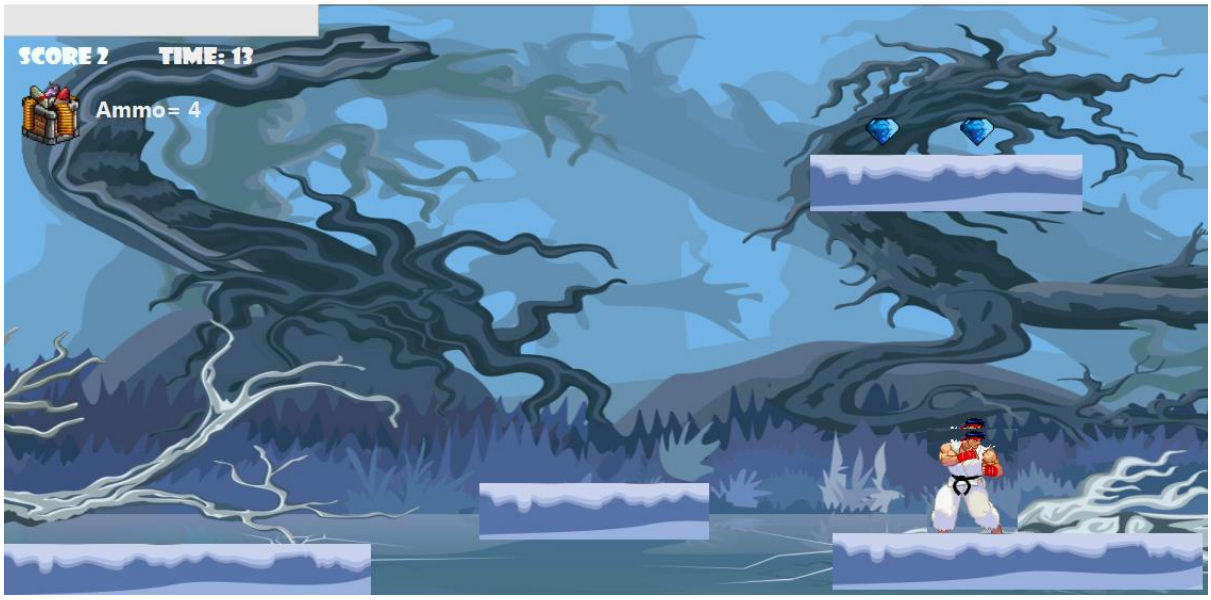
```
if (Player.Bounds.IntersectsWith(ammo1.Bounds))
{
    ammo1.Visible = false;
    ammo++;
    ammo1.Location = ammotext.Location;
}
```

This line of code moves the ammo powerup picturebox to the ammobox picturebox to make sure the code doesn't keep executing when its invisible

I tried to figure out a function to remove the picturebox when the games running and I couldn't find a solution to it so I decided to make the picturebox visibility false and then make the picturebox teleport to another picturebox in the gamescreen which is the ammobox and then it should stop it from adding ammo continuously

TESTING THE SOLUTION TO AMMO ADDITION





This was a successful test as the ammo only added one onto it and didn't keep continuously adding it.

What is being tested	Input	Justification of input	Outcome	How to solve

HEALTH BOOST

For my stakeholders requirements, I need to have a healthbar so I decided to also create a health boost powerup for the player so they can pick it up when they have been damaged so they can replenish their health.

I started off creating a picturebox and adding the medic sprite into it and then setting up the name and tags for it.

This is a simple design and is created from pixel art so it carries on the theme of a cartoonish design of the game and it also makes it easier for the user to understand it replenishes health

```
if (Player.Bounds.IntersectsWith(MedKit.Bounds))
{
    if (playerHealth == 100)
    {
        healthbar.Value = playerHealth;
    }

    else
    {
        playerHealth = playerHealth + 5;
        MedKit.Visible = false;
        MedKit.Location = AmmoBox.Location;
    }

}
if (playerHealth > 1)
{
    healthbar.Value = playerHealth;
}
```

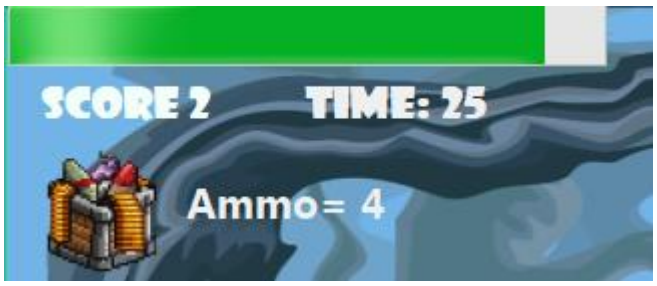
As all I need to in my code is to add onto my player health, making a health boost is code efficient and easy to make. In the screenshot, the game checks if the players picturebox bounds intersects with the medkit bounds and it sets the visibility of the medkit to false so the player cant see it after they touch it. If the playerhealth is still 100 the playerhealth value stays the same so it doesn't crash the game and if its below the value it adds onto the value of the players health. I also created a if statement that checks if the playershealth and if its above 1 it matches the healthbars value to the player health.

TESTING THE HEALTH BOOST (SUCCESS)

After creating the code I decided to test the game out to see if the game would replenish my health.

To test this out I needed a way to reduce my health and since I didn't have enemies in the game yet I decided to make a line of code in the ammo pickup function to reduce the players health value.

```
if (Player.Bounds.IntersectsWith(ammo1.Bounds))
{
    ammo1.Location = AmmoBox.Location;
    ammo1.Visible = false;
    ammo++;
    playerHealth = playerHealth - 10;
}
```



This test was successful and there was no errors in making the health boost

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Gem pickup	Player interacts with the gem picturebox	The gem should not be visible after the player interacts with it	Picturebox is not visible when interacted with	NA
Ammo pickup	Player interacts with the ammo picturebox	The ammo should not be visible after the player interacts with it	Picturebox is not visible when interacted with	NA
Score	Player interacts with gem picturebox	The score should go up by 1	Score goes up by 1 and displays it	NA
Ammo	Player interacts with ammo picturebox	Ammo count should go up by 1	Ammo goes up by 1 and displays it	NA

REVIEW AND FEEDBACK (P3)

I have now completed the third prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.
A quick interview with REDACTED has been made (Available in Appendix R.3)

Summary of the conversation:

- **Powerups meet his requirements of how they work**
- **He wants a player speed boost powerup**

MAKING THE CHANGES REQUESTED BY REDACTED

I started off by making a picturebox and adding a image to it. When the player interacats with this it will increase the speed of the player and add onto the playerspeed integer.

```
//speed powerup
if (Player.Bounds.IntersectsWith(speed.Bounds))
{
    playerSpeed = playerSpeed + 5;
    speed.Visible = false;
    speed.Location = AmmoBox.Location;
}
```

This is similar code to the other powerups so I don't need to explain this again.

```
ctureBox && (string)x.Tag == "flag" || x is PictureBox && (string)x.Tag == "speed")
```

I also put it inside the MoveGameElement function which will allow for it to move with the player and not just stay in one place.

TESTING THE SOLUTION (SUCCESS)

The testing was a success and the powerup was not visible when interacted with and the playerspeed increased.

REQUIREMENTS WORKED ON

Requirements	Justification	Met?	Justification
Powerups	The game must have some sort of boost or power-up feature so it isn't stale and the player has a way to improve at the game, which in turn increases the sense of achievement and the overall relaxing experience.	Yes	This requirement was successfully met as I made powerups that successfully changed the character variables inside the game

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
5 second powerups	**Power-ups will not last permanently and will last for 10 seconds before reverting the character to normal as it was before**	No	I could not meet this success criteria due to it being too difficult to add a temporary timer that would remove and add variables to the game
Gems	**There will be around gems every level and they will be placed in easy to reach spots not blocked my enemies and some will be behind enemies you have to defeat**	Yes	This success criteria was fully successful as I made gem pickups that allowed the gem counter to go up for the game

PROTOTYPE 4 (FIREBALL)

I started off with creating a picturebox and adding my fireball image into it and I made it a cartoon theme. The justification for this is that the stakeholder wanted a cartoon theme game and this is the closest image I could find that matched what he wanted.

```
//fireball
bool shortFireball = false;
int fireballSpeed = 20;
```

I created a Boolean variable that checks when the fireball is short and it sets it at false at the start of the game so it doesn't fire off automatically. I then created integer variable that has a speed of 20 and I can tweak it to however fast I want it to go

```
if(e.KeyCode == Keys.Space)
{
    if (ammo > 0)
    {

    }
}
```

I started off at creating a if statement in the keydown function and I created another if statement inside it to check if the ammo is more than 0 to execute.

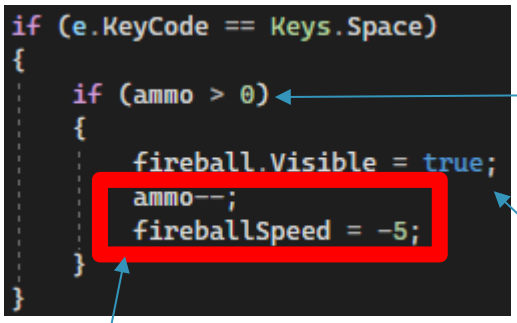
```
if(e.KeyCode == Keys.Space)
{
    if (ammo > 0)
    {
        fireball.Visible = true;
        ammo--;
        fireballSpeed = -5;

    }
}
```

This makes the fireball visible as the player pressed the spacebar

This removes the ammo everytime the spacebar is pressed

This sets the speed of the fireball when the spacebar is pressed

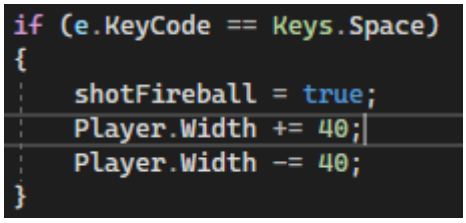
After this, I have to make it so the game is constantly checking if the spacebar is pressed so I wrote some code in the keydown and keyup function so it would only fire when the character is pressed. This screenshot is from inside the keydown function



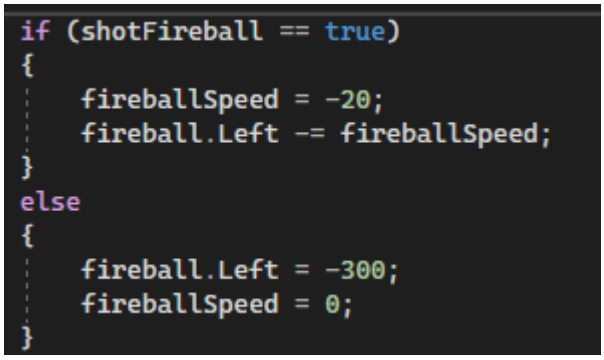
This if statement checks if the ammo variable integer is higher than 0 and if it is, it executes the code inside it

This reduces one ammo from the ammo counter and sets the fireball speed to -5 which makes it move right of the screen.

This sets the picturebox to visible and enables the fireball to be seen



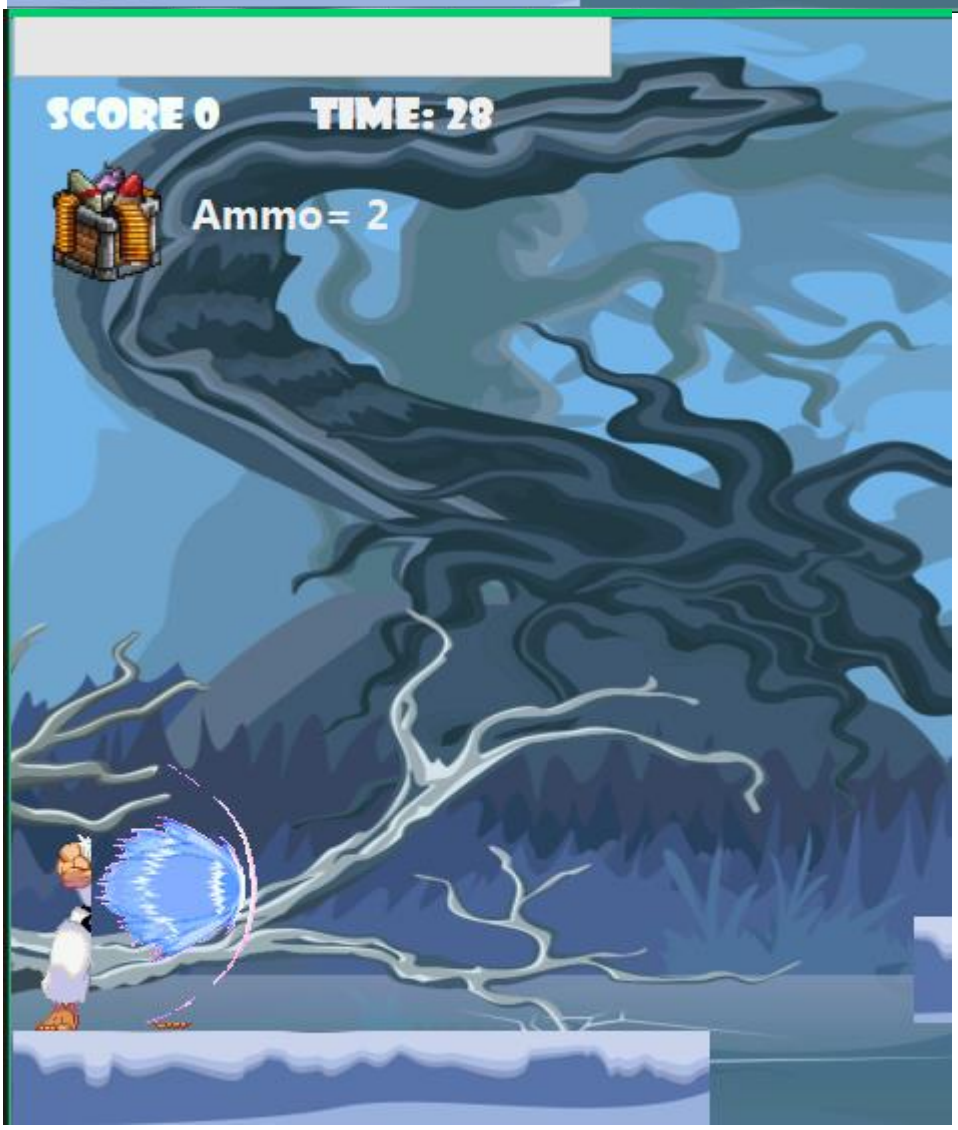
This is inside the keyup function which checks when the spacebar character is depressed and then is enables the shotFireball boolean variable to true which allows me to create a if statement in the main function to shoot the fireball and set it off. The players width is set so the fireball is shot from near the middle of the character picturebox.



This if statement is inside the maintimer function which is constantly checking when the shot fireball variable is set to true and then changes the speed of the fireball to -20 which allows the fireball move at a good speed to the right. The left of the picture box is set to the fireball speed variable. Otherwise the coordinate of the fireball is set to the player at all times.


SCORE 0
TIME: 27
Ammo= 3


SCORE 0
TIME: 29
Ammo= 2


SCORE 0
TIME: 28
Ammo= 2

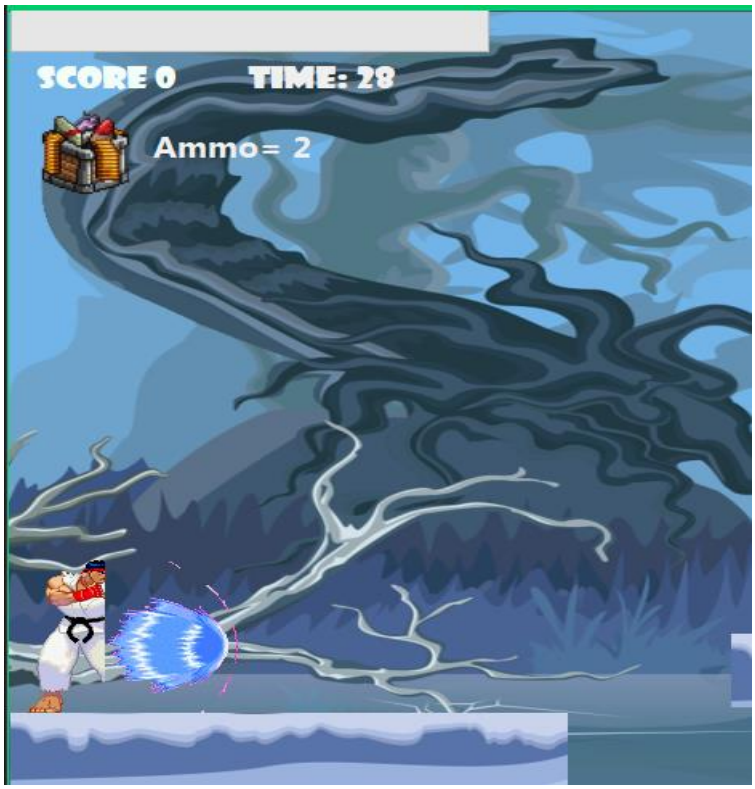
The test was successful in the sense that the fireball was now being fired across the game but I now have a bug that doesn't allow the player to shoot the fireball again after the first one. To fix this I will have to either remake the code or implement some lines of code to fix this issue.

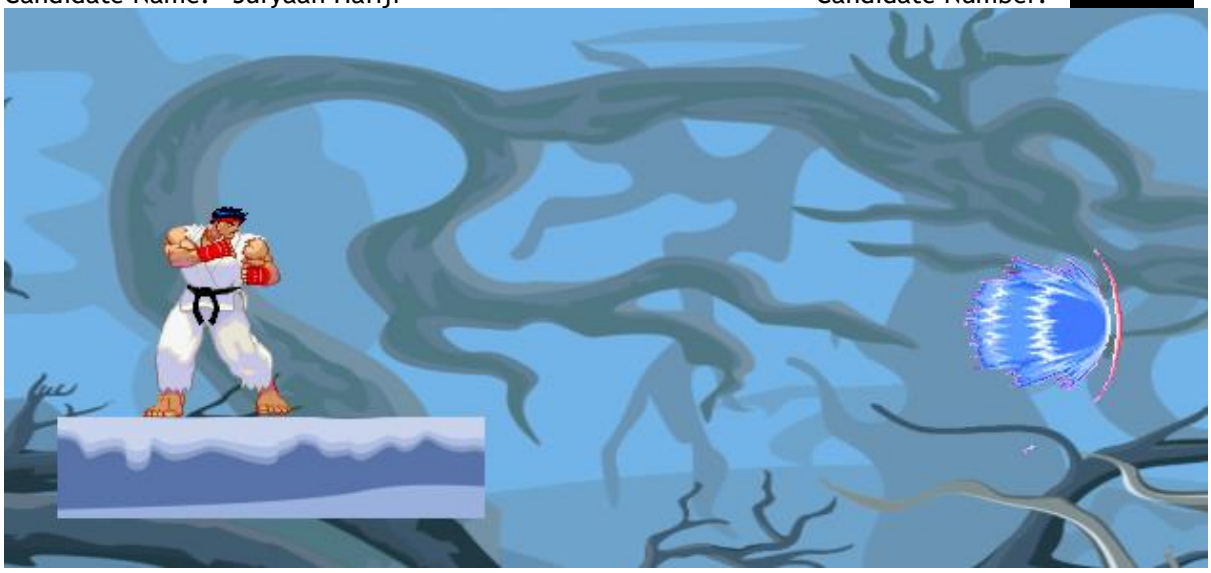
SOLUTION TO THE BUG

As the issue isnt with the fireball firing I decided to check where the fireball was going and why it was not returning the player. This was due to the fact I was using a single fireball to shoot and it set the fireball to carry on going for infinite. To stop this from happening I wrote a line of code that creates a new location of the fireball to shoot at from so it keeps returning to the player after its shot.

```
if (e.KeyCode == Keys.Space)
{
    if (ammo > 0)
    {
        fireball.Visible = true;
        ammo--;
        fireballSpeed = -5;
        fireball.Location = new Point(Player.Location.X + 100, Player.Location.Y);
    }
}
```

TESTING THE SOLUTION TO THE BUG/ISSUE





This was a successful test as I shot the first fireball and it sent the fireball to the right of the screen but then I reposisitoned myself up on a higher platform and tried it again and it proved to have fixed the issue that was made in the code.

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Fireball being shot	Spacebar key	As my shooting button is the spacebar key	Fireball animation executes when the spacebar is pressed	NA
Ammo being removed	Spacebar key	As the player shoots the fireball 1 ammo should be removed	1 ammo is removed and displays it	NA
Shooting with no ammo	Spacebar key	When the ammo = 0 the fireball should not shoot	No fireball animation when the spacebar is pressed when ammo is 0	NA

REVIEW AND FEEDBACK (P4)

I have now completed the fourth prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.

A quick interview with REDACTED has been made (Available in Appendix R.4)

Summary of the conversation:

- Fireball animation is good
- Everything meets his standards except there is no animation for the player to shoot the fireball

Candidate Name: <Sufyaan Hafiji> Candidate Number: <[redacted]>
I did everything for the fireball correctly but REDACTED requested I made a animation that changes when I press spacebar so it looks more immersive.

MAKING THE CHANGES REDACTED REQUESTED

To make this more immersive I decided to change the gif animations of the player picturebox when the spacebar is pressed and when the spacebar is depressed it changes back to normal.

```
if (e.KeyCode == Keys.Space)
{
    shotFireball = true;
    Player.Width += 40;
    Player.Width -= 40;
    Player.Image = Properties.Resources.standing;
}
```

This is in the KeyUp function and the line of code I added is the player.image line which fetches the gif from resources from when I first added it and changes it to that.

```
//shoots a fireball
if (e.KeyCode == Keys.Space)
{
    if (ammo > 0)
    {
        fireball.Visible = true;
        ammo--;
        fireballSpeed = -5;
        fireball.Location = new Point(Player.Location.X + 100, Player.Location.Y);
        Player.Image = Properties.Resources.fireball1;
    }
}
```

I then added the same line but with the other gif of the fireball inside the keydown function.

TESTING THE SOLUTION(SUCCESS)

This solution was a success because it now added a animation when I pressed spacebar so it shoots and changes gifs.

REQUIREMENTS WORKED ON

Requirements	Justification	Met?	Justification
Animations	To not make the game looks boring I need to add animations when the player moves and shoots	Yes	I successfully met this requirement as I created a good fireball animation and changed the characters gifs according to keys pressed

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
NA	NA	NA	NA

PROTOTYPE 5 (ENEMIES)

ENEMY DAMAGE

I started off by creating a picturebox inside the gamescreen and I put into it a GIF of a plant enemy that attacks when the user is playing the game.

```
if (Player.Bounds.IntersectsWith(enemy.Bounds))
{
    playerHealth = playerHealth - 20;
}
```

After creating and setting up the enemy, I created a if statement inside the score timer function to check when the player bounds intersects with the enemy bounds and then removes 20 health points from the playerHealth integer variable and reduces the healthbar.

TESTING ENEMY DAMAGE (SUCCESS)

This test was a success as it reduces the players health by 20 every second the scoretimer increments it.

ENEMY DEATH

After successfully creating a enemy that damages the player, I decided to move onto the enemy dying when the fireball interacts with the player but i may also add a healthbar onto the enemy when i create my other levels.

I started off by creating a if statement inside the main timer function to check when the fireball bounds intersects with the enemy sprite and then if it does, it executes the code inside it that makes the sprite die and not visible to the player

```
//enemy death
if (fireball.Bounds.IntersectsWith(enemy.Bounds))
{
    enemy.Visible = false;
    enemy.Location = AmmoBox.Location;
}
```

As I had a problem with the ammo pickup last time, I decided to do the same thing and make the location of the enemy move the ammobox location so it doesn't keep damaging the player after the visibility is set to false.

TESTING THE ENEMY DEATH (SUCCESS)

This test was successfull as the enemy was removed when the players fireball comes into contact with the enemy sprite.

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Does enemy do damage	Player interacts with the enemy	To test if the enemy does damage I need to interact with it so I can see if it does damage	Enemy does damage when interacted with	NA

Does the enemy die	Fireball interacts with the enemy	To test if the enemy health is taken away when the fireball comes into contact with it	Enemy dies when the fireball interacts with it	NA
Enemy death	NA	The enemy should die and not be visible when they do die	The enemy is no longer visible when it dies	NA

REVIEW AND FEEDBACK (P5)

I have now completed the fifth prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.

A quick interview with REDACTED has been made (Available in Appendix R.5)

Summary of the conversation:

- **Enemy AI is good**
- **Only change needed is make the enemy do more damage to the player**

MAKING THE CHANGES REDACTED REQUESTED

As this is a quick and easy change there will not be much to change other than a number.

Before

```
//enemy damage
if (Player.Bounds.IntersectsWith(enemy.Bounds))
{
    playerHealth -= 20;
}
```

After

```
//enemy damage
if (Player.Bounds.IntersectsWith(enemy.Bounds))
{
    playerHealth -= 30;
}
```

TESTING THE SOLUTION (SUCCESS)

To test this I just tested what the damage was at 20 and the damage at 30.

Before

After

REQUIREMENTS WORKED ON

Requirements	Justification	Met?	Justification
NA	NA	NA	NA

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
20 player damage	**When the player interacts with either the enemies of the void they should either die instantly or slowly start to die**	Yes	I successfully met this success criteria as I created enemy damage and it was 20 for the stakeholder

PROTOTYPE 6 (SCROLLING BACKGROUND)

As my game is going to be 800 by 600, I need to have the background scroll with the player but also all the pictureboxes need to keep moving with the player when they are pressing the keys so it needs to give the illusion that everything is moving.

I started off creating a new function calling it MoveGameElements and the parameters was the string direction. Inside it I created a foreach loop that constantly executed the code inside it.

```
0 references
private void MoveGameElements (string direction)
{
    foreach (Control x in this.Controls)
    {

    }
}
```

This is a method that is called MoveGameElements so its easy for me to remember what it is. (String direction) is the parameters for the method as it takes the string of whats needed to move and direction is which way they will move

```
foreach (Control x in this.Controls)
{
    if (x is PictureBox && (string)x.Tag == "platform")
    {
        if (direction == "back")
        {
            x.Left -= backgroundSpeed;
        }
        if (direction == "foward")
        {
            x.Left += backgroundSpeed;
        }
    }
}
```

This creates a if statement and checks if the picturebox is a platform and inside it is more if statements which makes the direction equals backwards or forwards

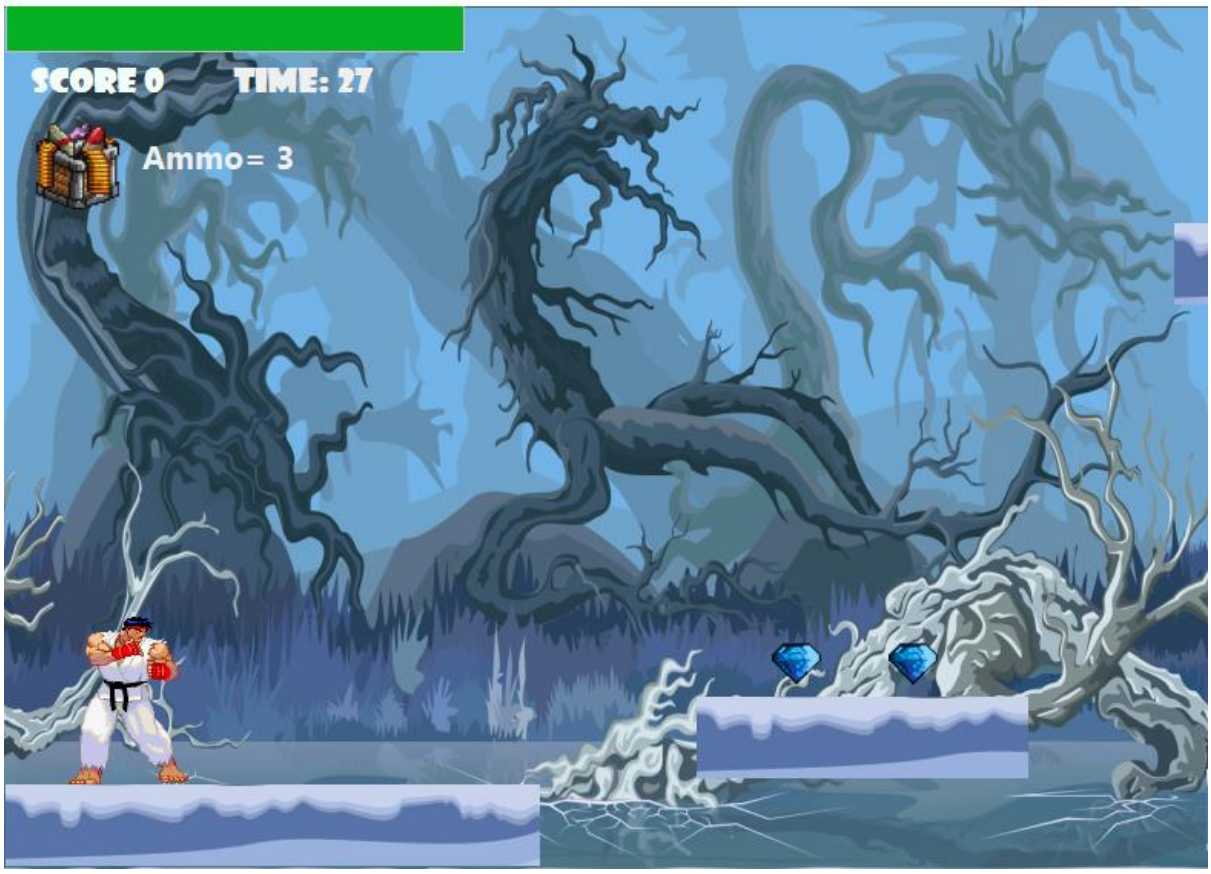
Makes the background speed a negative so it goes left and makes the background speed positive so it goes right

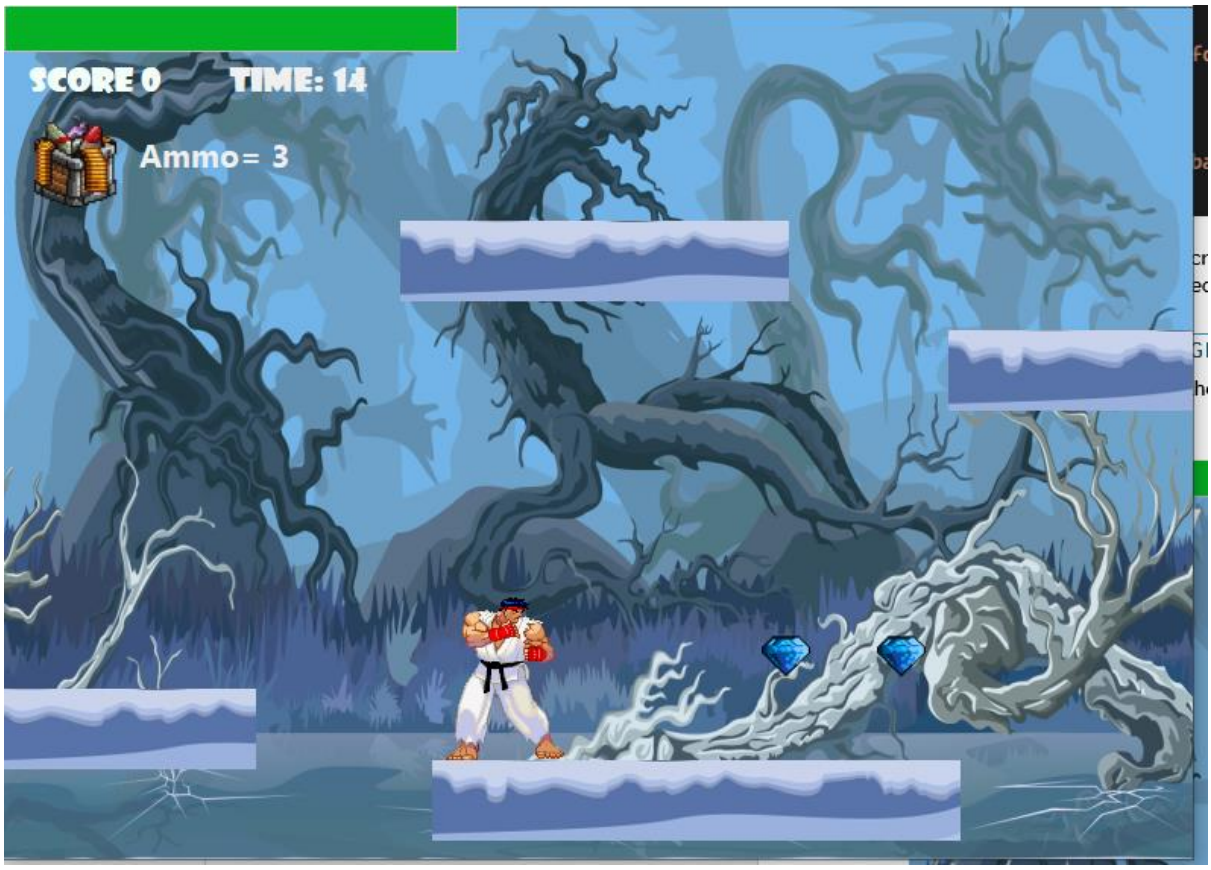
```
if (goRight == true)
{
    MoveGameElements("forward");
}
if (goLeft == true)
{
    MoveGameElements("back");
}
```

Inside the main timer event I created if statements which check if the Boolean variables goRight and goLeft are true, it will execute the MoveGameElements forward or backward if statements.

TESTING SCROLLING BACKGROUND (SUCCESS+BUG/ISSUE)

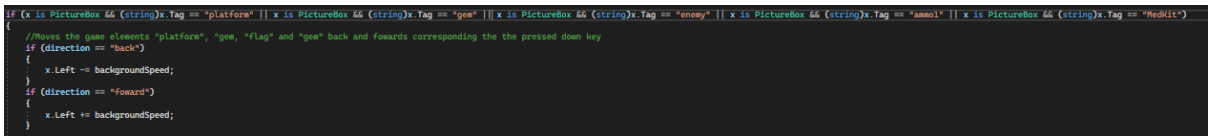
To test this effectively, I set the size of the window to 800 by 600 to make sure that it is moving the background.





This was a successful test as the code gave the illusion that the background was moving but only the platform would move with the player so I will need to alter the code I have written or write code that will fix it.

SOLUTION TO THE BUG/ISSUE

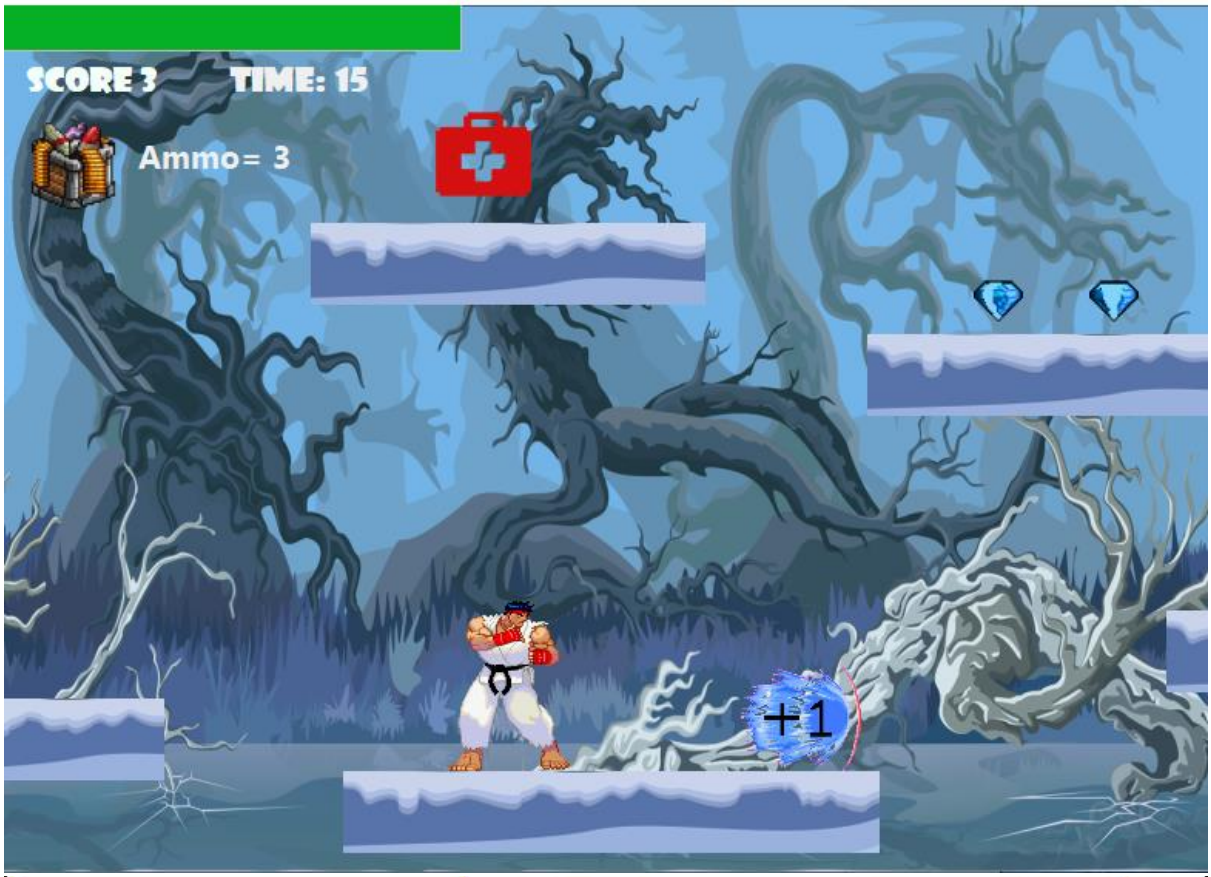


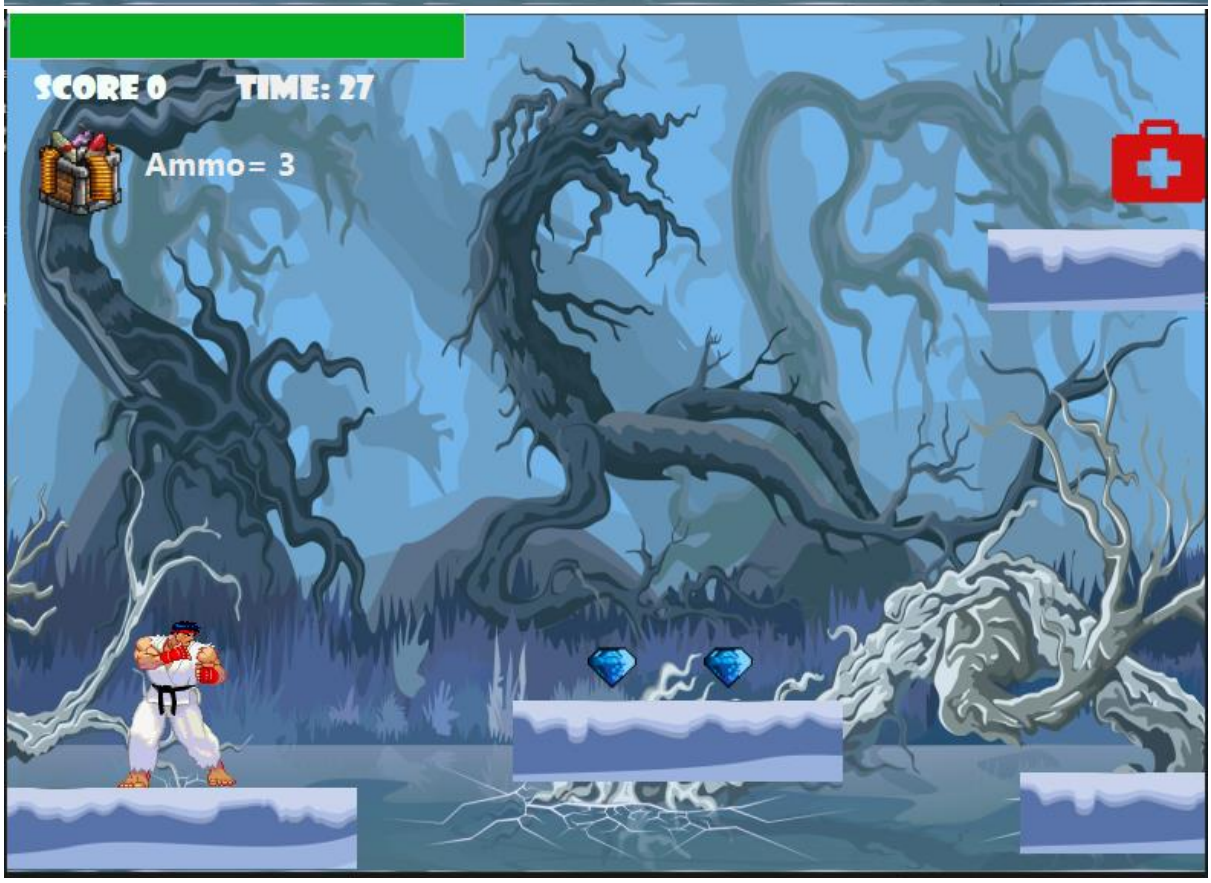
```
if (x is PictureBox && (string)x.Tag == "platform" || x is PictureBox && (string)x.Tag == "gem" || x is PictureBox && (string)x.Tag == "enemy" || x is PictureBox && (string)x.Tag == "Ammo1" || x is PictureBox && (string)x.Tag == "MedKit")
{
    //Moves the game elements "platform", "gem, "flag" and "gem" back and fowards corresponding the the pressed down key
    if (direction == "back")
    {
        x.Left -= backgroundSpeed;
    }
    if (direction == "foward")
    {
        x.Left += backgroundSpeed;
    }
```

I created if statements inside the code but to be more code efficient I decided to separate the if statements with I I which indicates that it's a new if statement. I wrote down every picturebox in the game that needed to be moved with the tag of it so it moves with the player.

TESTING THE SOLUTION (SUCCESS)

The new code fixed the game and the test was successful as everything in the game that needed to give the illusion that the background is moving.


SCORE 3
TIME: 15
Ammo= 3
+1


SCORE 0
TIME: 27
Ammo= 3

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Background moving right	Right arrow key	To test if the background moves with the character	Background moves right with the player when the right arrow key is pressed	NA
Background moving left	Left arrow key	To test if the background moves with the character	Background moves left with the player when the left arrow key is pressed	NA

REVIEW AND FEEDBACK (P6)

I have now completed the sixth prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.

A quick interview with REDACTED has been made (Available in Appendix R.6)

Summary of the conversation:

- Scrolling background was good
- It was immersive and didn't need change

As REDACTED liked the scrolling background I do not need to make any changes so I will leave it as it is.

REQUIREMENTS WORKED ON

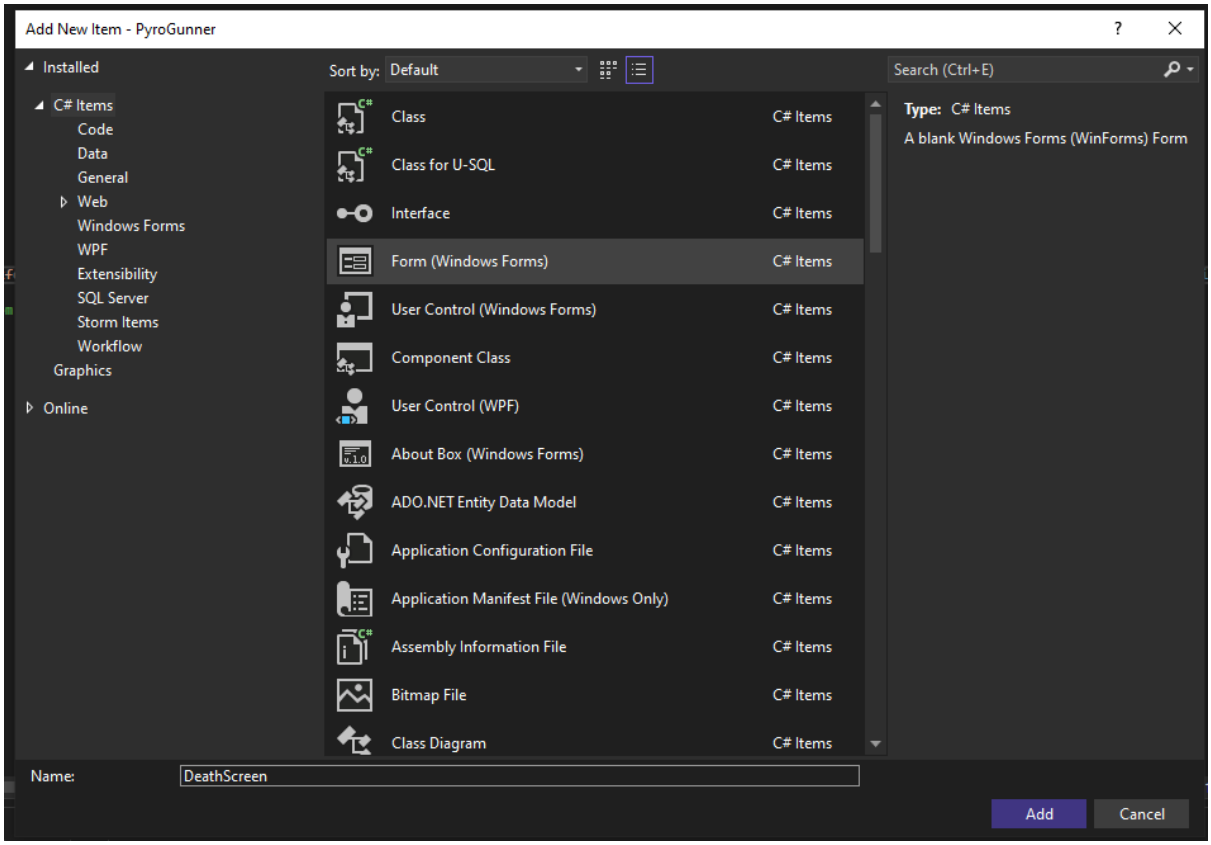
Requirements	Justification	Met?	Justification
Background must be scrolling	To make the game more immersive the background must not be static but it should move with the character	Yes	I successfully met this success criteria as I made the game have a scrolling background just as requrested by REDACTED
Style of the game	REDACTED requested that the game not be too realistic in design and should have a cartoonish style to it	Yes	I met this as I made it a cartoon style game with the characters platforms gems and enemies

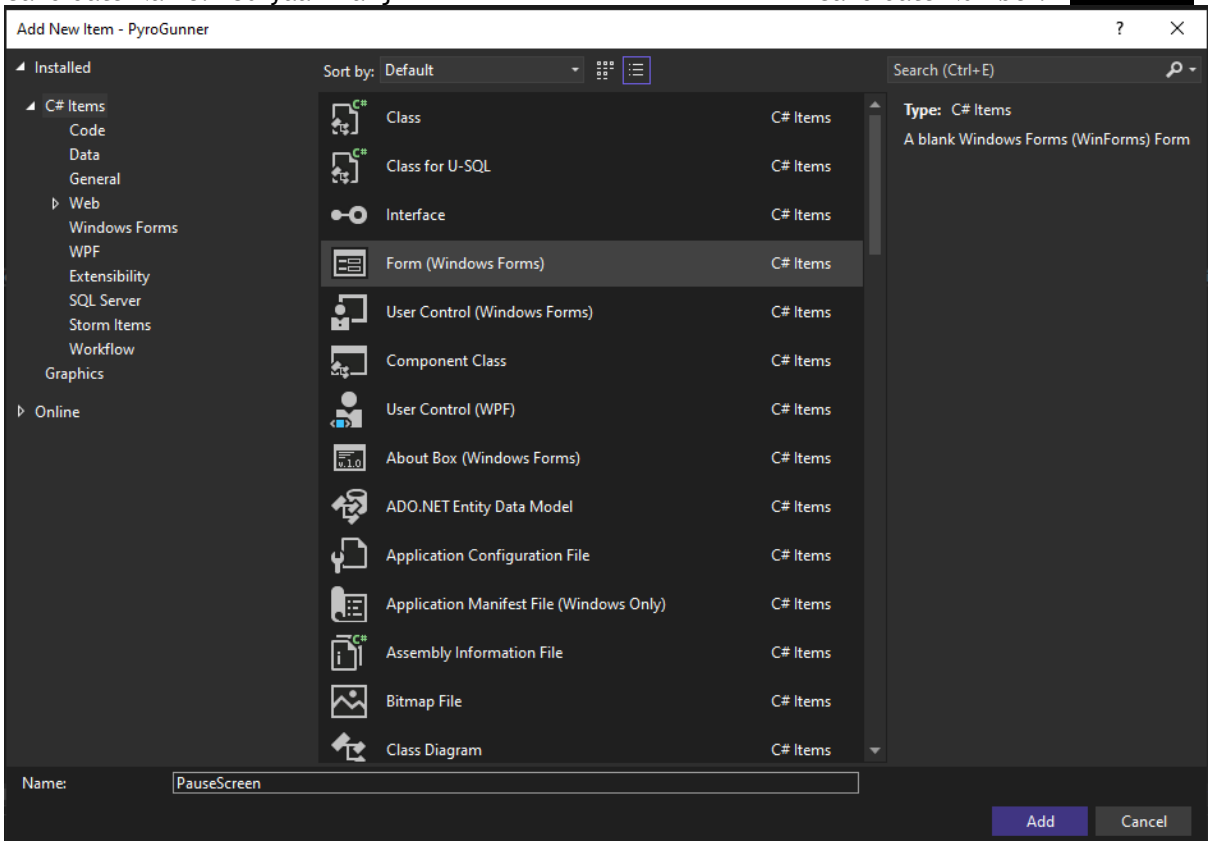
SUCCESS CRITERIA WORKED ON

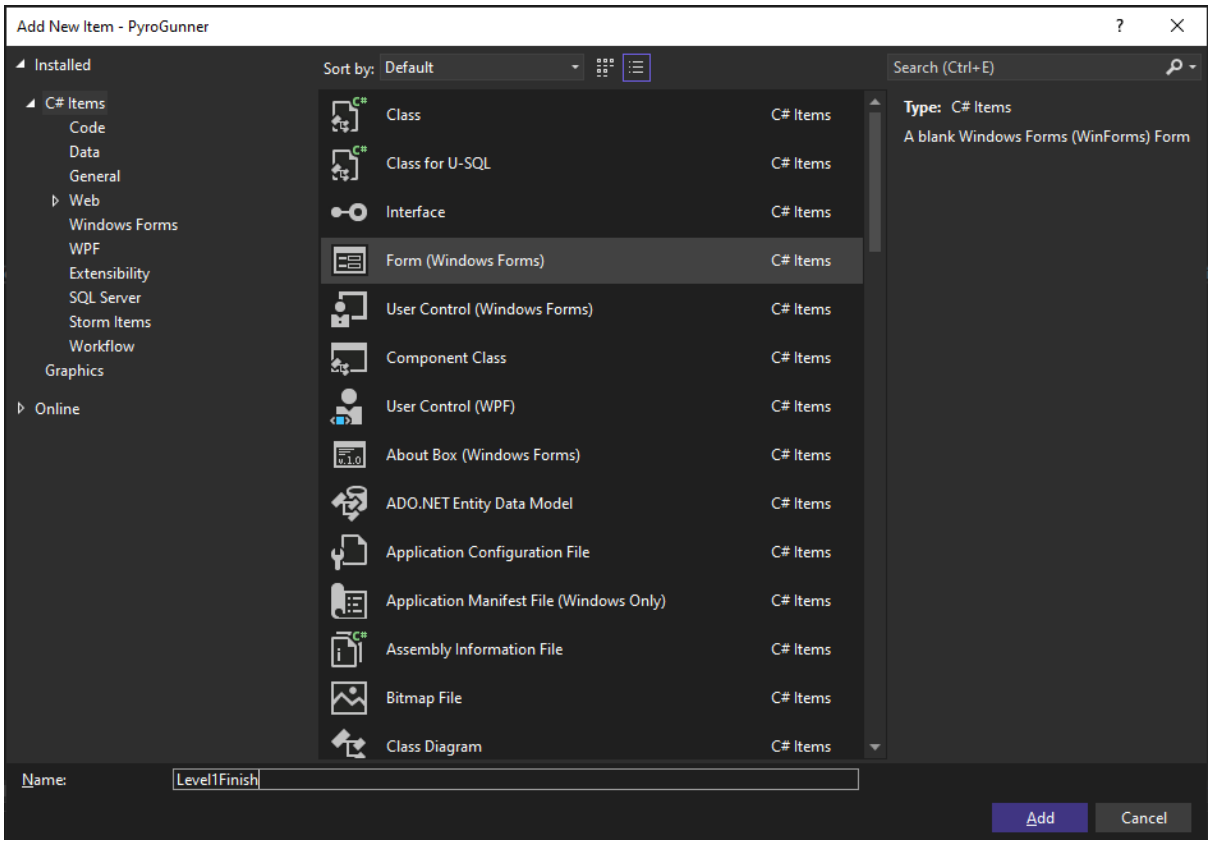
Success criteria	Justification	Met?	Justification
Different backgrounds	**Game must have a variety of background so it looks more professional and immerisve**	Yes	I used a variety of backgrounds for the game forms and menus

PROTOTYPE 7 (PAUSE+DEATH SCREEN+LEVEL FINISH SCREEN)

I started off with creating the forms by selecting windows forms and naming them appropriately so that I know what to write in the code when im changing things inside it.







DEATH SCREEN

I started off with setting the background to game over as it's closest to my game design sketch in the design section of the NEA. This indicates to the user that they have failed and I will also be adding a level restart button and level selector button.

I then wrote code inside the button functions of the game which directs them to whichever the button goes to.

```
private void L1Restart_Click(object sender, EventArgs e)
{
    //restarts the level
    GameScreen gameWindow = new GameScreen();
    gameWindow.Show();
    this.Hide();
    GameScreen.score = 0;
    GameScreen.scoretimer = 30;
}
```

This selects the name of the form I want to show and it creates a new instance of it. It then hides the current window and shows the window that has been called

Sets the integer variables in the gamescreen window score to 0 and the scoretimer to 30

```
private void L1Select_Click(object sender, EventArgs e)
{
    //goes to level selector
    LevelSelecter gameWindow = new LevelSelecter();
    gameWindow.Show();
    this.Hide();
}
```

This button is for the level selector and it brings up the level selector window, hides the game over screen and shows it.

```
private void ExitButton_Click(object sender, EventArgs e)
{
    //exits to main menu
    StartScreen gameWindow = new StartScreen();
    gameWindow.Show();
    this.Hide();
}
```

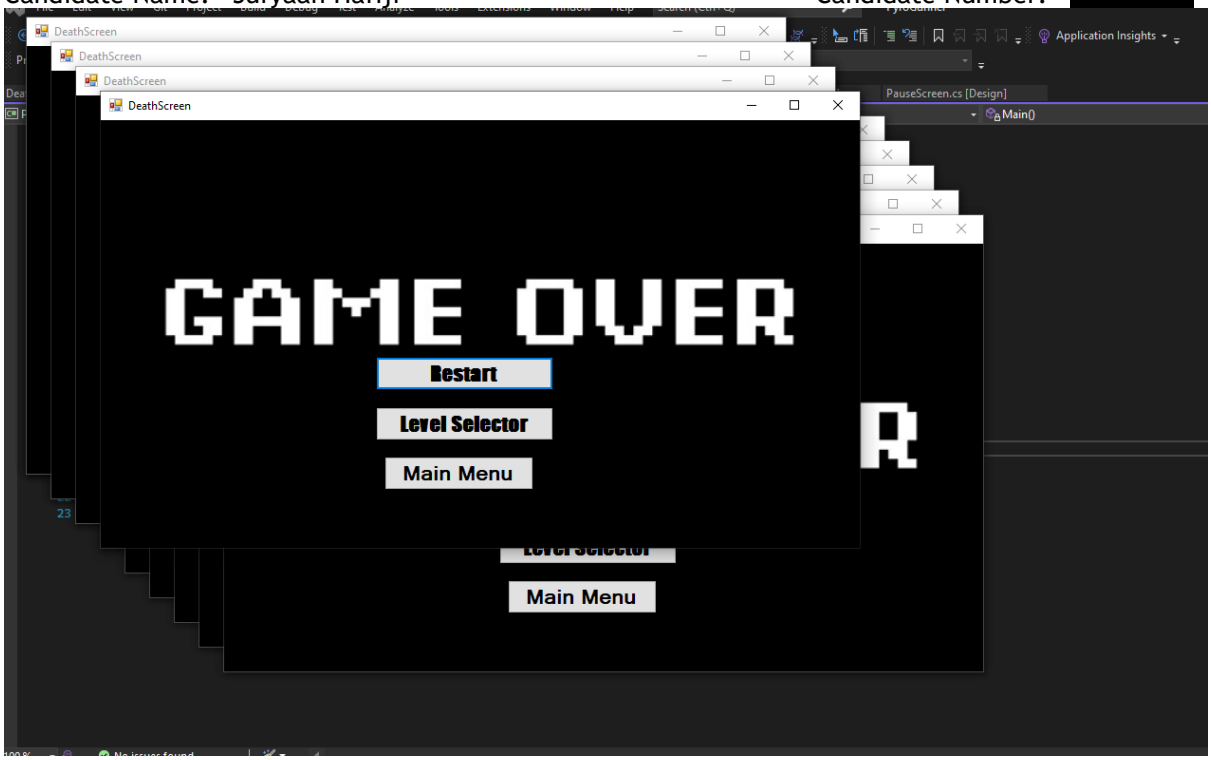
This button sends the player back to the start screen and hides the window and shows the other one.

After this I decided to go into the code and add code that brings the death screen up which is when the player healths reaches 0 or when the score timer equals to 0.

```
//player death
if (playerHealth < 1)
{
    MainTimer.Stop();
    DeathScreen gameWindo = new DeathScreen();
    gameWindo.Show();
    this.Hide();
}

//timer runs out death
if (score < 0)
{
    MainTimer.Stop();
    DeathScreen gameWindo = new DeathScreen();
    gameWindo.Show();
    this.Hide();
}
```

TESTING THE DEATH SCREEN (BUG/ISSUE) (WILL COME BACK TO LATER)

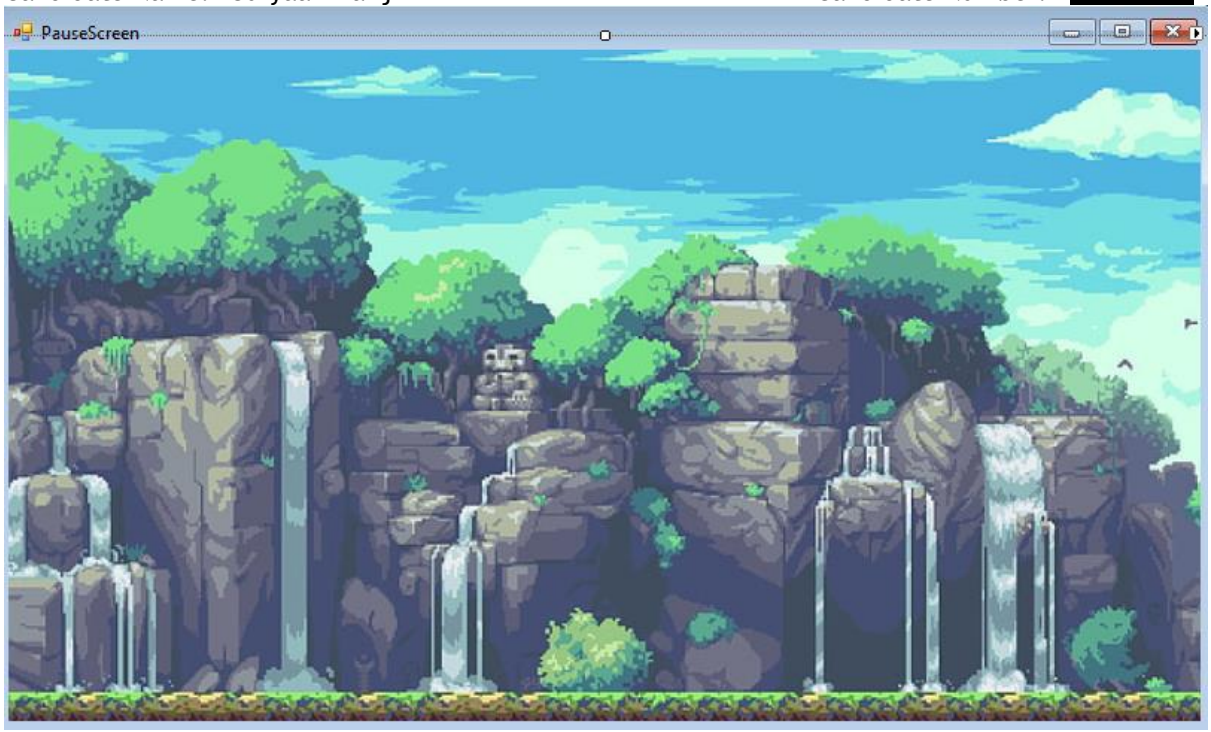
The death screen successfully loaded but it loaded a lot of forms so this is a very big bug as its not playable because the user will have to keep closing the forms.



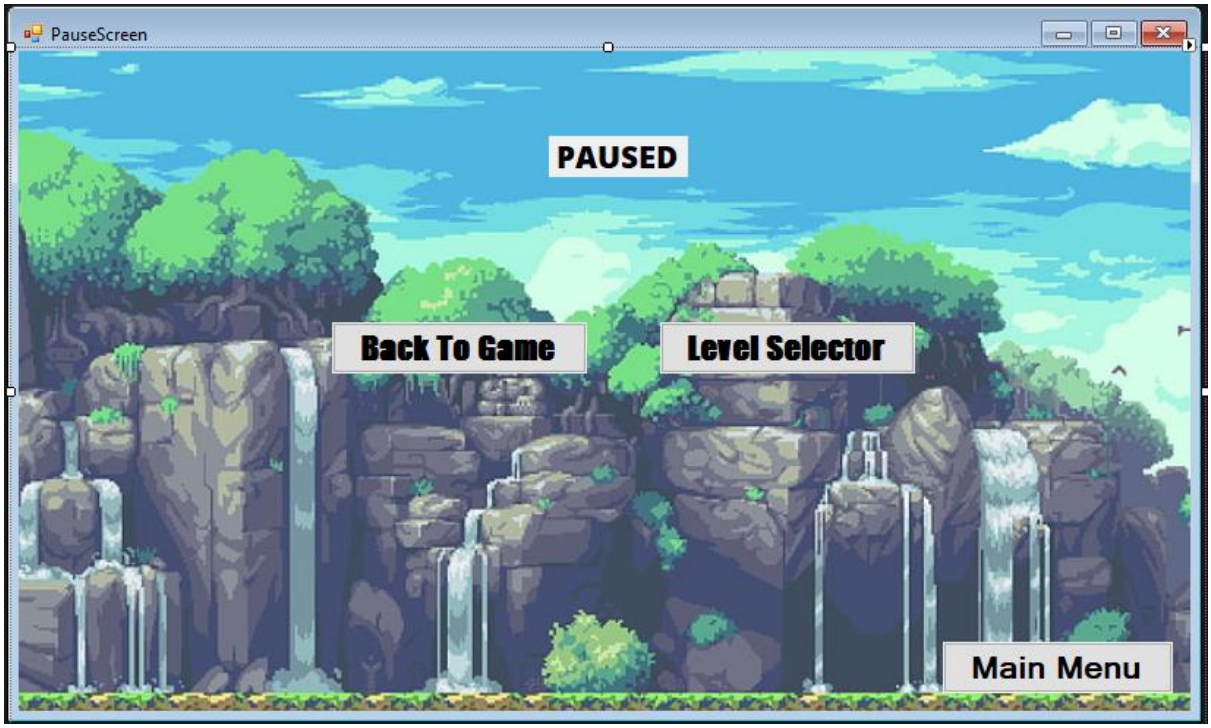
I checked all my timer intervals and they were all correct and I also checked when the if statements were executing. I do not see what the issue/error is so I will be coming back to this after ive finished a lot more of the game.

PAUSE SCREEN

As the stakeholder has requested, I will be making a pause feature in the game so the player can pause the game and come back to it when they want. To do this I will have to create a new form and design it for the requirements and then add a feature that when they press the “P” character the timers will stop and then they can go back to it whenever they want without losing any progress.



This design fits the requirement of having a cartoon theme to the game and not too realistic.



After this I created a textbox to show that the game is paused and then 3 button which are the back to game and the level selector but also the main menu where it goes back to the start.

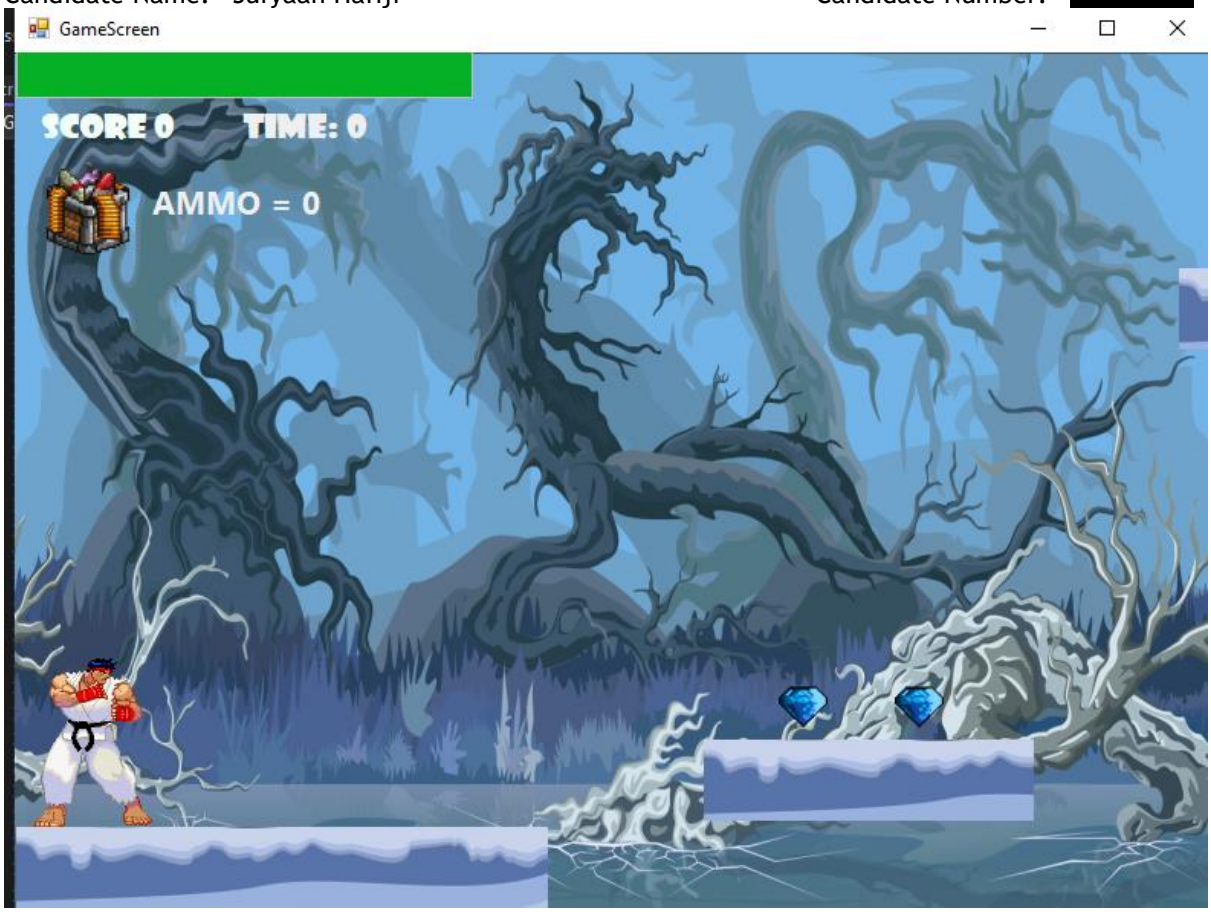
```
//Sends the player back to the game
1 reference
private void BTG_Click(object sender, EventArgs e)
{
    this.Hide();
}
//Send the player to level selector screen
1 reference
private void L1Select_Click(object sender, EventArgs e)
{
    LevelSelector gameWindow = new LevelSelector();
    gameWindow.Show();
    this.Hide();
}
//Sends the player to the main menu
1 reference
private void ExitButton_Click(object sender, EventArgs e)
{
    StartScreen gameWindow = new StartScreen();
    gameWindow.Show();
    this.Hide();
}
```

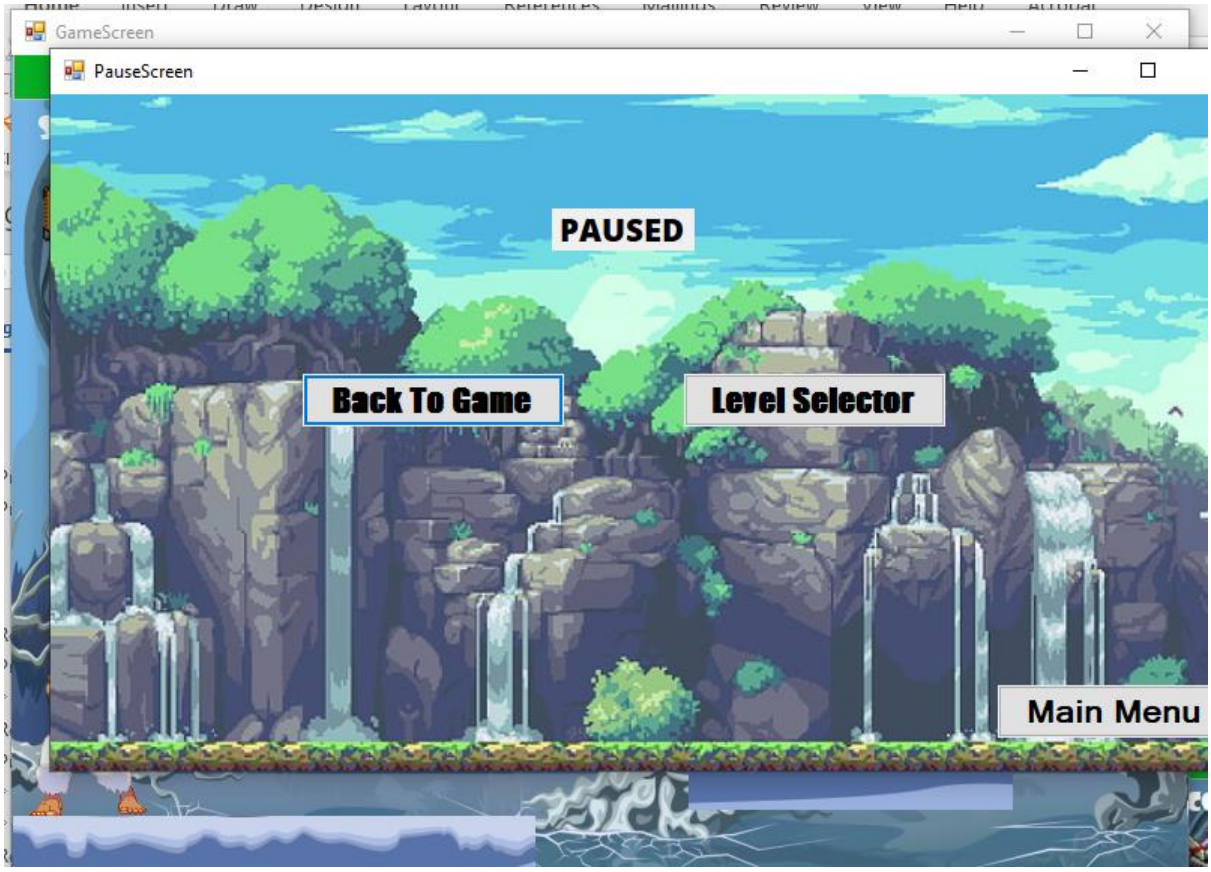
These are all the same concepts from previous form navigation I have created so I don't need to explain everything as its already been explained a few times.

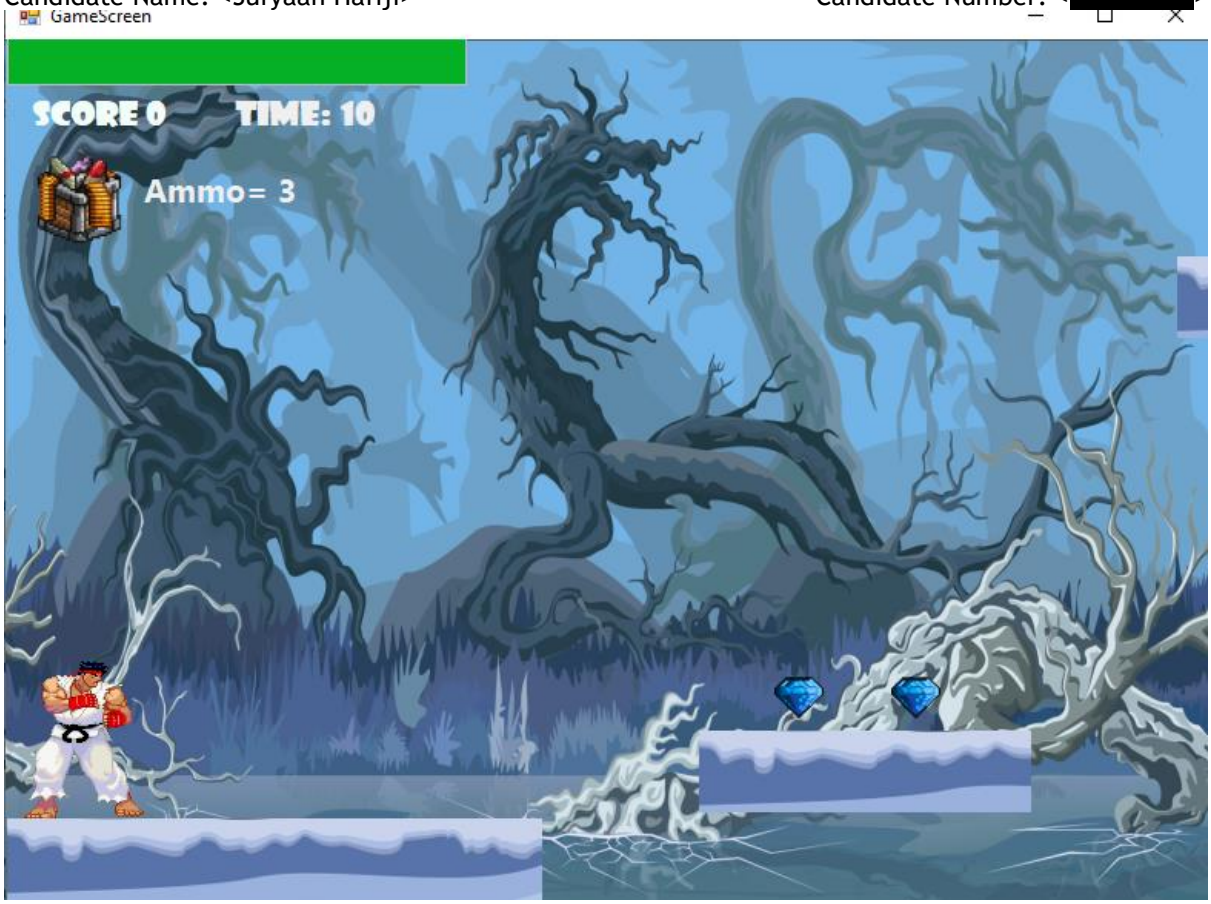
```
//pauses the game
if (e.KeyCode == Keys.P)
{
    PauseScreen gamewindow= new PauseScreen();
    gamewindow.Show();
}
```

This is inside the keydown function that check when the character P is pressed and opens up the pause screen and shows it and stops the timer.

TESTING THE PAUSE SCREEN (SUCCESS)


GameScreen
SCORE 0
TIME: 0
AMMO = 0


GameScreen
PauseScreen
PAUSED
Back To Game
Level Selector
Main Menu

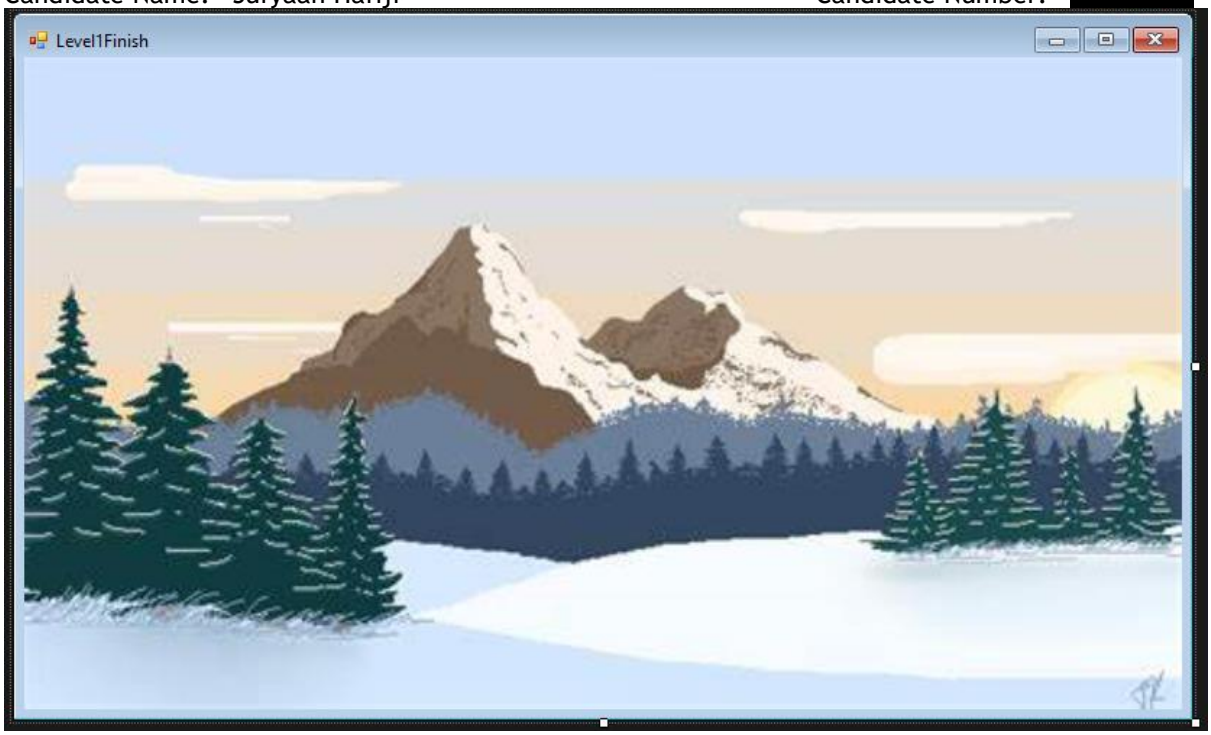


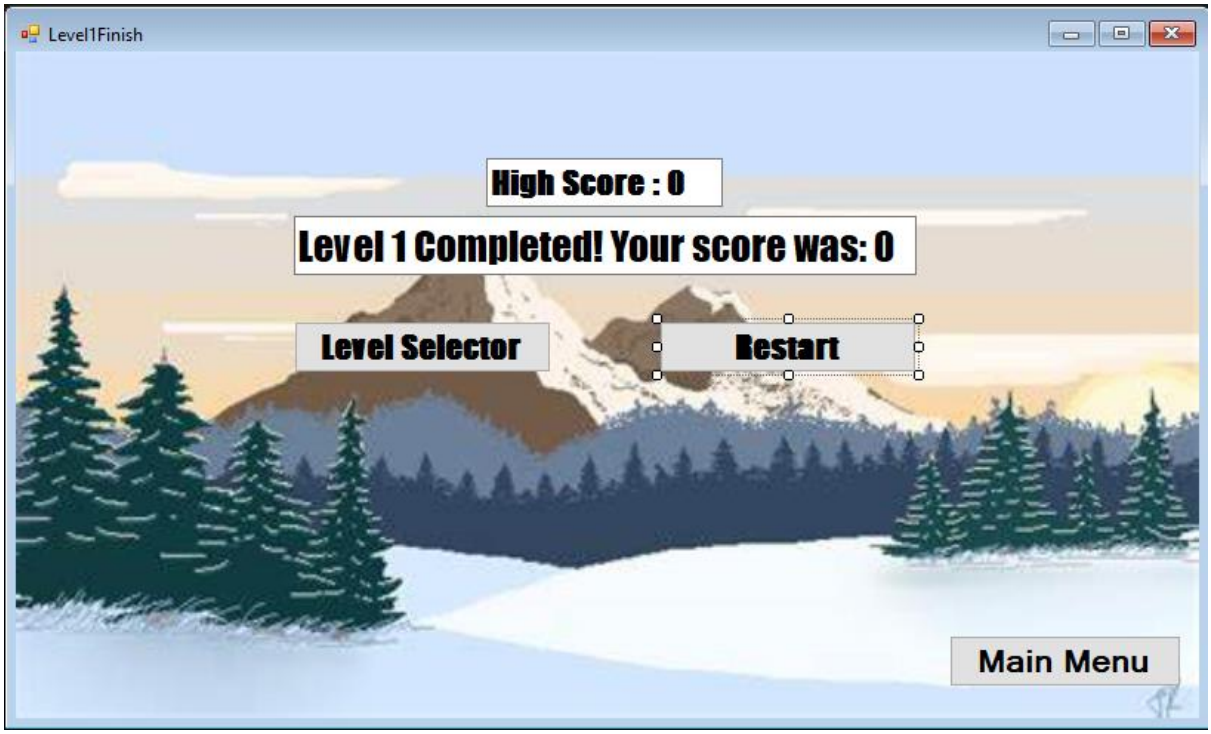
This was a successful test as it paused the game and stopped everything and carried on when I click the back to game button.

LEVEL FINISH SCREEN

This is the screen that will show up when the player interacts with the finishing flag and shows their score and the buttons to restart the game or go to level selector or the main menu.

I stared off making a new form and setting background to a cartoon pixel theme.





The score will be displayed on the screen by using the integer variable score. The level selector, main menu and restart will do what I have been creating in the other forms. The high score will compare the score to the high score and then display it as that.

```
public static int score;
public static int highscore;
```

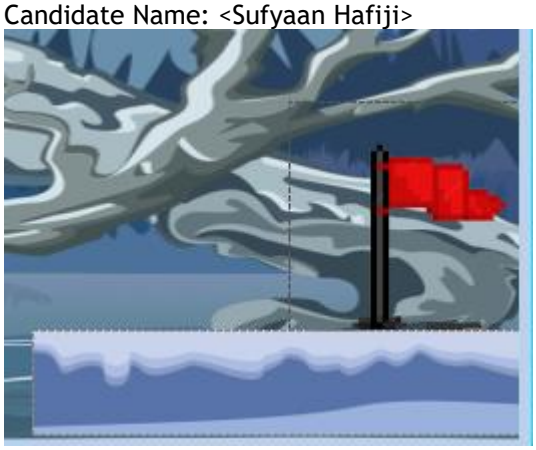
I started off creating a public static integer variable so it can be accessed from any other form which will be needed when im doing it in my level finish.

```
//goes back to level selector
1 reference
private void L1Select_Click(object sender, EventArgs e)
{
    LevelSelector gameWindow = new LevelSelector();
    gameWindow.Show();
    this.Hide();
}

//restarts the level
1 reference
private void L1Restart_Click(object sender, EventArgs e)
{
    GameScreen gameWindow = new GameScreen();
    gameWindow.Show();
    this.Hide();
    GameScreen.score = 0;
    GameScreen.scoretimer = 30;
}

//goes back to the main menu
1 reference
private void ExitButton_Click(object sender, EventArgs e)
{
    StartScreen gameWindow = new StartScreen();
    gameWindow.Show();
    this.Hide();
}
```

This code screenshot is just the window switch and show and hide like from before.

```
//Displays the score and highscore
1 reference
private void Level1Finish_Load(object sender, EventArgs e)
{
    GameScreen.score = GameScreen.score * GameScreen.scoretimer;
    EndScore.Text = "Level Completed! Your score was: " + GameScreen.score;
    if (GameScreen.score > GameScreen.highscore)
    {
        GameScreen.highscore = GameScreen.score;
        HighScore.Text = "High Score: " + GameScreen.highscore;
    }
    else
    {
        HighScore.Text = "High Score :" + GameScreen.highscore;
    }
}
```

This code screenshot fetches the integer variable score from gamescreen and times it to the scoretimer which is then put onto the end score text. For the highscore if the gamescreen score is more than the highescore currently, it will replace it and if it isnt it will just display the highscore as normal.

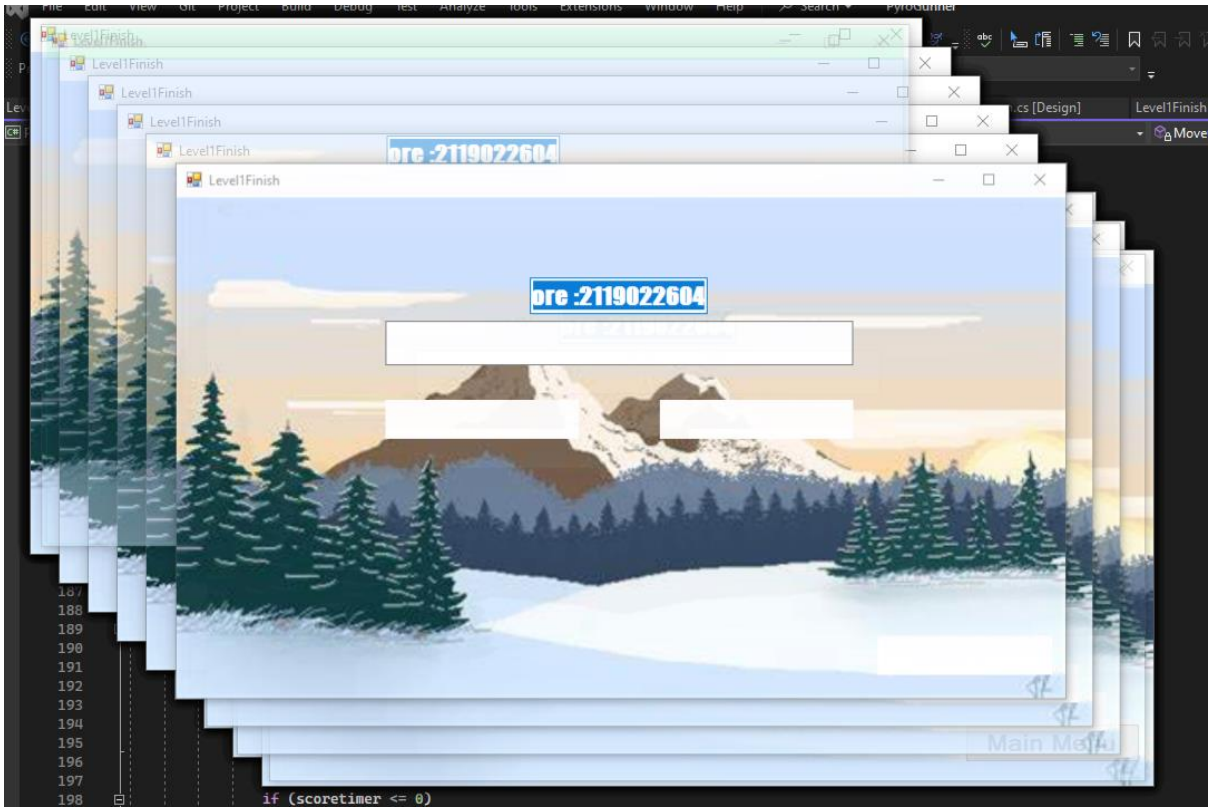


To reach the finish screen I added a flag which is on a if statement to check whether they have interacted with the flag and then execute the code inside it.

```
if (Player.Bounds.IntersectsWith(flag.Bounds))
{
    Level1Finish gameWindow = new Level1Finish();
    this.Hide();
    gameWindow.Show();
}
```

TESTING THE FINISH SCREEN (BUG/ISSUE)

After creating all the code and creating the picturebox I decided to test the program but it did the same thing as the death screen where a lot of forms would just open up and not just one



This is now a serious issue as 2 features of my game requested by my stakeholder do not work as intended so i need to figure out why this is happening even though my code works fine

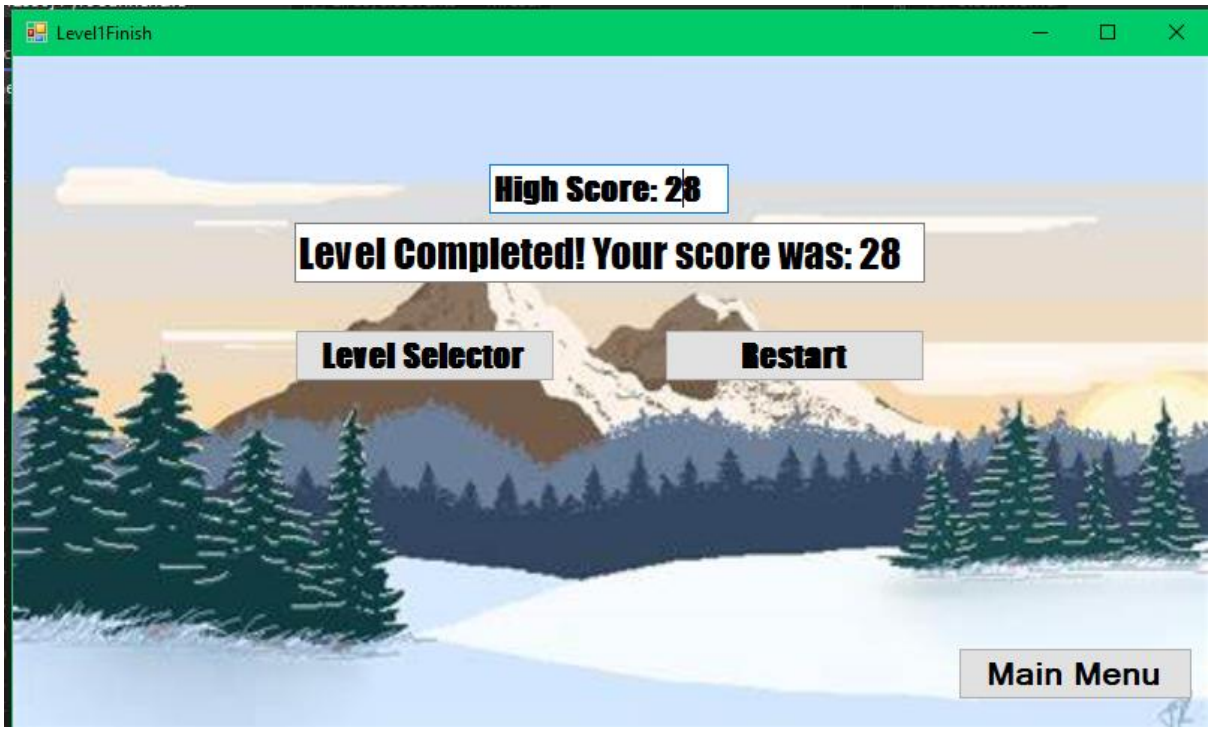
SOLUTION TO THE BUG/ISSUE + DEATH SCREEN

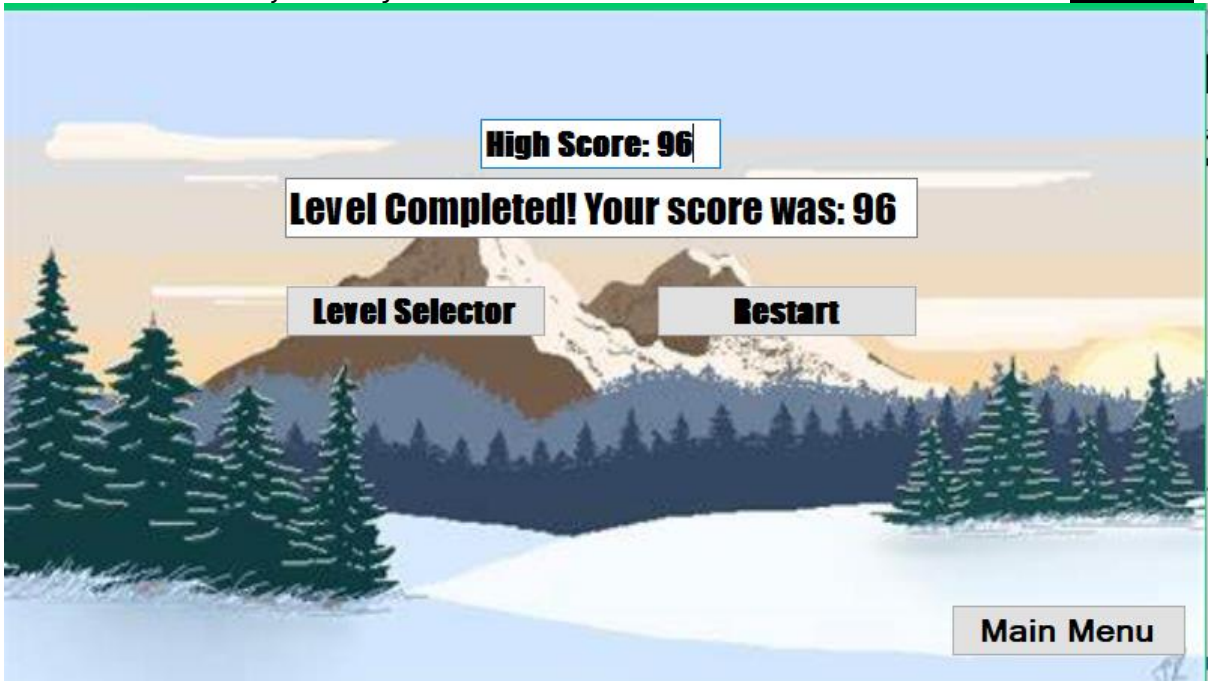
After thoroughly going through the code, I confirmed that none of my code was incorrect but the placement of where it was caused the issue. As the game would keep looping and bringing up the forms, I decided to check where it was placed and it was inside the foreach loop from before. Moving the code to a different area should fix this as its no longer in a loop for it to be causing this issue.

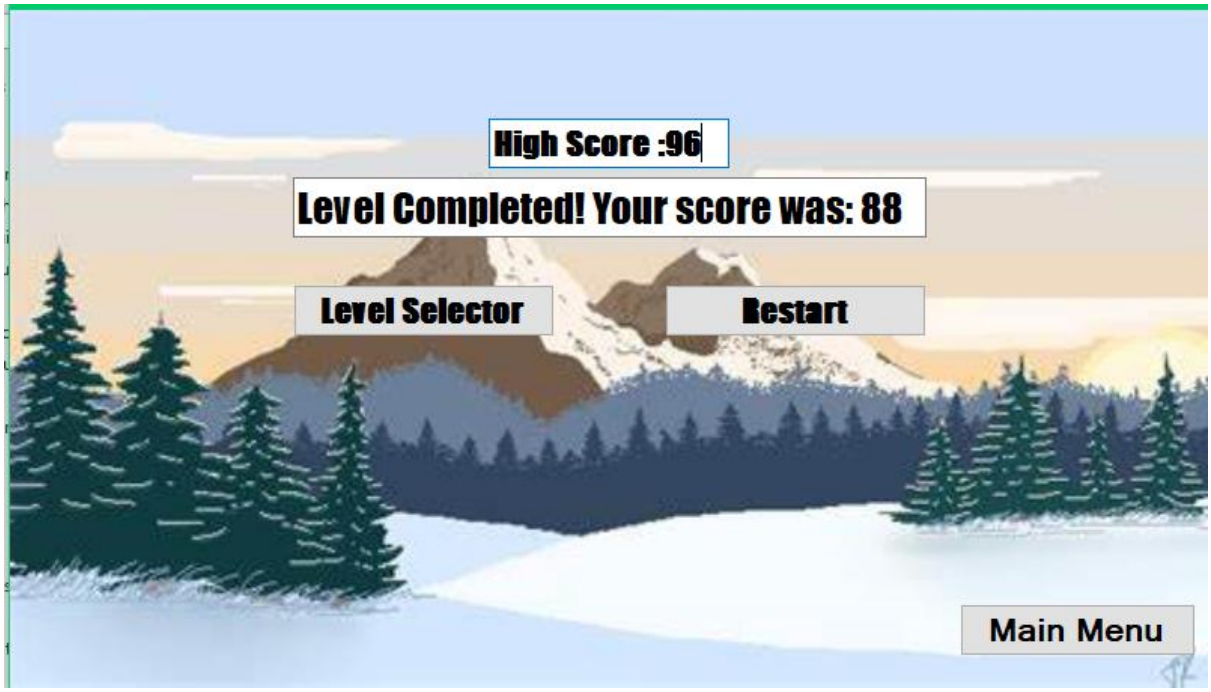
```
if (Player.Bounds.IntersectsWith(flag.Bounds))
{
    Level1Finish gameWindow = new Level1Finish();
    this.Hide();
    gameWindow.Show();
    MainTimer.Stop();
}
//player death
if (playerHealth < 1)
{
    MainTimer.Stop();
    DeathScreen gameWindo = new DeathScreen();
    gameWindo.Show();
    this.Hide();
}
```

TESTING THE SOLUTION(SUCCESS)

This was a successful test as it now only showed 1 form and it also displayed the right scores. This issue also helped me fixed the death screen and it now only shows on of them




High Score: 96
Level Completed! Your score was: 96
Level Selector
Restart
Main Menu


High Score :96
Level Completed! Your score was: 88
Level Selector
Restart
Main Menu

TEST AND JUSTIFICATION

What is being tested	Input	Justification of input	Outcome	How to solve
Pause screen	Character P is pressed	As said in the design, I would be assigning the P character to pause screen	Game pauses and brings up the pause screen	NA
Death screen	Character interacts with enemy and reaches 0 health	The death screen should pop up when the players health reaches 0	Death screen bought up when health = 0	NA
Score timer running out	NA	Score timer running out to 0 should bring up the game over screen	Game over screen is bought up when timer = 0	NA
Level finish screen	Player interacts with finishing flag	To finish the level the player has to interact with the flag	Level finish screen is bought up when the player interacts with the flag	NA

REVIEW AND FEEDBACK (P7)

I have now completed the seventh prototype so REDACTED will be reviewing the game see if he wants any changes made to game game.

A quick interview with REDACTED has been made (Available in Appendix R.7)

Summary of the conversation:

- All the forms look good
- They do as intended

- Immersive and meets his success criteria/requirements
- He wishes to have a textbox saying how long it took when he finishes the level

Overall REDACTED was happy with my interrupt screens and met his success criteria/requirements and the only change he wants made is having his time being displayed in the level finish.

MAKING THE CHANGES REDACTED REQUESTED

I started off with making another textbox inside the level 1 finish form.


```
int finishtime;
1 reference
public Level1Finish()
{
    InitializeComponent();
    GameScreen.scoretimer = finishtime;
}
```

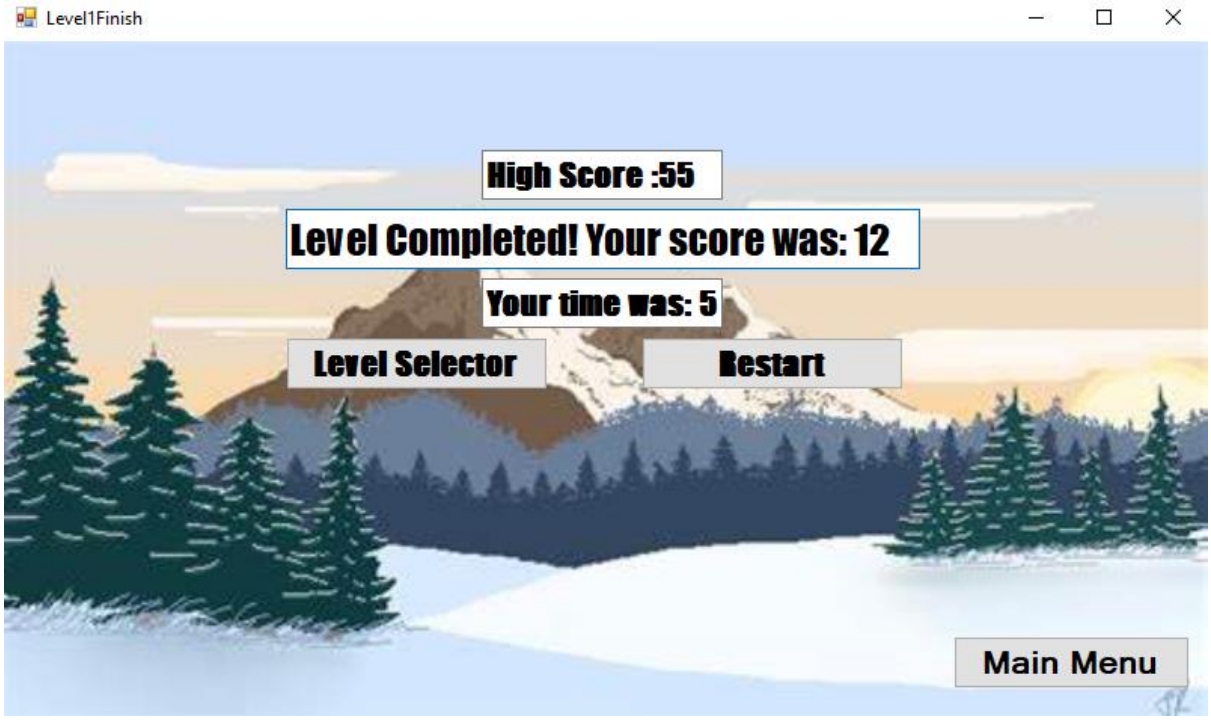
After that I created a new integer variable inside the level 1 finish form and set the gamescreen score timer variable as the same.

```
Finishtimer.Text = "Your time was: " + finishtime;
```

I then added this onto the textbox and it should now meet REDACTEDs requirements

TESTING THE SOLUTION

This test successful as it allowed the player to see what their time was with the score and high score. This has met the requirement that REDACTED requested with the feedback



REQUIREMENTS WORKED ON

Requirements	Justification	Met?	Justification
Game over screen	When the user fails to complete the level in time or dies due to a enemy or the void a death screen should appear	Yes	When the user fails to complete the level in time or dies due to a enemy or the void a death screen should appear
Levels	REDACTED mentioned that he wanted a variety of difficulty and a way to achieve that was by adding levels and making them harder as you progress	Partially met	I created levels for the game and was intended to make it increase in diffuculy but due to time constraints I cannot do it

SUCCESS CRITERIA WORKED ON

Success criteria	Justification	Met?	Justification
Different menus	**The menu will be simple and not too complicated. There may only be 2-3 options on the menu so its easy to navigate and easy to start the game.**	Yes	I met this because I added multiple menus in my game with a variety of navigation options

Score displayer	**There will be a score bar that will display the amount of gems collected, the timer and the total score.**	Yes	I displayed the score in game and in the level completed screen
Timer	**He has requested a timer to add intensity to the game**	Yes	I added a timer in the game and I added it onto the level completed screen
3 Levels	**The game will have 5 levels starting from easy then progressing harder and harder every level.**	Partially met	I created levels for the game and was intended to make it increase in diffuculy but due to time constraints I cannot do it
High score system	**The score system will be calculated by how many gems you have collected times the time left on the timer when you finish. For example 30 gems and 20 seconds left (30x20=60 score)**	Yes	I created this in the level completed form and it successfully worked

D. EVALUATION

POST DEVELOPMENT TESTING: INTEGRATION TESTING

MAIN MENU SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"LMC" on level selector button	Valid	Open level selector screen	Opens the selector screen	Fully met	
"LMC" on background	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on textbox	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on control button	Valid	Open up control screen	Opens up control screen	Fully met	
"LMC" on audio button	Valid	Turn off audio	Audio button does not work	Not met	

PAUSE MENU SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"LMC" on back to game button	Valid	Opens up game screen	Goes back to game screen	Fully met	
"LMC" on level selector button	Valid	Opens up level selector	Goes to level selector	Fully met	
"LMC" on background	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on main menu button	Valid	Opens up main menu	Goes back to main menu	Fully met	
"LMC" on textbox	Invalid	Nothing should happen	Lets you edit the textbox	Not met	

LEVEL SELECTOR SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"LMC" on level 1 button	Valid	Opens up level 1	Goes to level 1	Fully met	
"LMC" on level 2 button	Valid	Opens up level 2	Does not open up level 2	Not met	

"LMC" on level 3 button	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on level 4 button	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on level 5 button	Invalid	Nothing should happen	Nothing happens	Fully met	

CONTROLS SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"LMC" on menu button	Valid	Opens up menu screen	Goes to main menu	Fully met	
"LMC" on image	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on textbox	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on background	Invalid	Nothing should happen	Nothing happens	Fully met	

PLAYER CONTROLS

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"A" key is pressed	Valid	Moves the player left	Moves the player left	Fully met	
"S" key is pressed	Invalid	Nothing should happen	Nothing happens	Fully met	
"W" key is pressed	Valid	Makes the player jump	Player jumps when pressed	Fully met	
"Spacebar" key is pressed	Invalid	Should not move the character	Shoots fireball	Fully met	
"LMC" on player	Invalid	Nothing should happen	Nothing happens	Fully met	

SHOOTING

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
"LMC" on player	Invalid	Nothing should happen	Nothing happens	Fully met	
"P" key is pressed	Invalid	Player should not shoot	Pause screen is opened	Fully met	

"Spacebar" key is pressed	Valid	Player should shoot	Player shoots fireball	Fully met	
Ammo = 0	Invalid	Player should not shoot	Player does not shoot	Fully met	
Ammo is > 0	Valid	Player should shoot	Player can shoot	Fully met	

HEALTH REMOVAL/DEATH

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
Player shoots fireball at enemy	Invalid	Health should not be removed	Health is not removed	Fully met	
Player interacts with enemy	Valid	Health is taken away from player	Health is taken away	Fully met	
"Spacebar" key is pressed	Invalid	Nothing should happen	Nothing happens	Fully met	
"LMC" on healthbar	Invalid	Nothing should happen	Nothing happens	Fully met	
Health = 0	Valid	Player dies	Player dies	Fully met	

GAME OVER SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
Health = 0	Valid	Player dies	Game over screen	Fully met	
Score = 0	Invalid	Game should not end	Game over screen does not show	Fully met	
Time = 0	Valid	Game over	Gamer over screen shows	Fully met	
"P" key is pressed	Invalid	Nothing should happen	Nothing happens	Fully met	
"Spacebar" key is pressed	Invalid	Nothing should happen	Nothing happens	Fully met	

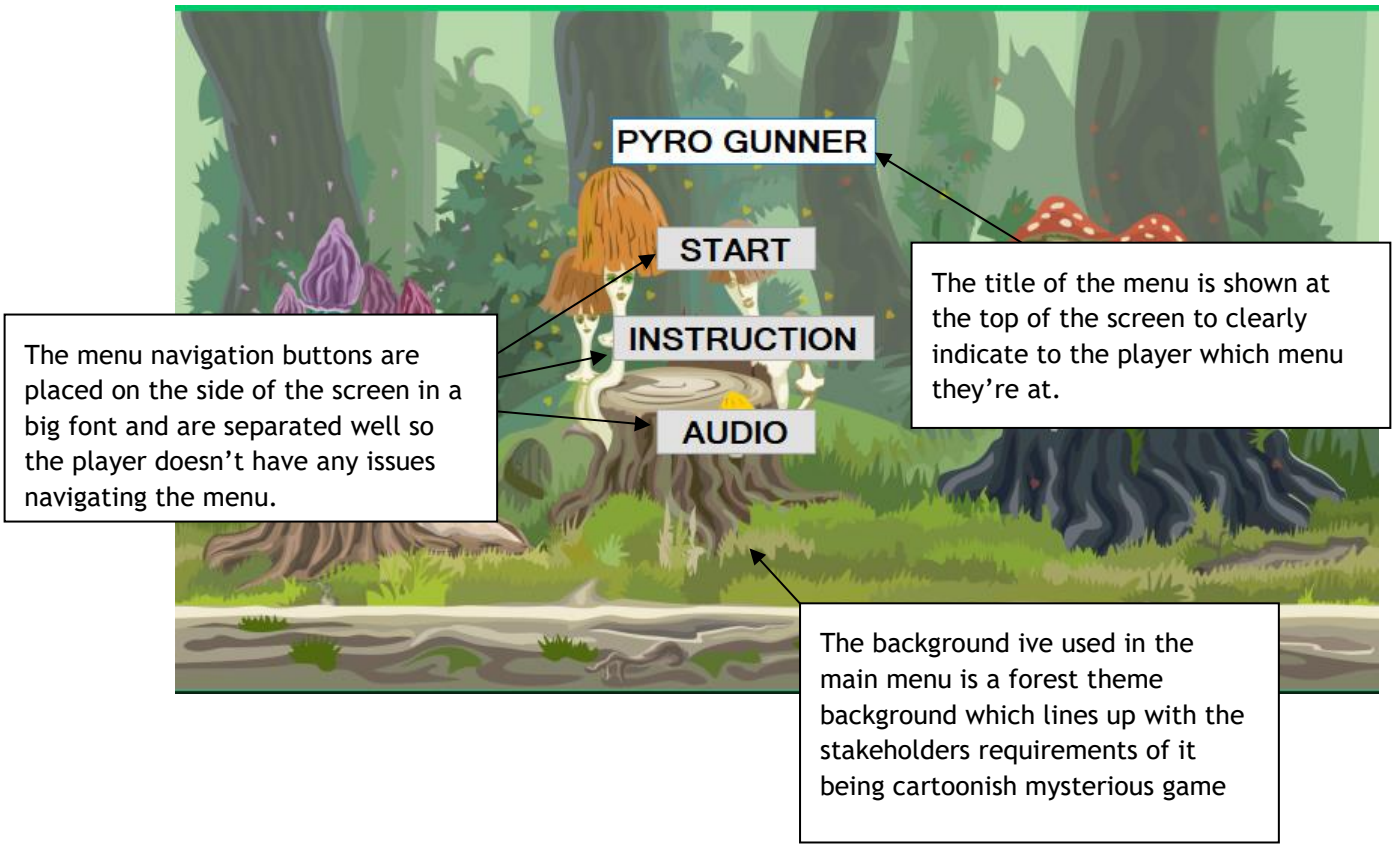
FINISH SCREEN

Test	Type	Expected Outcome	Actual outcome	Met?	Test evidence
Player interacts with flag	Valid	Finish screen opens	Finish screen opens	Fully met	
Score = 0	Invalid	Nothing should happen	Nothing happens	Fully met	

Time = 0	Invalid	Nothing should happen	Nothing happens	Fully met	
Health = 0	Invalid	Nothing should happen	Nothing happens	Fully met	
"P" key is pressed	Invalid	Nothing should happen	Nothing happens	Fully met	

USABILITY FEATURES

MAIN MENU

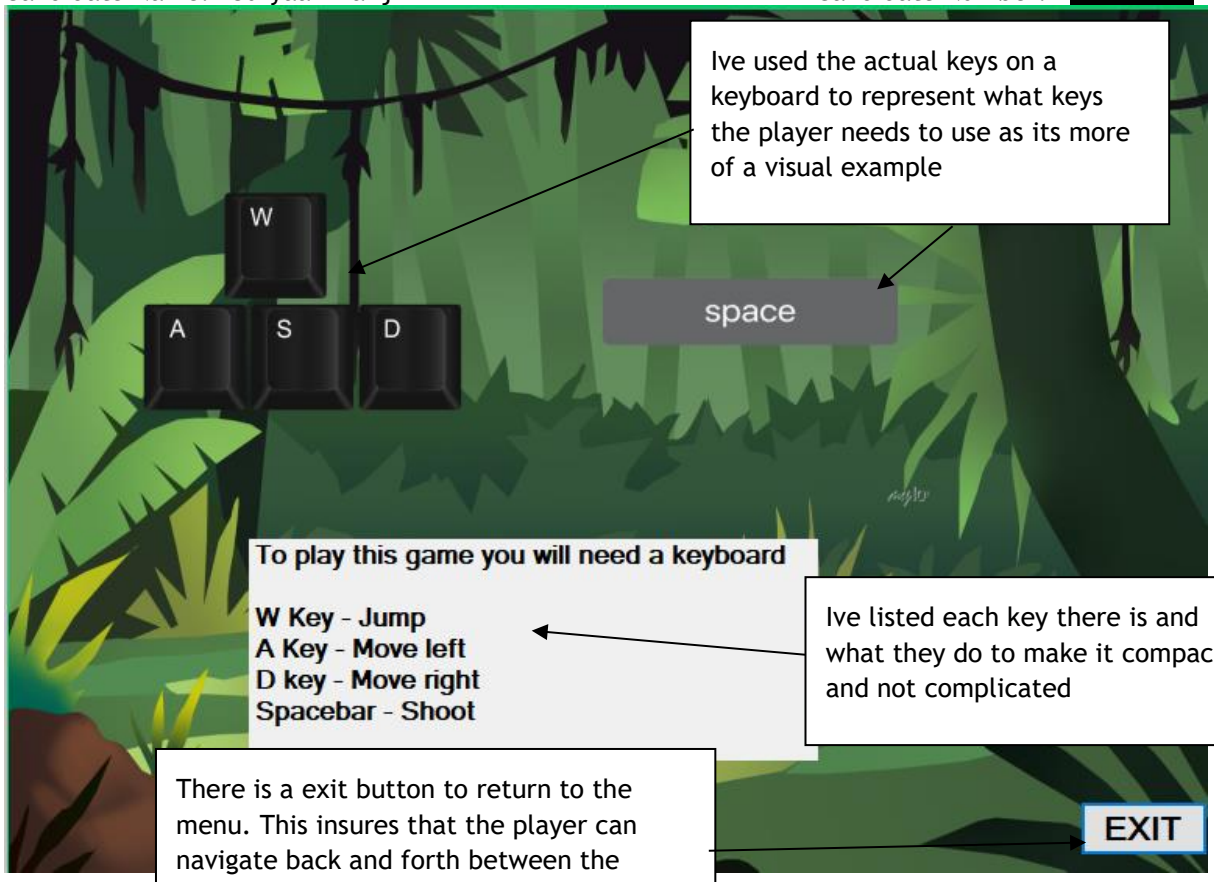


INTERVIEWING REDACTED FOR FEEDBACK

REDACTED: *I like the position of all the buttons and the background has met my request of having a cartoon theme to it. One thing you could change is the back fill of the textbox so it doesn't look a bit more professional*

Me: Im glad you like the design and style but due to the software im using, I cannot change the backfill of the textbox due to it not having any option to make it transparent

CONTROLS MENU



Ive used the actual keys on a keyboard to represent what keys the player needs to use as its more of a visual example

Ive listed each key there is and what they do to make it compact and not complicated

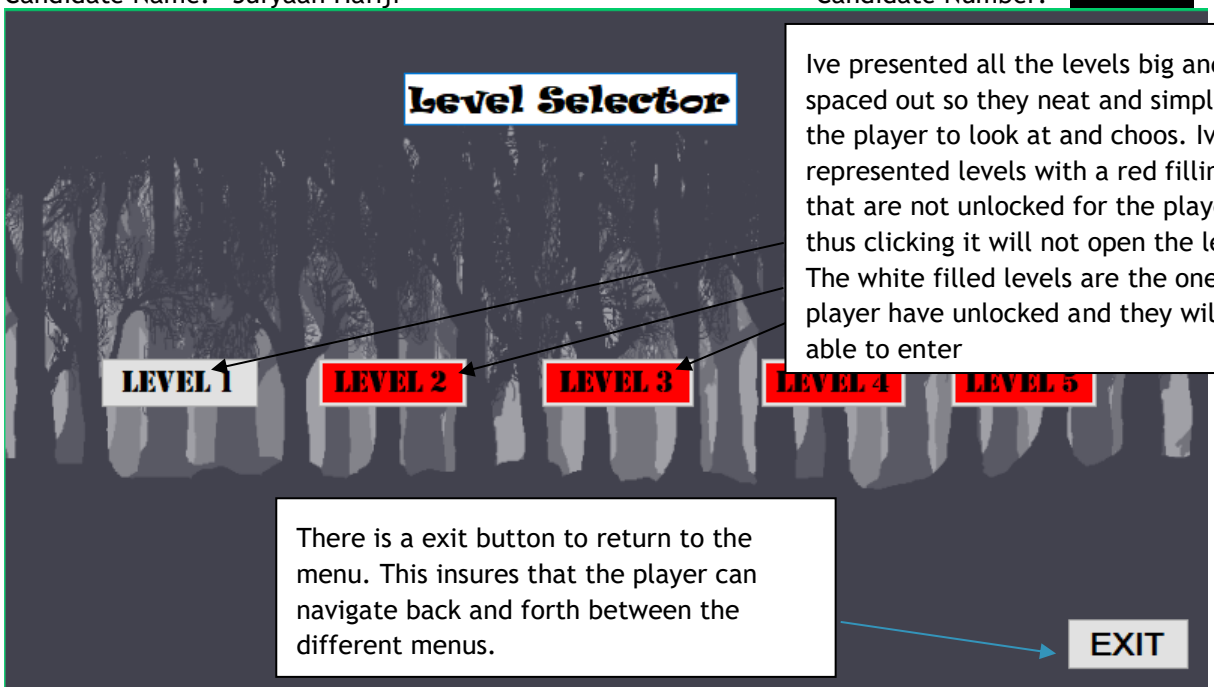
There is a exit button to return to the menu. This insures that the player can navigate back and forth between the different menus.

INTERVIEWING REDACTED FOR FEEDBACK

REDACTED: *The controls menu is easy to read and to figure out what the keys do but one thing that's incorrect about it is that its WASD instead of arrow keys.*

Me: Im glad its easy to read and I forgot to change the image and explanation so I will change this as soon as I can

LEVEL SELECTOR MENU



Ive presented all the levels big and spaced out so they neat and simple for the player to look at and choos. Ive represented levels with a red filling that are not unlocked for the player thus clicking it will not open the level. The white filled levels are the ones the player have unlocked and they will be able to enter

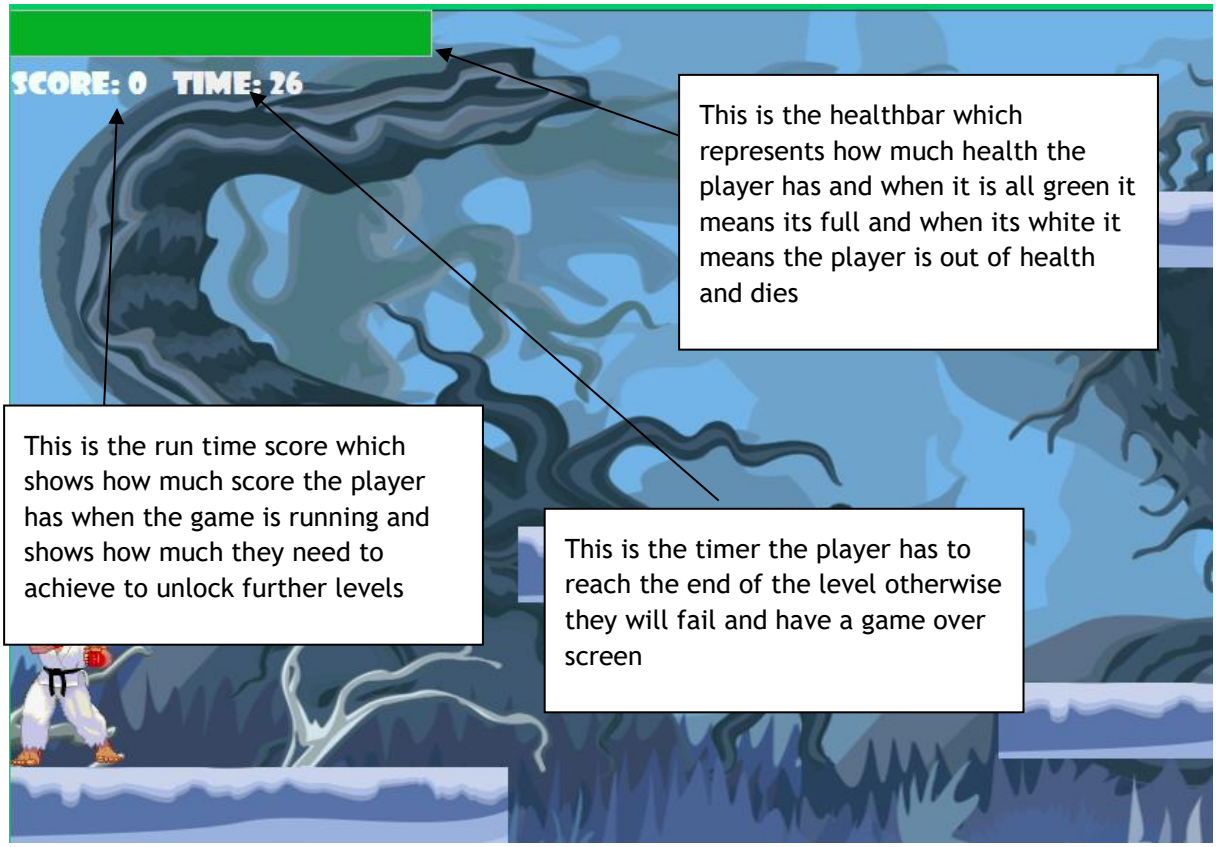
There is a exit button to return to the menu. This insures that the player can navigate back and forth between the different menus.

INTERVIEWING REDACTED FOR FEEDBACK

REDACTED: *I really like the background for this image and the red backfill for the buttons that are not able to be used is a really nice touch to show which are unlocked and which are not. I have nothing that I can say about it to change it*

Me: I did this to show a level of immersiveness and I appreciate you liked it

GAMEPLAY SCREEN



This is the healthbar which represents how much health the player has and when it is all green it means its full and when its white it means the player is out of health and dies

This is the run time score which shows how much score the player has when the game is running and shows how much they need to achieve to unlock further levels

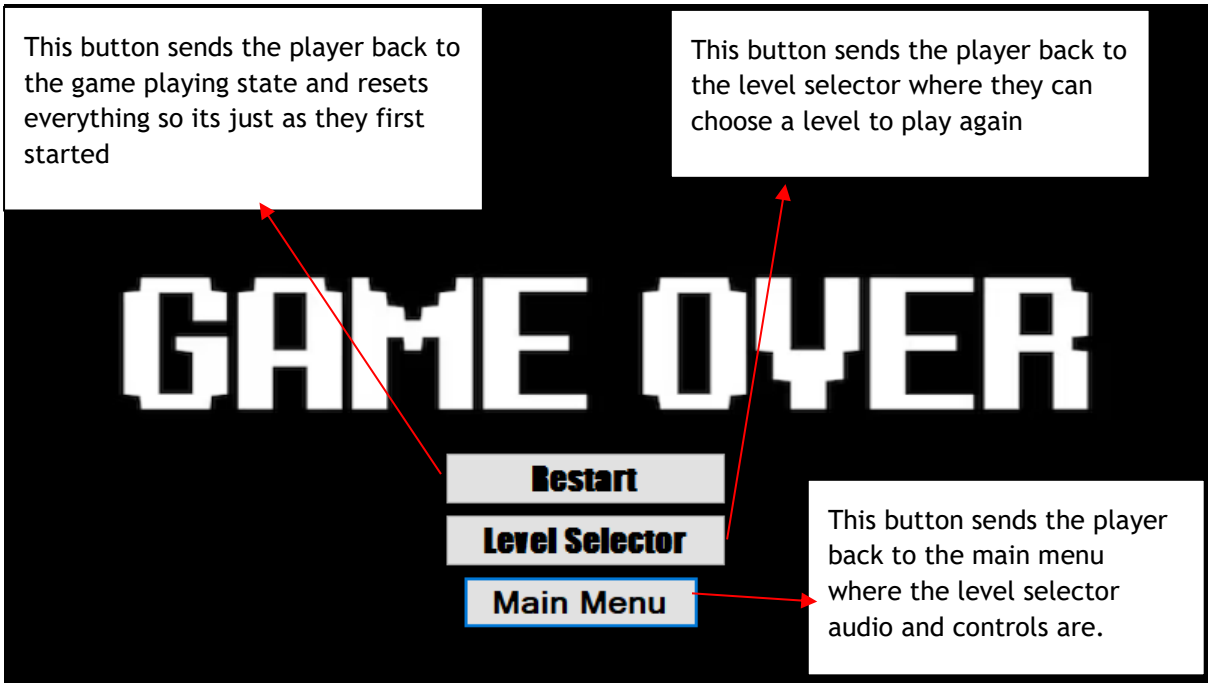
This is the timer the player has to reach the end of the level otherwise they will fail and have a game over screen

INTERVIEWING REDACTED FOR FEEDBACK

REDACTED: *This screen features is good overall as its simple but not boring and I can clearly tell things apart and things aren't cluttered and hard to read. I would add a ammo box counter just to add a bit more immersiveness but overall its good.*

Me: I will take that into consideration and update it as soon as possible

DEATH SCREEN MENU

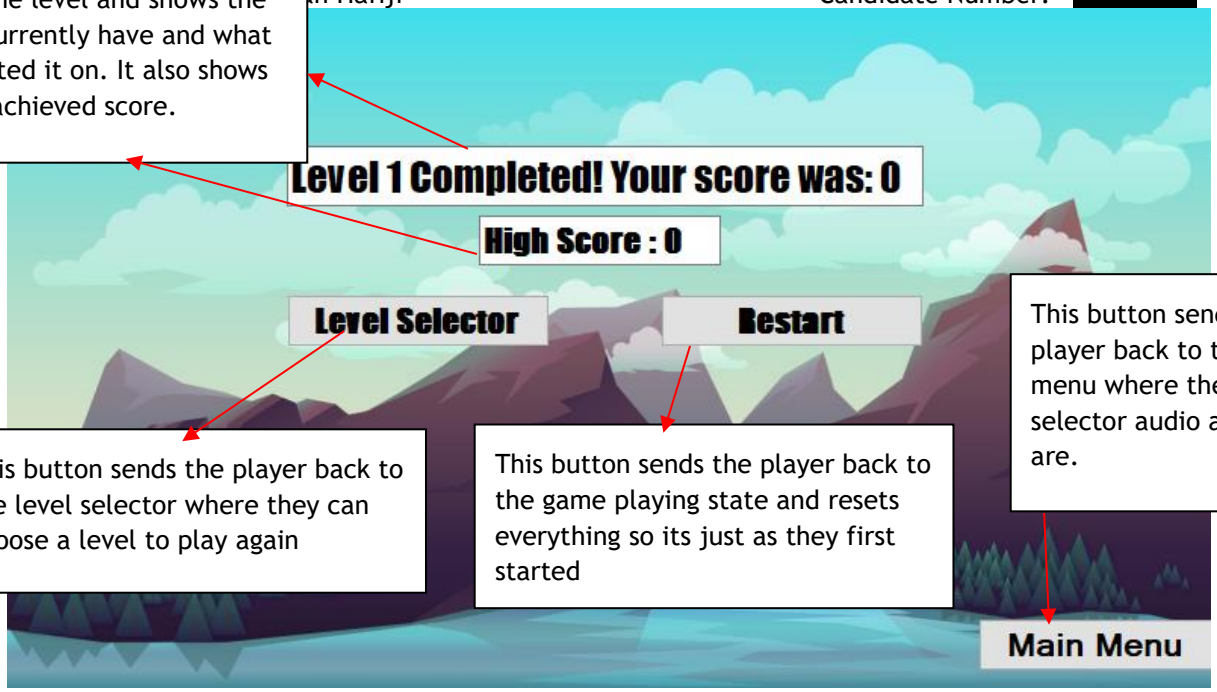


INTERVIEWING REDACTED FOR FEEDBACK

REDACTED: *This is a lot different to the design sketches you first made but im glad you changed it as it's a lot better and gives options for the player to go back to the level change it or go back to menu. Overall this has no issues.*

Me: Im glad you like it. Thank you

LEVEL FINISHED MENU



INTERVIEWING REDACTED FOR FEEDBACK

Me: I crated this level finished form and included your score and high score as requested by you. What do you thin of it?

REDACTED: *This definitely met my requirements and I really like the background choice you have selected. Overall I like it*

USABILITY FEATURES CONCLUSION

Usability feature	Success	Justification	Future actions to be taken
Main menu	Success	I created the main menu to how the stakeholder wanted it to be like and it doesn't need any changes	NA
Controls menu	Partially success	The controls menu is good in terms of comprehension due to images and text but it does not allow for the player to change the keybinds	If I want to be able to make it accessible to everyone I need to be able to make it so the player can change the keybinds for the controls
Level selector menu	Success	The level selector menu met the stakeholders requirements and does not need any changes	NA
Gameplay screen	Success	The gamescreen was simple but informative for the stakeholder when playing the game so	NA

		no additions need to be made	
Death screen menu	Borderline success	This was partially successful due to the game having a game over screen when the player reaches 0 health or timer = 0 but it does not have the design same as the design section	I wanted to have a death screen similar to the game Elden ring where the "You died" text comes up but due to time constraints I couldn't so in the future I would try implement this
Level finished menu	Partially success	I made a decent level finished menu which displays the score the time and the highscore of the current session and navigation buttons	To be able to meet the stakeholders requirements I would need to create a database that stores everyones names and the scores they have and display it as a scoreboard when you finish a level

REQUIREMENTS MET

Requirements	Justification	Met?	Explanation	Test evidence
Main menu and in-game menu	The game must have a main menu and a quick menu in-game to allow for navigation in the solution.	Fully met	NA	Acceptance test #8 #9
Easy controls	REDACTED mentioned that he didn't want complicated controls for the game so it must be simple to use	Fully met	NA	Post development test: Play controls
Levels	REDACTED mentioned that he wanted a variety of difficulty and a way to achieve that was by adding levels and making them harder as you progress	Partially met	I created levels for the game and was intended to make it increase in difficulty but due to time constraints I cannot do it	Post development test: Level selector screen
Background must be side scrolling	To make the game more immersive the background must not be static but it should move with the character	Fully met	NA	NA

Timer	Essential for giving the player a challenging environment as they are trying to beat the time	Fully met	NA	NA
Game over screen	When the user fails to complete the level in time or dies due to a enemy or the void a death screen should appear	Fully met	NA	Post development test: Game over screen
Power ups	The game must have some sort of boost or power-up feature so it isn't stale and the player has a way to improve at the game, which in turn increases the sense of achievement and the overall relaxing experience.	Fully met	NA	Acceptance test #11
Style of the game	REDACTED requested that the game not be too realistic in design and should have a cartoonish style to it	Fully met	NA	Acceptance test #7
Animations	To not make the game looks boring I need to add animations when the player moves and shoots	Fully met	NA	Acceptance test #12

SUCCESS CRITERIA

Success criteria	Justification	Met	Explanation	Test Evidence
Power ups	**REDACTED has requested the difficulty of the game to increase gradually but also make the character strong with it so there is not a power imbalance**	**Fully met**	**NA**	**Acceptance test #14**
Controls	**He has requested to use arrow keys for movement and spacebar to shoot**	**Fully met**	**NA**	Post development test: Player controls

Timer	**He has requested a timer to add intensity to the game**	**Fully met**	**NA**	**NA**
High score system	**The score system will be calculated by how many gems you have collected times the time left on the timer when you finish. For example 30 gems and 20 seconds left (30x20=60 score)**	**Partially met**	**I partially met this because I created a successful high score system but it doesn't save when the game is turned off so I would need to create a data base to do that but its too difficult and long to do it within the deadline**	**Acceptance test #1**
Lives	**I will be adding a health bar, not as much health as Stickfight but enough that a few hits wont instantly kill you. I may also add power-ups to add more hp to the player.**	**Fully met**	**NA**	**Acceptance test #6**
Score displayer	**There will be a score bar that will display the amount of gems collected, the timer and the total score.**	**Fully met**	**NA**	**Acceptance test #4**
Different Menus	**The menu will be simple and not too complicated. There may only be 2-3 options on the menu so its easy to navigate and easy to start the game.**	**Fully met**	**NA**	**Acceptance test #8 #9**
Different Background	**Game must have a variety of background so it looks more professional and immerisve**	**Fully met**	**NA**	**Acceptance test #7**
3 Levels	**The game will have 5 levels starting from easy then progressing harder and harder every level.**	**Partially met**	**Due to time constraints I could not create 3 levels so this success criteria is partially met**	Post development test: Level selector screen

Gems	**There will be around gems every level and they will be placed in easy to reach spots not blocked my enemies and some will be behind enemies you have to defeat**	**Fully met**	**NA**	**Acceptance testing #3**
5 Second Power-up timers	**Power-ups will not last permanently and will last for 10 seconds before reverting the character to normal as it was before**	**Partially met**	I could not meet this success criteria due to it being too difficult to add a temporary timer that would remove and add variables to the game	Acceptance test #11
20 Player damage	**When the player interacts with either the enemies of the void they should either die instantly or slowly start to die**	**Fully met**	**NA**	**Acceptance test #14**

REQUIREMENT/CRITERIA CONCLUSION

Improvement	How it will be done	Why it will improve the game
High score system	I will need to create a csv file that saves data such as the username of the player and the score they recieved. This should be able to load previous data into the game	This will allow the player to play the game, save their score and name and whenever they come back to it, it will load and show them their previous high scores and other people can also play
More levels (3-5)	I will need to create 2-4 more levels for my game and increase the difficulty, size and features they have	Increasing the amount of levels increases the amount of difficulty so it captures the players attention and makes them want to play the game more to complete it
Timed powerups	I will need to create separate timers for the powerups and start and stop them when the powerup is interacted with to change the variables that the powerup effects	In future levels, this could be a problem if there are multiple powerups as it would just keep stacking so making it timed allows for it to be added and removed so it doesn't make the game break

ACCEPTANCE TESTING

Number	Requirements	Input	Expected output	Actual output	Justification
1	The users final score is added to the scoreboard if the score is great than the scoreboards lowest score	**Valid:** A final score of 300 **Invalid:** The string'Pixelgamer49'	**Valid:** The final score of the player will be placed on the leaderboard accordingly next to the users name **Invalid:** The string 'Pixelgamer49' is displayed as the score and the name on the leaderboard	**Final score of the player is placed on the highscore textbox.**	**The justification for this is that the game should not display the username as the score so its incorrect if it does**
2	The users name can be used for the leaderboard if the name is > 2 and < 20 characters	**Valid:** The string 'Pixelgamer49' **Invalid:** The string 'A'	**Valid:** The string 'Pixelgamer49' is used on the leaderboard and placed next to the players final score **Invalid:** The string 'A' is placed on the leaderboard and placed next to the players final score3	**User cannot use names that re less than 2 letters or long er than 20**	**The justification for this is that a username cannot be less that 2 characters or longer than 20 so the game looks more compact and immersive for the player if its realistic names**
3	The player gains 1 point when a coin is picked up	**Valid:** When the player sprite comes into contact with a coin sprite, a point is added to the score **Invalid:** When the player sprite comes into contact with a coin sprite, a point isn't added	**Valid:** 1 point is added to the players score **Invalid:** No points is added to the players score	**1 point is added to the gem count/score of the player in the textbox in the game**	**The justification for this is that when the coin is picked up a point should be added to the score and this test checks if it works as it's a requirement from the stakeholder**
4	Score box at the top left of the screen	**N/A**	The scorebox is in the top left of the screen	**Scorebox is at the top left of the screen and changes**	**The justification for this is that the stakeholder**

				when certain conditions are met	**made this a requirement for the player as it needs a interactable health bar to keep track of how much health they have**
5	Ammo box at the bottom left	**N/A**	The scorebox is in the bottom left of the screen	**Ammo box is placed on the top left under the scorebox of the game and changes when the spacebar is presssed**	**The justification for this is that the player should be able to see how much ammo they have to make it easier so they don't have to guess how much they have**
6	100 health	**Valid:** The life variable will be set to 100 **Invalid:** The life variable will be set to another number than 100	**Valid:** When the level is selected, the life variable will be reset to 100 **Invalid:** When the level is selected, the life variable will be reset to a number other than 100	**Player receives 100 health when the game starts and resets each time**	**The justification for this is that the player requested for 100 health in the design success criteria and requirements so I adding this meets their requirements**
7	All sprites and backgrounds will be in a pixel art style	**N/A**	All sprites and backgrounds will be in a pixel art style	**Sprites and background are cartoon pixel art style**	**The justification for this is that a cartoon design theme is a requirement from the success criteria as the stakeholder specifically**

					requested this
8	Pressing the 'P' key to pause the game	**Valid:** When in the playing state of the game, the character 'P' is pressed **Invalid:** When in the playing state of the game, the character 'W' is pressed	**Valid:** The game state will change from the playing state from to paused state **Invalid:** The player's character will jump	**The pause screen shows and hides the game screen when the letter P is pressed**	**The justification for this is that the stakeholder requested that I create a pause function for the game**
9	Pressing the 'P' key to un-pause the game	**Valid:** When in the paused state of the game, the character 'P' is pressed **Invalid:** When in the playing state of the game, the character 'W' is pressed	**Valid:** The game state will change from paused to the playing state `+state **Invalid:** The player's character will jump	**The pause screen is hidden and shows the game screen when it is pressed again**	**The justification for this is that the player requested that I created a pause function for the game that works**
10	Dying animation for the player	**Valid:** Life counter for the player reaches 0 **Invalid:** Life counter for the player is > 0	**Valid:** The dying gif will play when the player life reaches 0 **Invalid:** The dying gif will not play	**When the players health reaches 0 or less the dying animation is played**	**The justification for this is that the player should have a dying animation when their health reaches 0 to indicate that they have died**
11	Player picks up power up	**Valid:** When the player sprite comes into contact with a power up it is removed **Invalid:** When the player sprite comes into contact with a power up it isn't removed	**Valid:** Player will receive a power up according to its sprite picture e.g. more speed and the power up sprite is removed **Invalid:** Player will not receive a power up according to its sprite picture and the power up	**The player gets boosts when the powerups are picked up and changes correspondin g code and variables**	**The justification for this is that powerups are a requirement from the stakeholder and testing this and making it valid meets the stakeholders request**

			sprite is not removed		
12	GIF changes according to the players controls and movements	**Valid:** When in the playing state the player presses a control for example 'W', 'A', 'D', 'Spacebar' **Invalid:** When in the playing state the player presses a control for examples 'O', 'Y', 'T', 'Delete'	**Valid:** The gif of the character changes according to the control that is pressed **Invalid:** The gif of the character does not change according to the control that is pressed	**The GIF changes when the player presses a control and does the correspondin g images**	**The justification for this is that GIF changes makes the game more immersive for the player and achieves the stakeholders requirement s**
13	Ammo box is 3	**Valid:** The ammo box variable is 3 as default **Invalid:** The ammo box variable is another number than 3 as default	**Valid:** When the level is selected, the ammo box variable is reset to 3 **Invalid:** When the level is selected, The ammo box variable is reset to a number that's not 3	**When the game starts the ammo box is reset to 3 and it can be removed or added to**	**The justification for this is that I am meeting the stakeholders requirement s of having 3 ammo when the game starts**
14	Life removal	**Valid:** Player sprite comes into contact with an enemy sprite an animation will play **Invalid:** Player sprite comes into contact with an enemy sprite an animation will not play	**Valid:** 1 life is removed when player sprite comes into contact with enemy sprite **Invalid:** No lives or more than 1 life is removed when the player sprite comes into contact with enemy sprite	**When the player comes into contact with a sprite it removes health from the health value and healthbar**	**The justification for this is that the game needs to be removing health when the player comes into contact with a enemy sprite**
15	Fireball removal	**Valid:** Player presses spacebar as an attack **Invalid:** Player presses P as an attack	**Valid:** Fireball animation is played and a bullet is removed from the ammo box **Invalid:** No bullets are removed and the game state goes	**When the spacebar is pressed the fireball image is shown and removes 1 ammo from the ammo variable**	**The justification for this is that the fireballs need to be remove when the player presses the spacebar button and**

			from playing state to paused state		**not any other button**
16	Sound effects	**Valid:** Player sprite comes into contact with a coin sprite **Invalid:** Player sprite comes into contact with a coin sprite	**Valid:** A coin pickup sound will play **Invalid:** No sound effects will be played	**Sound effects plays with correspondin g situations such as fireball shooting.**	**The justification for this is that there needs to be sound effects when the player interacts with something like a coin or presses spacebar**

ALPHA TESTING AGAINST SUCCESS CRITERIA

Criteria	Success	Evaluation	What could be done if not fully met
Power ups	Fully successful	I have successfully created working power ups such as the speed boost, health boost and ammo addition	NA
Controls	Fully successful	The controls are simple and it has met the success criteria of REDACTED for it to be simple keys that are commonly known	NA
Timer	Fully successful	Timer was successful and starts at 30 seconds and resets everytime the level is restarted or selected	NA
High score system	Fully successful	High score system was successful and met REDACTEDs request as it shows the value that had the highest score	NA
Lives	Fully successful	The health was successful as it restarts on 100 health everytime the player selects the level	NA
Score displayer	Fully successful	The score displayer was successful as it shows the current	NA

		score of the player and changes when certain conditions are met	
4 Menus	Fully successful	I have created 4 menus on the game that is efficient and easy to navigate for the user	NA
7 backgrounds	Partially successful	I didn't get to use enough backgrounds as I only made 2 levels for my games so there is not much variety in backgrounds	I need to add more levels or menus to reach the success criteria to be able to fulfill the stakeholders requirements
5 levels	Partially successful	Due to time constraints I could not meet this success criteria fully as I could only make 2 levels in the time period	I need to add 3 more levels to fulfill my stakeholders requirement
5 gems	Fully successful	Gem addition and the amount of gems are successful as I have met the players requests	NA
5 second power up timers	Partially successful	Power ups was a successful addition to my game but the timer for the power ups are something I did not have enough time to do	I couldn't give a timer to the power ups and it was just permanent but as its not a game breaking feature I will keep it like this until future updates.
20 player damage	Fully successful	This was successful as It would remove 20 player damage every time the player comes into contact with the enemy	NA

MAINTENANCE

Pyro gunner has some features that allow for future maintenance for example:

- Functions and scripts are organized
- Code is commented and explainations of them are included
- Blocks of code can be reused easily such as movement or shooting
- Variable are appropriately named for easy use
- Resources and materials for images and backgrounds are put inside in the game for future use

These features allow for either me or someone else to be able to either add onto my game or fix it so in terms of future maintenance the main functions and features of the game are easily understandable and easy to use.

However there are issues/features in my game that may hinder the ability to be able to maintain and update the game such as:

- Player progress saving through the game engine and everything is lost once the game is closed
- The limitations of what Visual Studio can actually do for example the size of a form is limited
- My game is 2d so if a 3d feature were to be tried to be implemented it would not work as well as it could or at all

Overall the game is incomplete as I have not finished all 5 levels I intended to make but due to time constraints I could not do it so any ideas ive made is not possible to translate into the game if I were to make it or someone else. The game also has a feature that wont be able to be translated into bigger or large form sizes which is the scrolling background as it only works best on a lengthy maps with a small width to give the illusion to the user that the background is moving.

FUTURE DEVELOPMENT AND THEIR LIMITATIONS

Feature	Justification	Limitations
Gem system	Most 2d platform games have a feature of having a currency system such as stickman archero which allowed for cosmetic upgrades or being able to unlock more levels in the game. This would be a good feature for my game for being able to maintain players playing the game	This system would be too complicated for a game such as mine due to the fact i would need a database to store players names and the amount of gems they have and previous purchases they have made in the game. I would also need a security system to make sure there are not any exploits in my game that allow for players to take advantage of
Storyline	To make a game more captivating a storyline would be able to allow for more immseriveness and involvement for the player which should allow for players to stay playing the game.	This would be limited by time constraints and creativity as i would need to be able to create a good storyline that makes sense and captivates a player and it needs to be done within a time frame to be able to meet deadlines
Sound effects	To make my game more immersive adding sound effects to the game would be able to also be a indicator for the player allowing them to know their surroundings easier and what things do.	The constraints for this would be the game engine as i have tried adding sound effects on my fireball and they worked but they were delayed and werent aligned the the button so using a different game engine could solve this issue
Original graphics	I could make my own graphics for things like the backgrounds, characters fireball gems and powerups because ive used the images from other websites and the internet so it could add a style to my game that i could create	Limitations of this is that im not experienced in the pixel art style of creating images and it would be a very lengthy process instead of just using images already made.

More levels	Instead of just leaving my game at 1 level i could create 4 more levels that i intended to do and increase the difficulty parallel to the level and it would create a challenge for the player	Limiations of this is the time constraints as i only had time to create 1 level for my game so i would need a lot more time or a increased deadline for it.

PROJECT APPENDIXES

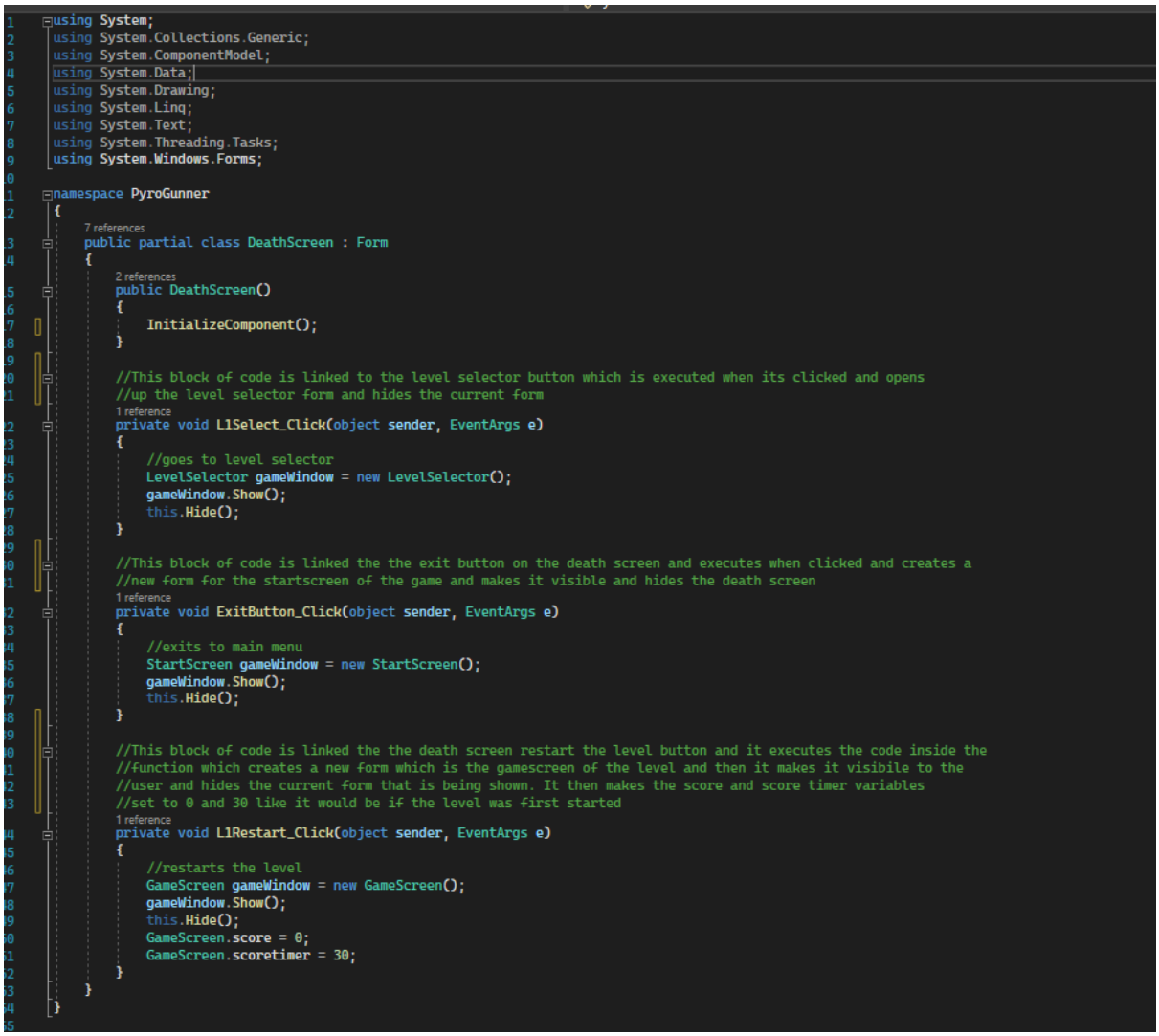
CODE LISTING

LEVEL 1 FINISH CODE

```
        }

        //This block of code is executed when the exit button is clicked on the level1finish form and it opens a new
        //form which is the startscreen and displays it and then hides the current form
        1 reference
        private void ExitButton_Click(object sender, EventArgs e)
        {
            StartScreen gameWindow = new StartScreen();
            gameWindow.Show();
            this.Hide();
        }

        //This function displays the score total of the player by multiplying the score variable to the scoretimer variable
        //and shows the score in the highscore textbox in the form. It then has a selection statement which chooses
        //whether the score was either higher or lower than the high score and executes corresponding to the outcome
        //which is either displaying that score or not replacing it. It also shows the amount of time taken in the textbox
        1 reference
        private void Level1Finish_Load(object sender, EventArgs e)
        {
            GameScreen.score = GameScreen.score * GameScreen.scoretimer;
            EndScore.Text = "Level Completed! Your score was: " + GameScreen.score;
            if (GameScreen.score > GameScreen.highscore)
            {
                GameScreen.highscore = GameScreen.score;
                HighScore.Text = "High Score: " + GameScreen.highscore;
            }
            else
            {
                HighScore.Text = "High Score :" + GameScreen.highscore;
            }
            Finishtimer.Text = "Your time was: " + finishtime;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    5 references
    public partial class Level1Finish : Form
    {
        //Makes a integer variable called finishtime to record how long it took the player to finish
        int finishtime;
        1 reference
        public Level1Finish()
        {
            InitializeComponent();
            //sets the gamescreens scoretimer variable to this forms finishtime integer variable
            GameScreen.scoretimer = finishtime;
        }

        //This block of code is linked to the level selector button which is executed when its clicked and opens
        //up the level selector form and hides the current form
        1 reference
        private void L1Select_Click(object sender, EventArgs e)
        {
            LevelSelector gameWindow = new LevelSelector();
            gameWindow.Show();
            this.Hide();
        }

        //This block of code is linked the the death screen restart the level button and it executes the code inside the
        //function which creates a new form which is the gamescreen of the level and then it makes it visibile to the
        //user and hides the current form that is being shown. It then makes the score and score timer variables
        //set to 0 and 30 like it would be if the level was first started
        1 reference
        private void L1Restart_Click(object sender, EventArgs e)
        {
            GameScreen gameWindow = new GameScreen();
            gameWindow.Show();
            this.Hide();
            GameScreen.score = 0;
            GameScreen.scoretimer = 30;
        }

        //This block of code is executed when the exit button is clicked on the level1finish form and it opens a new
        //form which is the startscreen and displays it and then hides the current form
        1 reference
```

DEATH SCREEN CODE

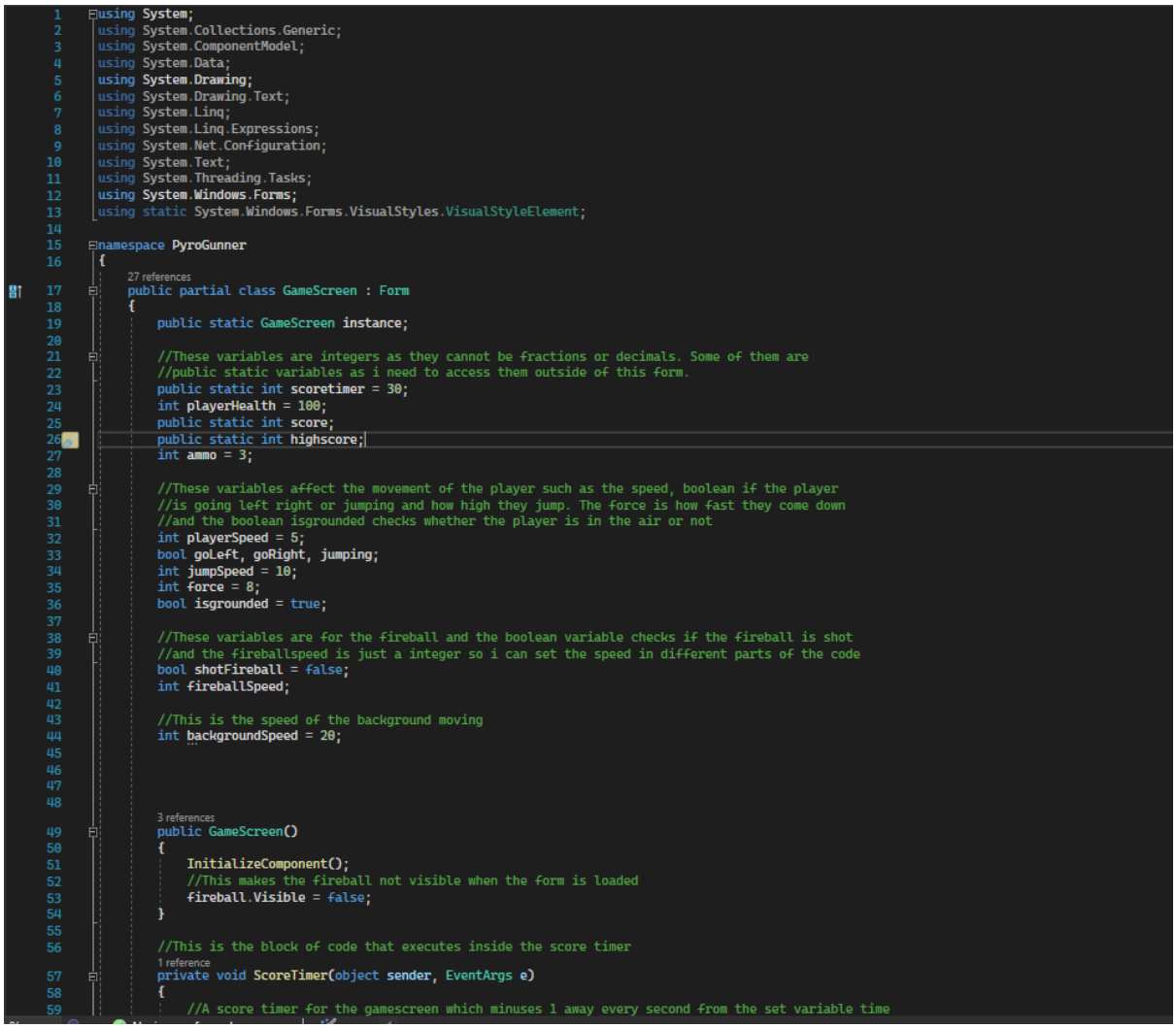


```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    7 references
    public partial class DeathScreen : Form
    {
        2 references
        public DeathScreen()
        {
            InitializeComponent();
        }

        //This block of code is linked to the level selector button which is executed when its clicked and opens
        //up the level selector form and hides the current form
        1 reference
        private void L1Select_Click(object sender, EventArgs e)
        {
            //goes to level selector
            LevelSelector gameWindow = new LevelSelector();
            gameWindow.Show();
            this.Hide();
        }

        //This block of code is linked the the exit button on the death screen and executes when clicked and creates a
        //new form for the startscreen of the game and makes it visible and hides the death screen
        1 reference
        private void ExitButton_Click(object sender, EventArgs e)
        {
            //exits to main menu
            StartScreen gameWindow = new StartScreen();
            gameWindow.Show();
            this.Hide();
        }

        //This block of code is linked the the death screen restart the level button and it executes the code inside the
        //function which creates a new form which is the gamescreen of the level and then it makes it visibile to the
        //user and hides the current form that is being shown. It then makes the score and score timer variables
        //set to 0 and 30 like it would be if the level was first started
        1 reference
        private void L1Restart_Click(object sender, EventArgs e)
        {
            //restarts the level
            GameScreen gameWindow = new GameScreen();
            gameWindow.Show();
            this.Hide();
            GameScreen.score = 0;
            GameScreen.scoretimer = 30;
        }
    }
}
```

PAUSE SCREEN CODE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    5 references
    public partial class PauseScreen : Form
    {
        1 reference
        public PauseScreen()
        {
            InitializeComponent();
        }

        //This function is linked to the back to game button which executes when the button is clicked
        //and hides the current form being shown
        1 reference
        private void BTG_Click(object sender, EventArgs e)
        {
            this.Hide();
        }

        //This block of code is linked to the level selector button which is executed when its clicked and opens
        //up the level selector form and hides the current form
        1 reference
        private void L1Select_Click(object sender, EventArgs e)
        {
            LevelSelector gameWindow = new LevelSelector();
            gameWindow.Show();
            this.Hide();
        }

        //This block of code is executed when the exit button is clicked on the level1finish form and it opens a new
        //form which is the startscreen and displays it and then hides the current form
        1 reference
        private void ExitButton_Click(object sender, EventArgs e)
        {
            StartScreen gameWindow = new StartScreen();
            gameWindow.Show();
            this.Hide();
        }
    }
}
```

GAME SCREEN CODE



```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Text;
using System.Linq;
using System.Linq.Expressions;
using System.Net.Configuration;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace PyroGunner
{
    27 references
    public partial class GameScreen : Form
    {
        public static GameScreen instance;

        //These variables are integers as they cannot be fractions or decimals. Some of them are
        //public static variables as i need to access them outside of this form.
        public static int scoretimer = 30;
        int playerHealth = 100;
        public static int score;
        public static int highscore;
        int ammo = 3;

        //These variables affect the movement of the player such as the speed, boolean if the player
        //is going left right or jumping and how high they jump. The force is how fast they come down
        //and the boolean isgrounded checks whether the player is in the air or not
        int playerSpeed = 5;
        bool goLeft, goRight, jumping;
        int jumpSpeed = 10;
        int force = 8;
        bool isgrounded = true;

        //These variables are for the fireball and the boolean variable checks if the fireball is shot
        //and the fireballspeed is just a integer so i can set the speed in different parts of the code
        bool shotFireball = false;
        int fireballSpeed;

        //This is the speed of the background moving
        int backgroundSpeed = 20;



        3 references
        public GameScreen()
        {
            InitializeComponent();
            //This makes the fireball not visible when the form is loaded
            fireball.Visible = false;
        }

        //This is the block of code that executes inside the score timer
        1 reference
        private void ScoreTimer(object sender, EventArgs e)
        {
            //A score timer for the gamescreen which minuses 1 away every second from the set variable time
```

```
            //and updates the time in the textbox. Also shows the amount of ammo the player has in the text
            scoretimer--;
            ammotext.Text = "Ammo= " + ammo;
            Time.Text = "TIME: " + scoretimer;


            //This if statement checks if the player has intersected with the enemy and if it has it removes
            //20 damage from the playerhealth
            if (Player.Bounds.IntersectsWith(enemy.Bounds))
            {
                playerHealth -= 20;
            }


            //This if statement executes when the scoretimer reaches 0 and stops the maintimer of the form
            //and displays the deathscreen form and shows it and makes the gamescreen form not visible
            //and stops the score timer
            if (scoretimer < 0)
            {
                MainTimer.Stop();
                DeathScreen gameWindo = new DeathScreen();
                gameWindo.Show();
                this.Hide();
                ST.Stop();
            }
        }

        //This is the block of code that executes inside the main timer of the gamescreen
        1 reference
        private void MainTimerEvent_Tick(object sender, EventArgs e)
        {
            //This displays the players score and sets the player picturebox to the jumpspeed variable
            Scorebox.Text = "Score " + score;
            Player.Top += jumpSpeed;

            //This if statement executes if the player intersects with the flag picturebox and makes a new
            //form and hides the current form and stops the main timer
            if (Player.Bounds.IntersectsWith(flag.Bounds))
            {
                Level1Finish gameWindow = new Level1Finish();
                this.Hide();
                gameWindow.Show();
                MainTimer.Stop();
            }


            //This if statement executes if the player health variable is less than 1 and it stops the main timer
            //and makes a new form as the death screen and hides the current form
            if (playerHealth < 1)
            {
                MainTimer.Stop();
                DeathScreen gameWindow = new DeathScreen();
                gameWindow.Show();
                this.Hide();
            }

            //These if statement checks if the goLeft or goRight variables are set to true and then makes the
            //MoveGameElements set to either foward or back to execute code inside that function
            if (goLeft == true)
            {
                MoveGameElements("foward");
            }
```

```
            if (goRight == true)
            {
                MoveGameElements("back");
            }


            //This piece of code checks if the shotFireball boolean variable is set to true and if it is, it sets
            //the fireballspeed to -15 and makes the left of the fireball picturebox the fireballspeed to make it
            //move fowards. If it doesnt it makes the fireball left of picturebox to -300 and set the fireballspeed
            //set to 0 so it doesnt move
            if (shotFireball == true)
            {
                fireballSpeed = -15;
                fireball.Left -= fireballSpeed;
            }
            else
            {
                fireball.Left = -300;
                fireballSpeed = 0;
            }


            //These if and else statement check if the jumping boolean variable is set to true and sets the jumpspeed
            //to -12 and the force to -1. If its false the jumpspeed is set to 12. If the jumping variable is true and
            //the force is less than 0 it sets the jumping to false
            if (jumping == true)
            {
                jumpSpeed = -12;
                force -= 1;
            }
            else
            {
                jumpSpeed = 12;
            }
            if (jumping == true && force < 0)
            {
                jumping = false;
            }


            //This is the playermovement in the form and if the goLeft variable is set to true and the player is further
            //than 60 on the form it sets the player picturebox to move left at the playerspeed variable. If its goRight
            //and the right if the picturebox is less than 60 off the width of the form, it sets the player picturebox
            //speed to playerspeed variable
            if (goLeft == true && Player.Left > 60)
            {
                Player.Left -= playerSpeed;
            }
            if (goRight == true && Player.Left + (Player.Width + 60) < this.ClientSize.Width)
            {
                Player.Left += playerSpeed;
            }


            //This is a foreach loop that sets a condition for the form when then controls is set to true
            foreach (Control x in this.Controls)
            {

                //This is the if statement which checks if the platform is being interacted with and executes code inside
                // if its true
                if (x is PictureBox && (string)x.Tag == "platform")
```

```
            if (x is PictureBox && (string)x.Tag == "platform")
            {
                //This forces the players picturebox to move upwards and the top of the player is put above the platform
                //and the jumpspeed is reduced to 0. It also changes the boolean variable is grounded to true
                //which executes another function when the condition is met. It also brings the players picturebox to the
                //front of the platform
                if (Player.Bounds.IntersectsWith(x.Bounds) && jumping == false)
                {
                    force = 8;
                    Player.Top = x.Top - Player.Height;
                    jumpSpeed = 0;
                    isgrounded = true;
                }
                x.BringToFront();
            }


            //This is the if statement which checks if the gem picturebox has been interacted with
            if (x is PictureBox && (string)x.Tag == "gem")
            {
                //This checks if the player comes into contact with the gem and if the gem visibilty is true it executes the code inside
                // and adds score to the scorebox and makes the gem visibility to false to make it not show
                if (Player.Bounds.IntersectsWith(x.Bounds) && x.Visible == true)
                {
                    score++;
                    x.Visible = false;
                }
            }


            //This is the if statement which checks if the ammo powerup is interacted with and if it is, it sets the location to the ammobox
            //and makes it not visibile and add 1 to the ammo variable
            if (Player.Bounds.IntersectsWith(ammo1.Bounds))
            {
                ammo1.Location = AmmoBox.Location;
                ammo1.Visible = false;
                ammo++;
            }


            //This is the if statement which checks if the speed powerup is interacted with and if it is, it increases the speed of the player
            // and makes it not visible
            if (Player.Bounds.IntersectsWith(speed.Bounds))
            {
                playerSpeed = playerSpeed + 5;
                speed.Visible = false;
                speed.Location = AmmoBox.Location;
            }


            //This is the if statement for the medkit powerup and if the medkit is interacted with and the player health is 100, nothing will happen
            //but if its not it adds 5 onto the player health variable
            if (Player.Bounds.IntersectsWith(MedKit.Bounds))
            {
                if (playerHealth == 100)
                {
                    healthbar.Value = playerHealth;
                }

                else
                {
```

```
                if (Player.Bounds.IntersectsWith(MedKit.Bounds))
                {
                    if (playerHealth == 100)
                    {
                        healthbar.Value = playerHealth;
                    }

                    else
                    {
                        playerHealth = playerHealth + 5;
                        MedKit.Visible = false;
                        MedKit.Location = AmmoBox.Location;
                    }
                }


                //This makes the player health value set to the player health bar
                if (playerHealth > 1)
                {
                    healthbar.Value = playerHealth;
                }


                //If the fireball intersects with the enemy the enemy is set to not visible
                if (fireball.Bounds.IntersectsWith(enemy.Bounds))
                {
                    enemy.Visible = false;
                    enemy.Location = AmmoBox.Location;
                }
            }

        }


        //This is the keydown function and if a key is pressed code will execute in here
        1 reference
        private void KeyIsDown(object sender, KeyEventArgs e)
        {
            //Checks if the left arrow key is pressed and sets the goLeft variable to true
            if (e.KeyCode == Keys.Left)
            {
                goLeft = true;
            }

            //Checks if the right arrow key is pressed and sets the goRight variable to true
            if (e.KeyCode == Keys.Right)
            {
                goRight = true;
            }

            //Checks if the up arrow key is pressed and if jumping is false and isgrounded is true and
            //sets the jumping variable to true and is grounded to false
            if (e.KeyCode == Keys.Up && jumping == false && isgrounded == true)
            {
                jumping = true;
                isgrounded = false;
            }

            //Checks if the spacebar key is pressed and checks if ammo is more than 0 and shoots the fireball
            //and makkes it visible and shoots it from the middle of the player picturebox and starts the
            //fireball gif
            if (e.KeyCode == Keys.Space)
```

```
            if (e.KeyCode == Keys.Space)
            {
                if (ammo > 0)
                {
                    fireball.Visible = true;
                    ammo--;
                    fireballSpeed = -5;
                    fireball.Location = new Point(Player.Location.X + 100, Player.Location.Y);
                    Player.Image = Properties.Resources.fireball1;
                }
            }


            //Checks if the P key is pressed and opens up the pause screen and shows it
            if (e.KeyCode == Keys.P)
            {
                PauseScreen gamewindow= new PauseScreen();
                gamewindow.Show();
            }
        }


        //This function is for the keyup and executes code when a key is depressed
        1 reference
        private void KeyIsUp(object sender, KeyEventArgs e)
        {

            //Checks if the left arrow key is depressed and sets goLeft to false
            if (e.KeyCode == Keys.Left)
            {
                goLeft = false;
            }

            //Checks if the right arrow key is depressed and sets goRight to false
            if (e.KeyCode == Keys.Right)
            {
                goRight = false;
            }

            //Checks if the up arrow key is depressed and jumping is false and sets jumping to false
            // and makes the image to standing
            if (e.KeyCode == Keys.Up && jumping == false)
            {
                jumping = false;
                Player.Image = Properties.Resources.standing;
            }

            //Checks if the spacebar key is depressed and sets the shotfireball statement to true and sets
            // it to the player width and player height and sets it to standing
            if (e.KeyCode == Keys.Space)
            {
                shotFireball = true;
                Player.Width += 40;
                Player.Width -= 40;
                Player.Image = Properties.Resources.standing;
            }
        }

        //This is a function to move the game elements depending on the tag they have and the direction it is set in
        2 references
        private void MoveGameElements(string direction)
        {
```

```
        //This is a function to move the game elements depending on the tag they have and the direction it is set in
        2 references
        private void MoveGameElements(string direction)
        {

            //This is a foreach loop that sets a condition for the form when then controls is set to true
            foreach (Control x in this.Controls)
            {

                //Moves the game elements "platform", "gem, "flag" "ammo1" "speed" " medkit" and "enemy" back and fowards corresponding the the pressed down key
                if (x is PictureBox && (string)x.Tag == "platform" || x is PictureBox && (string)x.Tag == "gem" ||
                    x is PictureBox && (string)x.Tag == "enemy" || x is PictureBox && (string)x.Tag == "ammo1" ||
                    x is PictureBox && (string)x.Tag == "MedKit" || x is PictureBox && (string)x.Tag == "flag" ||
                    x is PictureBox && (string)x.Tag == "speed")
                {
                    //The goLeft and goRight variables changes these and sets the background speed to this
                    if (direction == "back")
                    {
                        x.Left -= backgroundSpeed;
                    }
                    if (direction == "foward")
                    {
                        x.Left += backgroundSpeed;
                    }
                }
            }
        }
    }
}
```

INSTRUCTION SCREEN CODE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    5 references
    public partial class InstructionScreen : Form
    {
        1 reference
        public InstructionScreen()
        {
            InitializeComponent();
        }

        //This block of code is executed when the exit button is clicked on the level1finish form and it opens a new
        //form which is the startscreen and displays it and then hides the current form
        1 reference
        private void ExitButton_Click(object sender, EventArgs e)
        {
            StartScreen gameWindow = new StartScreen();
            gameWindow.Show();
            this.Hide();
        }
    }
}

```

LEVEL SELECTOR CODE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    11 references
    public partial class LevelSelector : Form
    {
        4 references
        public LevelSelector()
        {
            InitializeComponent();
        }

        //Sets the gamescreen score and scoretimer variables to the set amount needed
        1 reference
        private void LevelSelector_Load(object sender, EventArgs e)
        {
            GameScreen.score = 0;
            GameScreen.scoretimer = 30;
        }

        //This block of code is executed when the exit button is clicked on the level1finish form and it opens a new
        //form which is the startscreen and displays it and then hides the current form
        1 reference
        private void ExitButton_Click(object sender, EventArgs e)
        {
            StartScreen gameWindow = new StartScreen();
            gameWindow.Show();
            this.Hide();
        }

        //Opens a new gamewindow which is the gamescreen and shows it and hides the level selector screen
        1 reference
        private void L1_Click(object sender, EventArgs e)
        {
            GameScreen gameWindow = new GameScreen();
            gameWindow.Show();
            this.Hide();
        }
    }
}
```

MAIN SCREEN CODE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PyroGunner
{
    13 references
    public partial class StartScreen : Form
    {
        6 references
        public StartScreen()
        {
            InitializeComponent();
        }

        //Opens a new window when the start button is clicked and hides the main menu screen
        1 reference
        private void start_Click(object sender, EventArgs e)
        {
            LevelSelector gameWindow = new LevelSelector();
            gameWindow.Show();
            this.Hide();
        }

        //Opens a new game window which is instruction screen and hides it and shows the screen
        1 reference
        private void InstructionMenu_Click(object sender, EventArgs e)
        {
            InstructionScreen gameWindow = new InstructionScreen();
            gameWindow.Show();
            this.Hide();
        }


    }
}
```

INTERVIEWS

INTERVIEW #1

Me: How do you like the design of the screens that are currently made?

Him: It looks good and it has met my requirement of the background style

Me: Do you need me to do any changes to it?

Him: No its all good

INTERVIEW #2

Me: I have created the controls and tried to meet your requirements. Are they up to your standards?

Him: The controls are good and easy to use as its just keys that are commonly used but I found a bug that holding the spacebar keeps making the character move up

Me: Thank you for the feedback ill fix it after this interview

INTERVIEW #3

Me: I've now added the powerup and gem pickup and want to know what you think.

Him: These powerups are good and meet my success criteria but could you add a player speed powerup as it's a bit slow

Me: I will do that as soon as the interview is done

INTERVIEW #4

Me: I've created a successful fireball that shoots from the player. What do you think of it?

Him: The animation of the fireball is really good and it moves smoothly so it looks good. Could you add animations to the player when you shoot the fireball?

Me: I'm glad you like the fireball and I will try to add a animation for the player

INTERVIEW #5

Me: I've created the enemy and want your opinion on it and any changes I can make.

Him: The enemy AI is good and it works as intended and meets my success criteria/requirements but the player damage is too small for it to be intense.

Me: Ok I will increase the damage

INTERVIEW #6

Me: Does the scrolling background meet your success criteria?

Him: Everything is good and immersive so you don't need to change everything

INTERVIEW #7

Me: Is the screens up to your standards?

Him: Everything is good but for the level finish screen is it possible for you to display the time taken to do the level

Me: I can certainly do that

BIBLIOGRAPHY

These are the references for the images and backgrounds I used.

BACKGROUND AND ENEMY SPRITE

https://craftpix.net/freebies/free-wizard-sprite-sheets-pixel-art/

https://craftpix.net/freebies/free-fairy-tale-game-backgrounds/

https://craftpix.net/freebies/free-forest-bosses-pixel-art-sprite-sheet-pack/

PLAYER SPRITE AND FIREBALL

C# Tutorial – Create a Street Fighter Game Demo in Windows Forms and Visual Studio | Moo ICT – Project Based Tutorials

The rest of the images are from google images so I cant find it again to reference.