ALONE

Name: Stergious Aji

Candidate Number: 4211

Centre Number: 71887

Lumen Christi College

Software Systems Development

Table of Contents

ALONE

Background

The renowned game-developing company, known as Monsoon Games, are undertaking their annual game making competition where people all around the world submit a game that they have designed and programmed in Visual Studio for a chance to earn a well-paying permanent post at the company where they can earn an annual salary of £60,000 and the opportunity for their game to be published and mass produced around the world. This year, Monsoon Games are requesting for simple, 2-Dimensional games, revolving around an 'Apocalyptic' Theme.

I have decided to create a zombie survival, top-down shooter game called 'Alone' to enter into this competition. It will have very basic game mechanics with easy-to-learn controls, as a result, it will be playable by anyone, no matter their level of skill in using computers. 'Alone' will be targeted at anyone above the age of 7 years as it will contain very mild forms of violence and will be slightly scary making it unsuitable for younger audiences. The sole purpose of the application will be to entertain. I want my game to have minimal game mechanics yet be fun at the same time for everyone to enjoy. To play the game, the user will have to register an account and login to start playing the first level of two. Help with controls and how to play the game will be available in a help page. I plan to make my game fun as well as challenging, so as the user progresses through the levels obtaining higher scores, the difficulty also builds up with the player's skills.

Initial Ideas

'Alone' will place the player into the middle of a post-apocalyptic world, 6 months after a world-ending, zombie epidemic. All the player knows is that they could be the very last survivor and that they must use the last of their will, and their limited resources, to ensure that it stays that way. The game will feature two real-life locations from New York City, the middle of the famous Central Park, and a Junction just outside the Park. The play areas will be restricted in size, giving the game a very claustrophobic and intense atmosphere.

'Alone' will offer a design screen for the user to design their character with various cosmetics at the start of the game. When the game starts, the character is transported to an unfamiliar place with only a pistol to defend themselves, with only one objective: survive. The player has to navigate the map and kill any enemies that spawn near them. Ammo will be sparse around the map and so the player is forced to play strategically to get a high score in the game. A score will be calculated when the player dies in the game, encouraging the player to replay the game to obtain a higher score. Survival and strategy will be the key to achieving a high score in 'Alone'. There will be a maximum of 2 levels/maps.

User Requirements

General Requirements

1. All parts of the application must have an aesthetically pleasing and eye-catching design using a variety of appropriate gender-neutral colours to appeal to a varied target audience and keep the user captivated and engaged.

2. All pages must have a user-friendly and fluid interface with various ways to access all other pages. Easily readable text should be present in order to aid the user in performing their intended task using simple language for users of all ages and backgrounds to understand. The user should be notified when appropriate, using message boxes, tool tips and error messages.

3. Music and various sound effects should play in the appropriate forms to further immerse the user into the game and give an enjoyable experience to the user.

4. Any place where the user must enter their password, a masked text field must be used for security, giving a more professional feel to the application.

Main Menu

5. The Main Menu page will be the hub or home page which will be the first form to load and where the title of the game will be displayed in large and all other forms can be accessible by clicking specific buttons. (Register button, Login button, Play button, Help button).

6. The play button should only be enabled once the user is logged in, hence users not logged in cannot gain access to the game levels and start playing. While the play button is disabled, a notice should be displayed when the user hovers over the button, telling the user to login first.

7. Once the user is logged in, the play button, once clicked, will transport the user to a game level select screen for the user to select one of the two levels to play. Each of the level buttons should be accompanied with a description of the level for the user to read. The user must be provided with a back button to go back to the main menu. The user will also be provided with a high scores button to transport them to the high scores table.

8. The user must be provided with a means to exit completely out of the application with an exit/quit button on the main menu screen.

Registration

9. Users are required to register an account to play the game on the registration page, providing their first and last names, a valid username, their e-mail address, their date of birth and a valid password for their account using various input methods (e.g. Text Boxes). Buttons to other pages (i.e. a Taskbar) should be available for easy manoeuvring between screens (Register button, Home button, Login button, Help button).

10. The first and last names should not contain any numerical values, white spaces or punctuation/symbols and the name text fields should not be left empty, otherwise an error will be displayed notifying the user what they have done wrong.

11. The username text field should not be left empty, otherwise an error will be placed telling the user to fill the text box. An error will be shown if the user inputted a username that already exists, telling the user to choose a different username.

12. The e-mail address should contain a '@' and a '.' characters to be a valid e-mail address and the text field should not be left blank, otherwise an error will be displayed advising the user to enter a valid e-mail address.

13. The user must be above 7 years of age to create an account and so their date of birth will be checked to see if they are 7 years or above. If not, then an error will be shown notifying the user that they cannot register an account.

14. The password must contain at least 6 characters and 1 digit to be secure and valid. If the password is not valid or the text field is not completed, then an error will be assigned informing the user what they have done wrong.

Login

15. Users are required to login their registered account on the login page, providing their username and their password in the text fields that will be provided in the page along with a task bar with buttons to access all other pages (Login button, Home button, Register button, Help page).

16. Errors must be displayed if any of the text in the text box fields do not match with existing member usernames and passwords or if left empty. The user cannot play the game until they pass the login page.

Help Guide

17. Users can visit the help page to check the controls of the game and get an explanation on what the game is, how to play it and how it functions. There should be a scroll bar to let the user scroll through the text.

Game Levels

18. The appropriate objects should draw onto the background during the run of the program.

19. The user can have various methods to move the player (e.g. Arrow keys or WASD keys). This makes the game playable by a range of users who may prefer a specific way to move the player.

20. The user should not be allowed to move the player over objects (e.g. Trees, cars, etc.) and the information labels (e.g. Score box, Health bar, Ammo box) in the background of the level.s In other words, invisible boundaries should be set up to restrict the play area.

21. The user should be able to fire their weapon using a trigger (e.g. Key press or Mouse Left Click). A gunshot sound effect should play to indicate that they have fired their weapon, this should not be too loud to not scare the user. When the bullets make contact with the zombies, the zombies should lose their health or get eliminated. The ammo in the player's gun clip should decrement and the player's score should increment when the zombies gets eliminated.

22. The user should be able to reload their weapon using a key press (e.g. 'R'). The reload sound effect should play alongside a progress bar on top of the player indicating the reload time delay. The reload event should transfer the ammunition from the player's inventory to the gun clip.

23. The zombies and ammunition refill boxes are required to spawn at random places on the form to create a different experience for the user every time the level is played making it more entertaining and fun. When the ammunition refill boxes are collected by the user, the ammunition must be stored in their ammo pouch.

24. The zombies must move by themselves towards the player to attack them. This will make the game seem alive and mimics the user fighting against the computer. When the zombies make contact with the player, the player should lose health.

25. On each of the game levels, the user will have score box, displaying their current score, a health box, displaying their current health status, and a gun information and ammunition status box, displaying current gun held and amount of ammunition the player has left. Each of these will update when a specific event occurs.

26. The health box and ammunition box will change colours to warn the user about the status of their health or amount of ammunition.

27. The user can pause the game using a trigger and get an opportunity to return to the main menu from the game levels with the aid of a button or pick up exactly where they left off in the game. In addition, a help button will also appear allowing the user to gain access to the help page directly from the game levels.

28. Each of the two levels will contain different designs for the background to help the user distinguish between them and make each level have their own unique characteristics, creating different refreshing experiences with each.

29. Each of the two levels must have a game over screen when the player's health reaches zero. Text should appear notifying the user that they have lost as well as a retry button, to play the level again, a main menu and help button.

- Level One

30. The first level will be a trial/beginner level with minimal difficulty for all users to have a fun and enjoyable experience. Enemies will be slow and exert small amounts of damage when in contact with the player and will appear in small quantities at a time.

31. If the player's score reaches a specific number (e.g. 100) then the difficulty will increase by a small amount to make the gameplay refreshing and prepare the user for the second level.

- Level Two

32. The second and final level will have a very high difficulty to truly test the user in their skill and ability in the game. The enemies will be faster, stronger and will spawn in large quantities at a time.

33. If the player's score reaches a specific number (e.g. 50) then the difficulty will increase by a small amount to make it more challenging and bring about a more refreshing and entertaining gameplay.

34. The second level should contain a different weapon for the user to use to make it unique and fun. The different weapon should play a different gunshot and reload sound effects to make the game more realistic.

High Score

35. The high score screen should display a table with the players with the highest score which should be capable of updating when the users get a higher score.

36. The currently logged in user's username should be clearly highlighted if it is present on the leader board.

37. The user must be provided with a back button to return them back to the game level select screen.

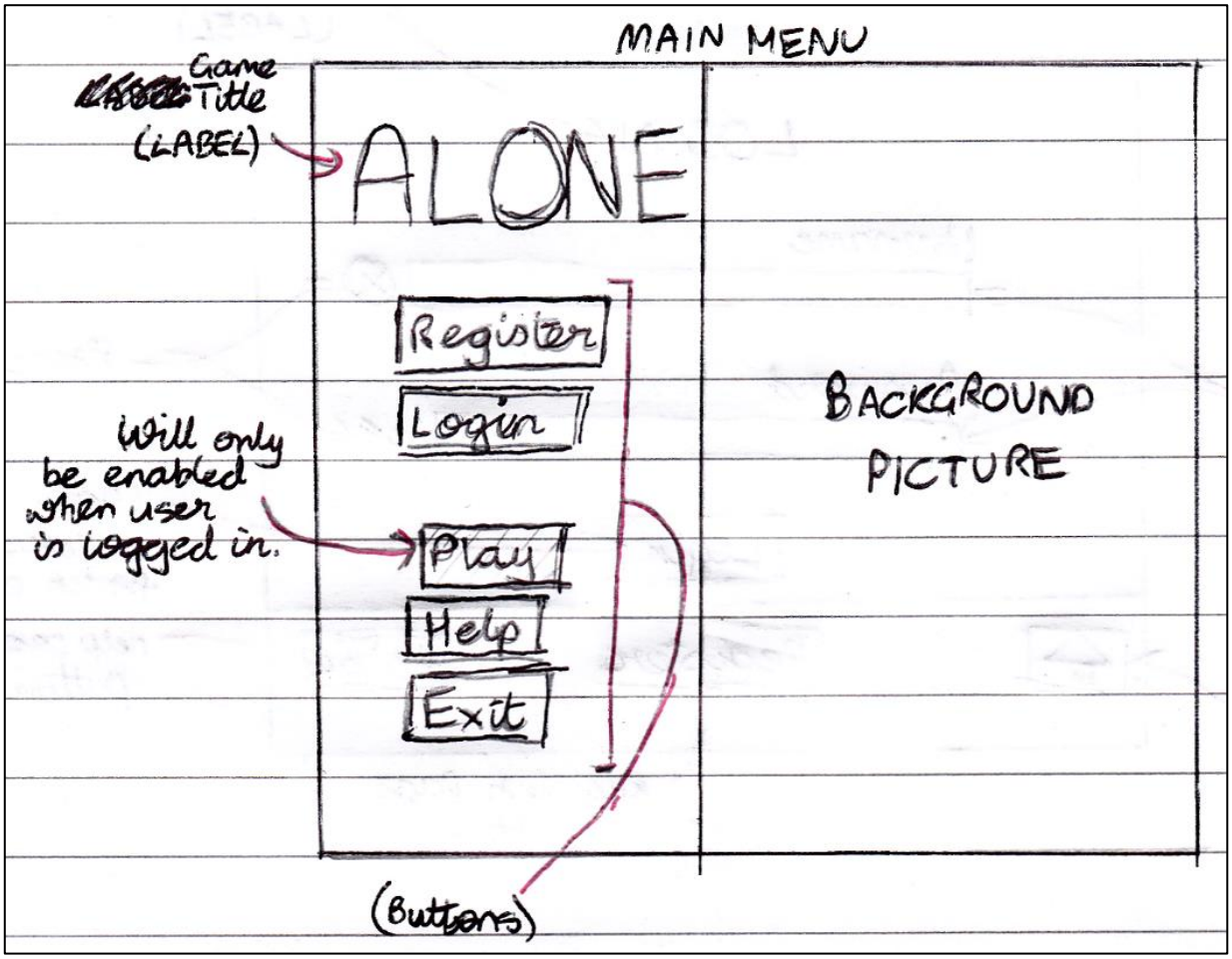
Overview of Game

‘Alone’ is a zombie survival, top-down shooter type game where the player is able to move a character in 8 different directions in a 2-Dimensional axis, using key presses. A number of zombies will spawn on random points on the screen and will move towards the player to attack them. The aim of the game is for the player to avoid the zombies by moving away from them and eliminating them using the weapon provided and to survive as long as possible. The zombie will attack the player if they make contact with the them, decreasing the health of the player. The game will end if their health reaches zero health points. The player will have a limited supply of ammunition which will run out fast, so it is important that the user utilises it intelligently. Ammunition refill boxes will be provided if the player runs out of all their ammo, this will spawn at a random location on the play area and the user must move towards it to collect it. If the bullet from the player’s gun reaches the zombies, then the zombies’ health will decrease until their health reaches zero which will count as an elimination. The player’s score increments every time the user terminates a zombie. As the player’s score increases, the difficulty will gradually increase meaning the enemies move faster and will be stronger and so will have more health and will deal more damage to the player. Additionally, the enemies will appear in larger quantities at a time.

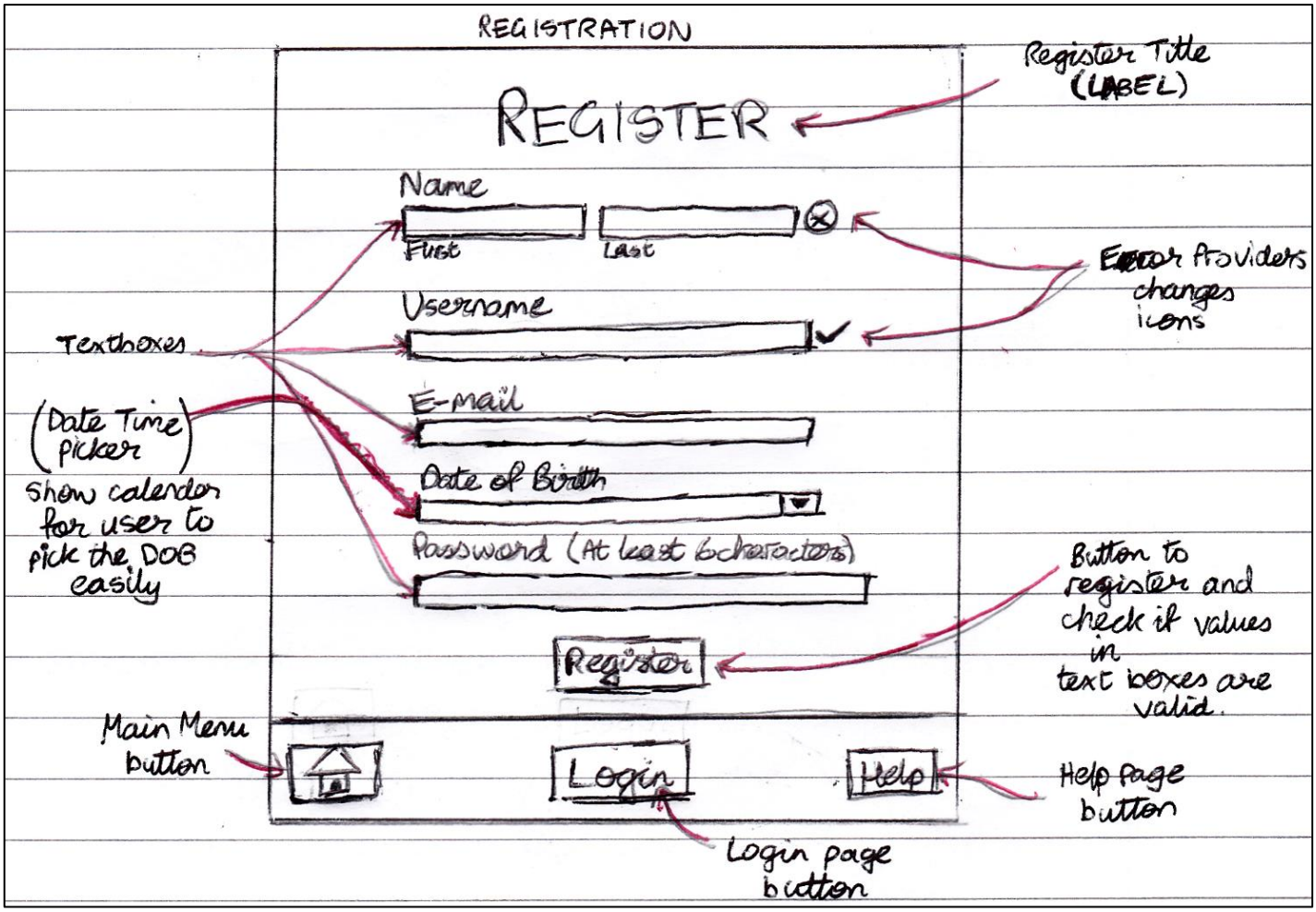
‘Alone’ will have a main menu screen, where all other pages can be accessed using button presses. The user will have to register an account to play the game, this will be made available in the registration page. When the user successfully signs up, or if they already have registered before, they can login on the login page, so their score and achievements made in the game can be saved to their user profile. A help/guide page will be made accessible from all other pages, so the user can use the application with ease. Once the user is logged in, they can then select which level they are willing to play. ‘Alone’ will have 2 levels. The first level will be a beginning or training level where the difficulty is kept to a minimum, so the user can easily learn to play the game and get used to the controls. The second level will be designed to be more complex for users who are very skilled at the game. Lastly, a leader board will also be available displaying the usernames and high scores of the players who have achieved the top ten high scores in the database.

Drawings

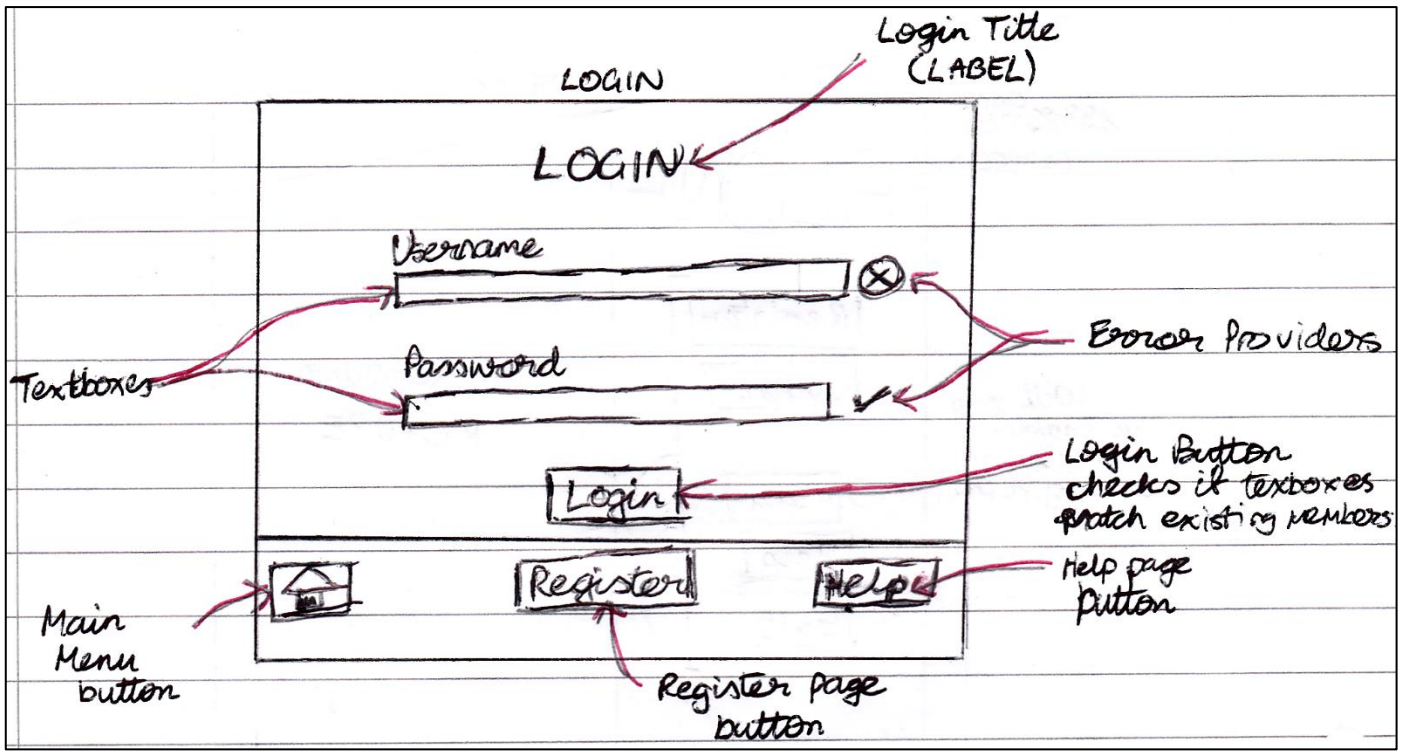
Main Menu



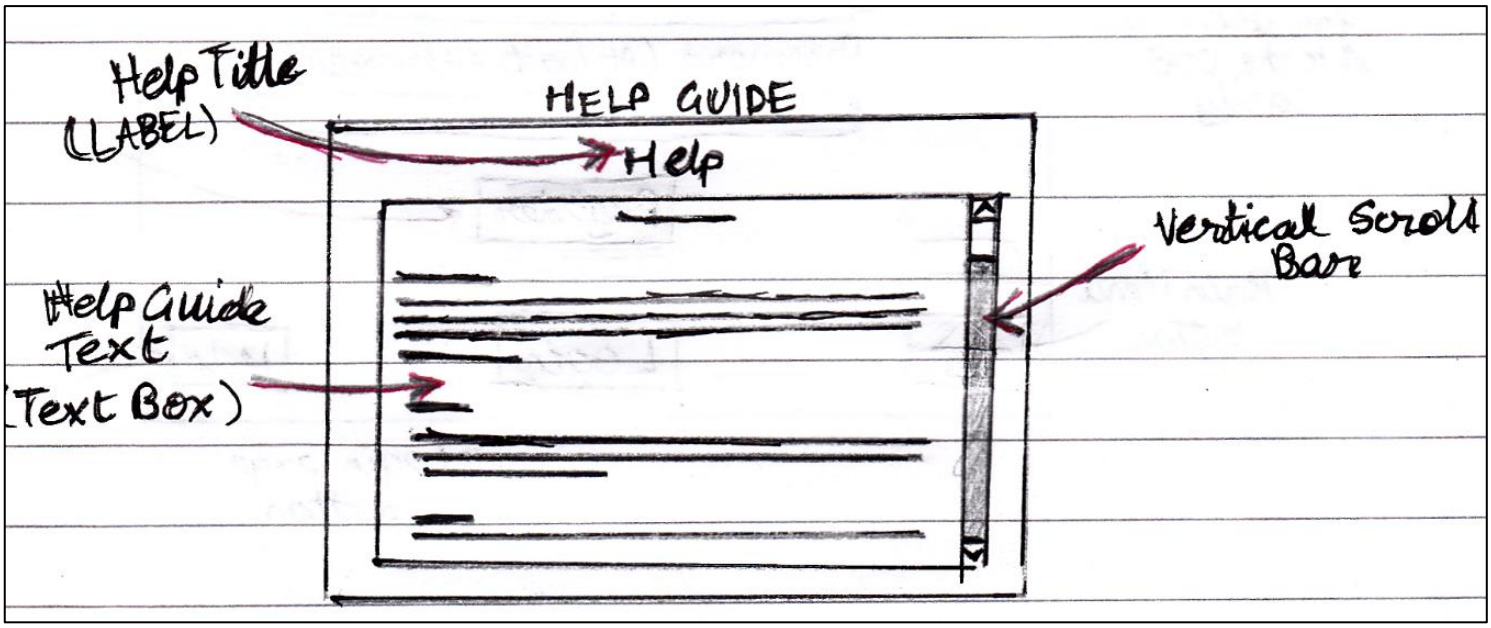
Registration



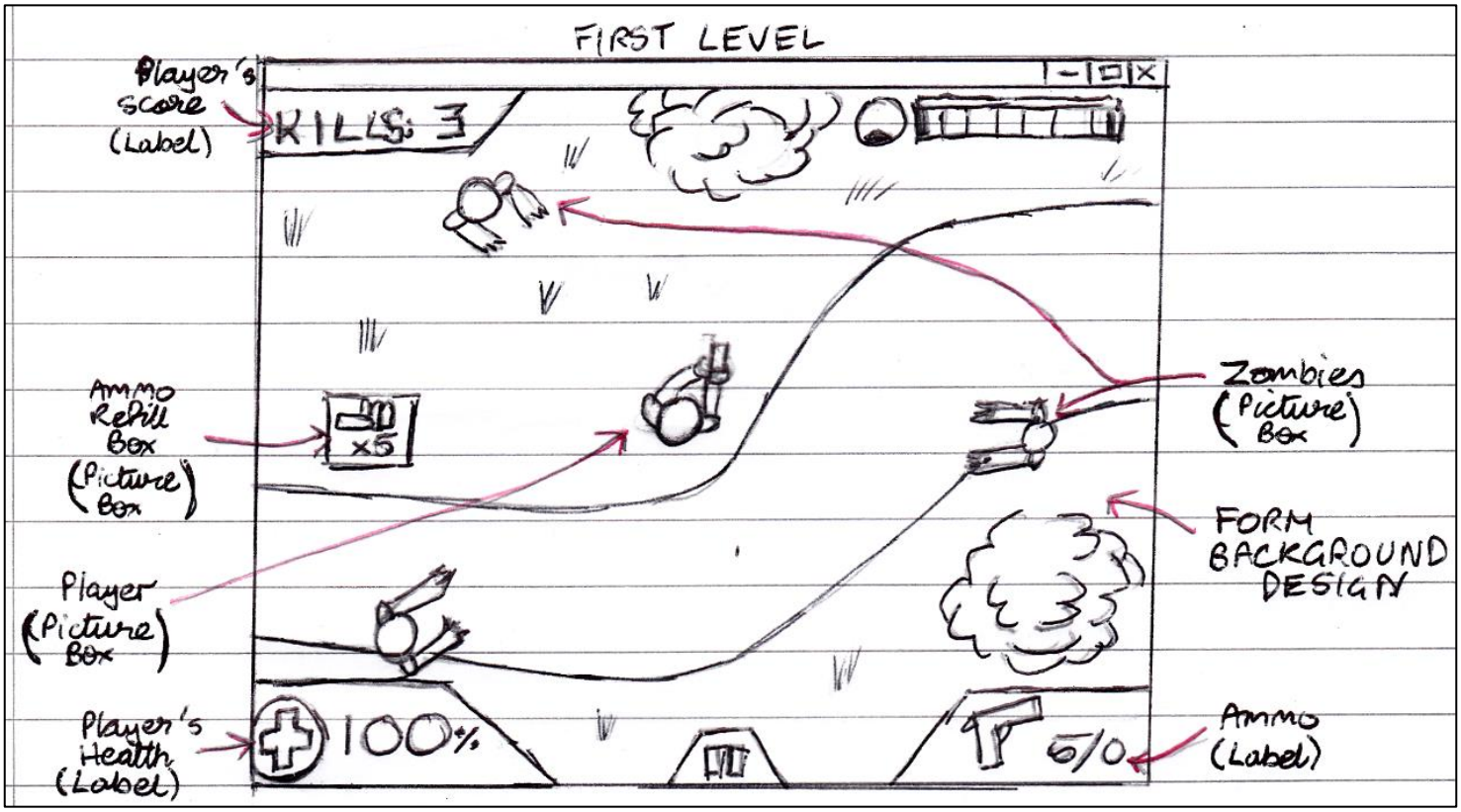
Login


LOGIN
Login Title (LABEL)
LOGIN
Username
Password
Textboxes
Error Providers
Login
Login Button checks if textboxes match existing members
Register
Help
Help page button
Main Menu button
Register page button

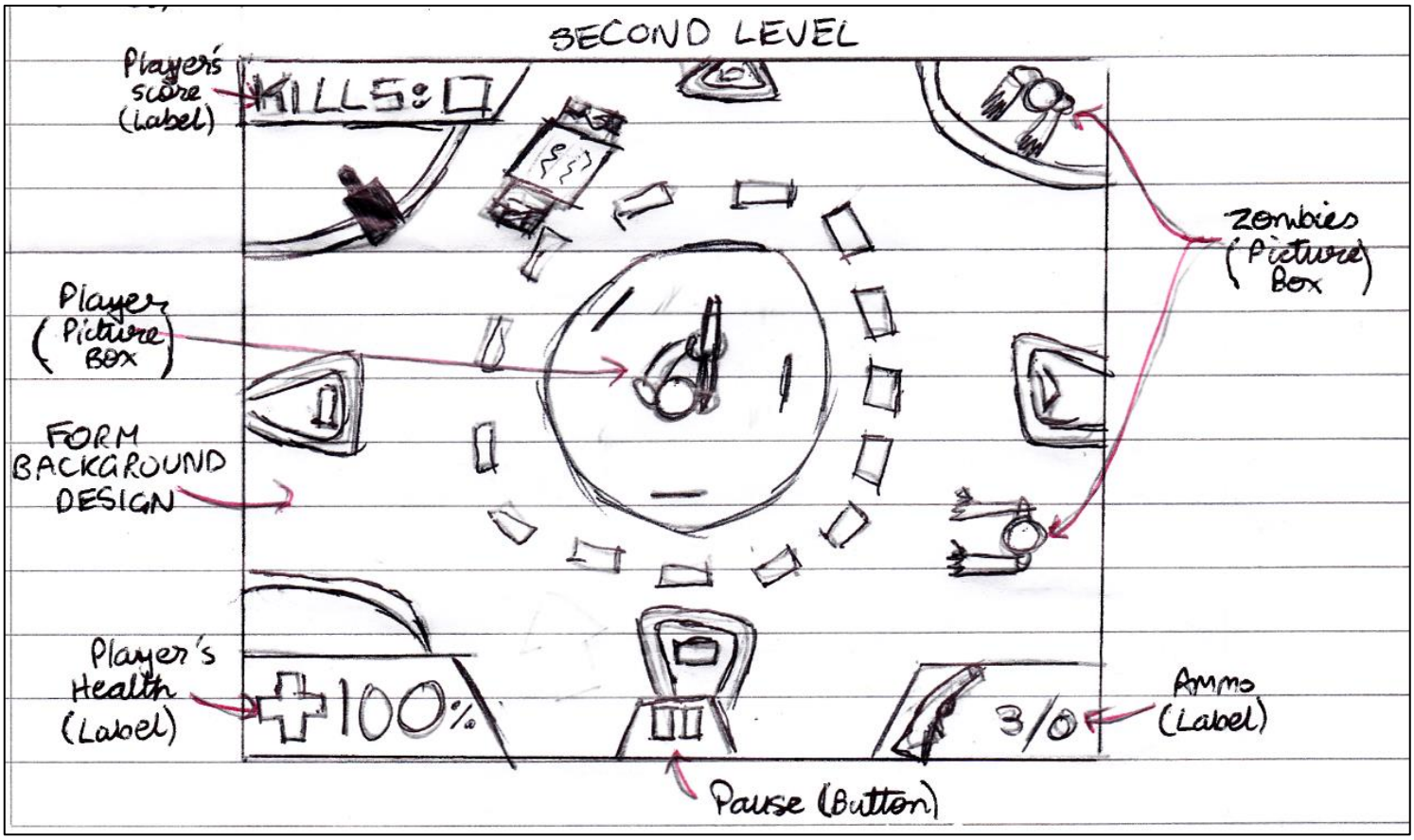
Help Guide



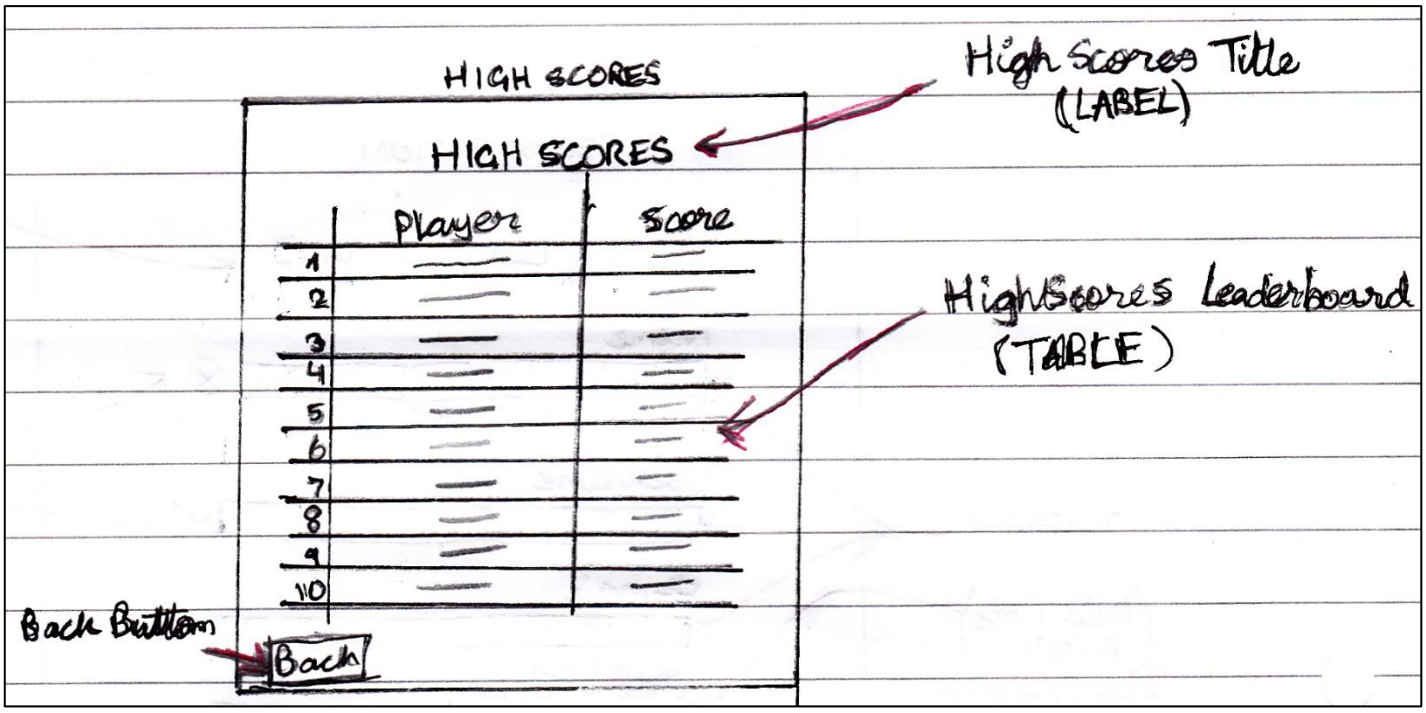
Game Level One



Game Level Two

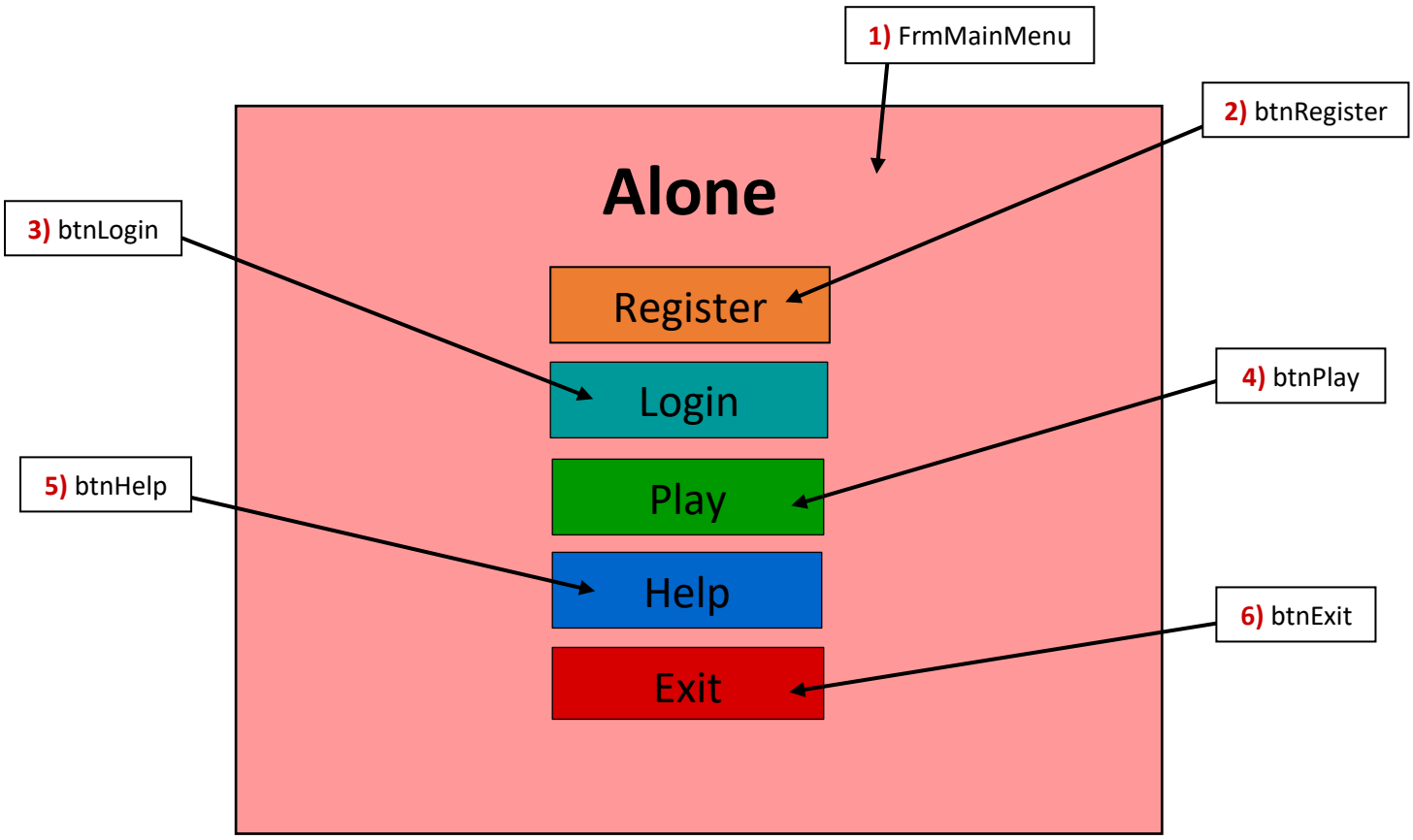


High Scores


HIGH SCORES
HIGH SCORES
High Scores Title
(LABEL)
Player
Score
1
2
3
4
5
6
7
8
9
10
HighScores Leaderboard
(TABLE)
Back Button
Back

Storyboard

Main Menu

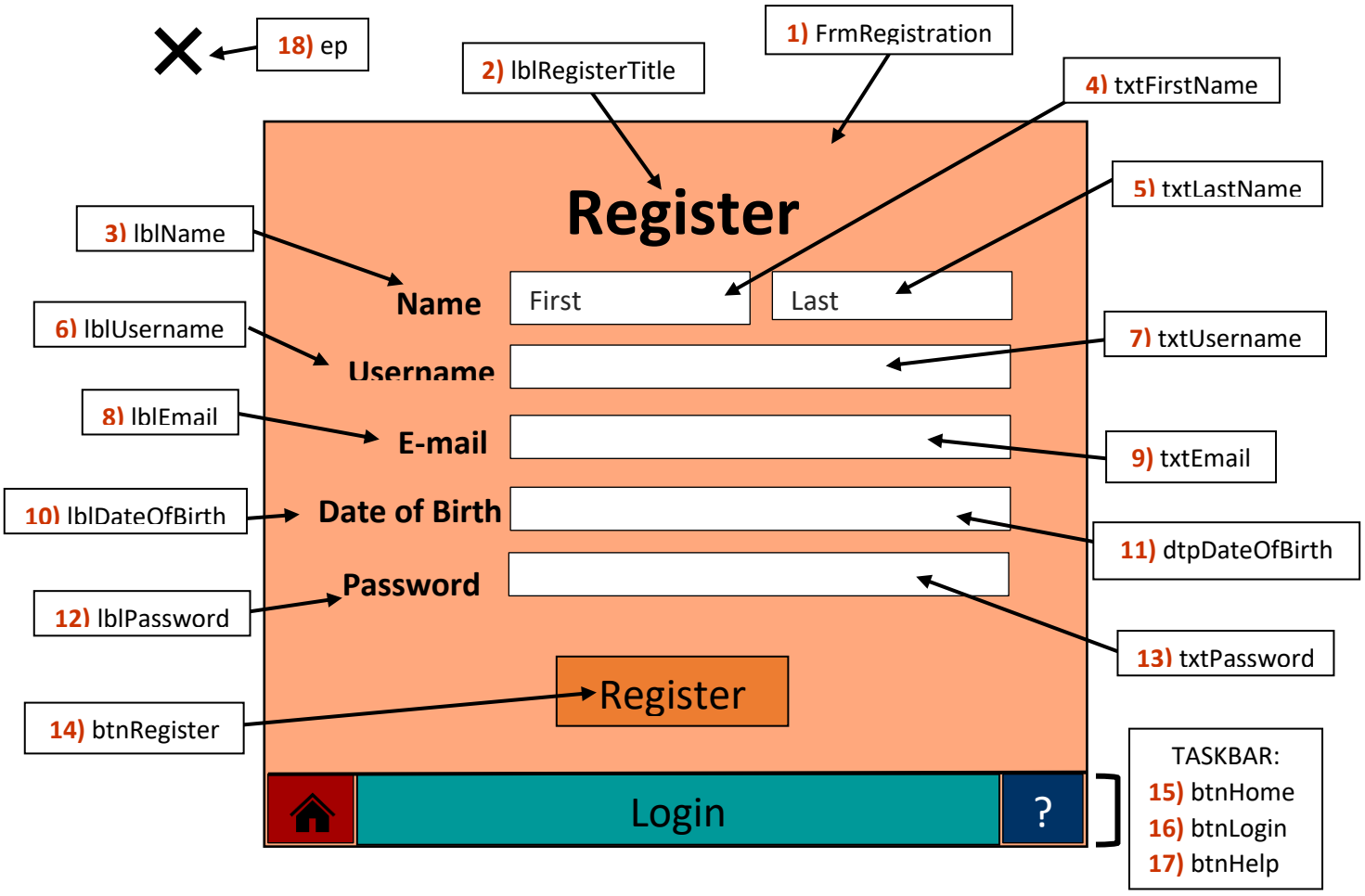


All the buttons on the Main Menu Form will use font 'Rockwell' and size 16pt.

1) **FrmMainMenu:** The Main Menu Screen will be the first form the user will see when the application is started.
2) **btnRegister:** This button will take the user to the Registration Form.
3) **btnLogin:** Pressing this button will take the user to the Login Form.
4) **btnPlay:** Pressing this button will change the Main Menu to the Game Level Select screen where the user can choose which level they want to play, or they can visit the high scores leader board to see where they are at. This button will only be clickable once the user is logged in, otherwise a tool tip will be displayed telling the user to log in when the hover over the button.
5) **btnHelp:** This is button will open the Help Guide which the user can read if they are having trouble.
6) **btnExit:** This button will simply exit the application completely.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmMainMenu Main Menu FixedSingle 830 x 640 (px) CentreScreen	N/A
Button	Name Text Cursor FlatStyle	btnRegister Register Hand Flat	BtnRegster_Click
Button	Name Text Cursor FlatStyle	btnLogin Login Hand Flat	BtnLogin_Click
Button	Name Text Cursor FlatStyle	btnPlay Play Hand Flat	BtnPlay_Click BtnPlay_MouseHover
Button	Name Text Cursor FlatStyle	btnHelp Help Hand Flat	BtnHelp_Click
Button	Name Text Cursor FlatStyle	btnExit Exit Hand Flat	BtnExit_Click

Registration



All the labels (except for lblRegisterTitle) and input fields will use font 'Maiandra GD' and size 13pt.

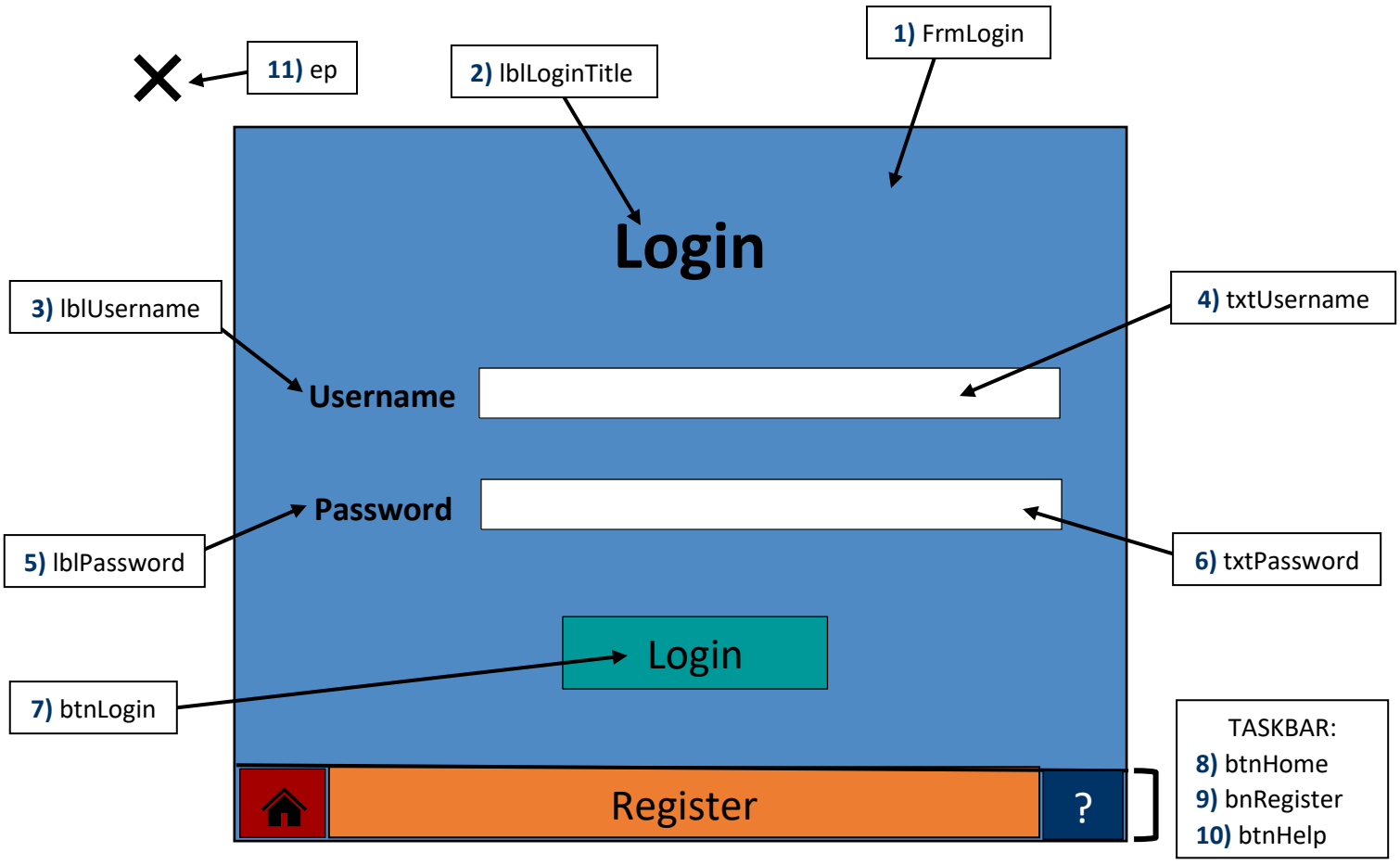
1) **FrmRegistration:** The Register Form will be the screen for unregistered users to create an account.
2) **lblRegisterTitle:** This label is only present for the user to easily identify that this is the Registration Form. It will have the bold font 'Palatino Linotype' and will use size 28pt.
3) **lblName:** This label indicates what the user must enter in the text field beside it, in this case their forename and surname must be entered.
4) **txtFirstName:** This text box will take the user's first name and will start with the default text "First" to guide the user.
5) **txtLastName:** This text box will take the user's last name and will start with the text "Last" to guide the user.
6) **lblUsername:** This label will help the user to enter a username in the adjacent text field.
7) **txtUsername:** This text box will take the user's desired username.
8) **lblEmail:** This label will tell the user to enter their e-mail address in the text field.
9) **txtEmail:** This text field will take the user's e-mail address.

10) **lblDateOfBirth:** This label will show the user that they have to input their date of birth in the next field.
11) **dtpDateOfBirth:** This date time picker will show a calendar for the user to easily pick their date of birth in a very interactive and easy fashion.
12) **lblPassword:** This label will indicate to the user to enter a password in the adjacent text box.
13) **txtPassword:** This text field will take the user's desired password. The text box will be masked, meaning that any text entered in it will be replaced by a single character (e.g. Asterisks). This will ensure that there is a level of security in the application, so the user's password will be concealed.
14) **btnRegister:** Pressing this button will attempt to register the user's credentials and will check if their input is valid. If any of the fields are invalid, errors will be displayed, and the user will not be registered.
15) **btnHome:** Pressing this will take the user back to the Main Menu Form.
16) **btnLogin:** Pressing this will take the user to the Login Form.
17) **btnHelp:** Pressing this will open the Help Guide.
18) **ep:** This error provider will display errors with captions beside the invalid fields.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmRegistration Registration FixedSingle 830 x 600 (px) CentreScreen	N/A
Label	Name Text	lblRegisterTitle Register	N/A
Label	Name Text	lblName Name	N/A
Text Box	Name Text	txtFirstName First	TxtFirstName_TextChanged
Text Box	Name Text	txtLastName Last	TxtLastName_TextChanged
Label	Name Text	lblUsername Username	N/A
Text Box	Name	txtUsername	TxtUsername_TextChanged
Label	Name Text	lblEmail E-mail	N/A
Text Box	Name	txtEmail	TxtEmail_TextChanged
Label	Name Text	lblDateOfBirth Date of Birth	N/A

Date Time Picker	Name	dtpDateOfBirth	DtpDateOfBirth_ValueChanged
Label	Name Text	lblPassword Password	N/A
Text Box	Name	txtPassword	TxtPassword_TextChanged
Button	Name Text Cursor FlatStyle	btnRegister Register Hand Flat	BtnRegister_Click
Error Provider	Name	ep	N/A
Button	Name Text Cursor FlatStyle	btnHome  Hand Flat	BtnHome_Click
Button	Name Text Cursor FlatStyle	btnLogin Login Hand Flat	BtnLogin_Click
Button	Name Text Cursor FlatStyle	btnHelp ? Hand Flat	BtnHelp_Click

Login

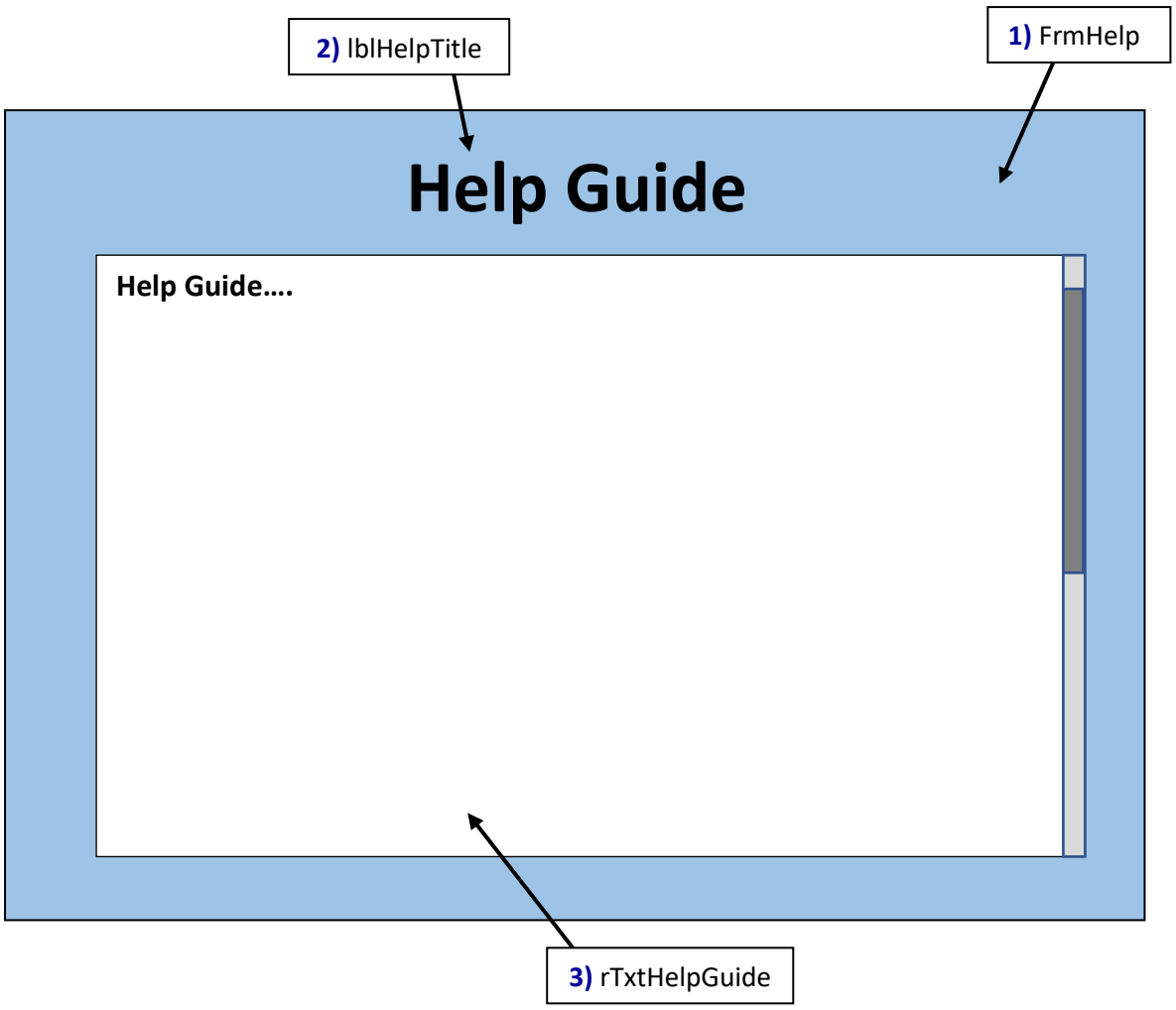


All the labels (except lblLoginTitle) and text boxes will use font 'Maiandra GD' and size 13pt.

1) **FrmLogin:** The Login Form will provide an interface for registered users to log in to start playing the game levels.
2) **lblLoginTitle:** This label will help the user to identify that they are on the Login Form. This label will use the bold font 'Palatino Linotype' and size 28pt.
3) **lblUsername:** This label will indicate that the user must enter their username in the adjacent text field.
4) **txtUsername:** This text box will take the user's username.
5) **lblPassword:** This label will indicate that the user must enter their password in the text box.
6) **txtPassword:** This text field will take the user's password and will be masked to create a secure and protected environment for the user to conceal their password from others.
7) **btnLogin:** Pressing this button will attempt to log the user in. In doing so, the text fields will be checked to see if the text entered matches existing members in the database. Errors will be placed if not.
8) **btnHome:** Pressing this button will return the user back to the Main Menu Form.
9) **btnRegister:** Pressing this button will take the user to the Registration Form.
10) **btnHome:** Pressing this will open the Help Guide.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmLogin Login FixedSingle 810 x 600 (px) CentreScreen	N/A
Label	Name Text	lblLoginTitle Login	N/A
Label	Name Text	lblUsername Username	N/A
Text Box	Name	txtUsername	N/A
Label	Name Text	lblPassword Password	N/A
Text Box	Name	txtPassword	N/A
Button	Name Text Cursor FlatStyle	btnLogin Login Hand Flat	BtnLogin_Click
Error Provider	Name	ep	N/A
Button	Name Text Cursor FlatStyle	btnHome  Hand Flat	BtnHome_Click
Button	Name Text Cursor FlatStyle	btnRegister Register Hand Flat	BtnRegister_Click
Button	Name Text Cursor FlatStyle	btnHelp ? Hand Flat	BtnHelp_Click

Help Guide

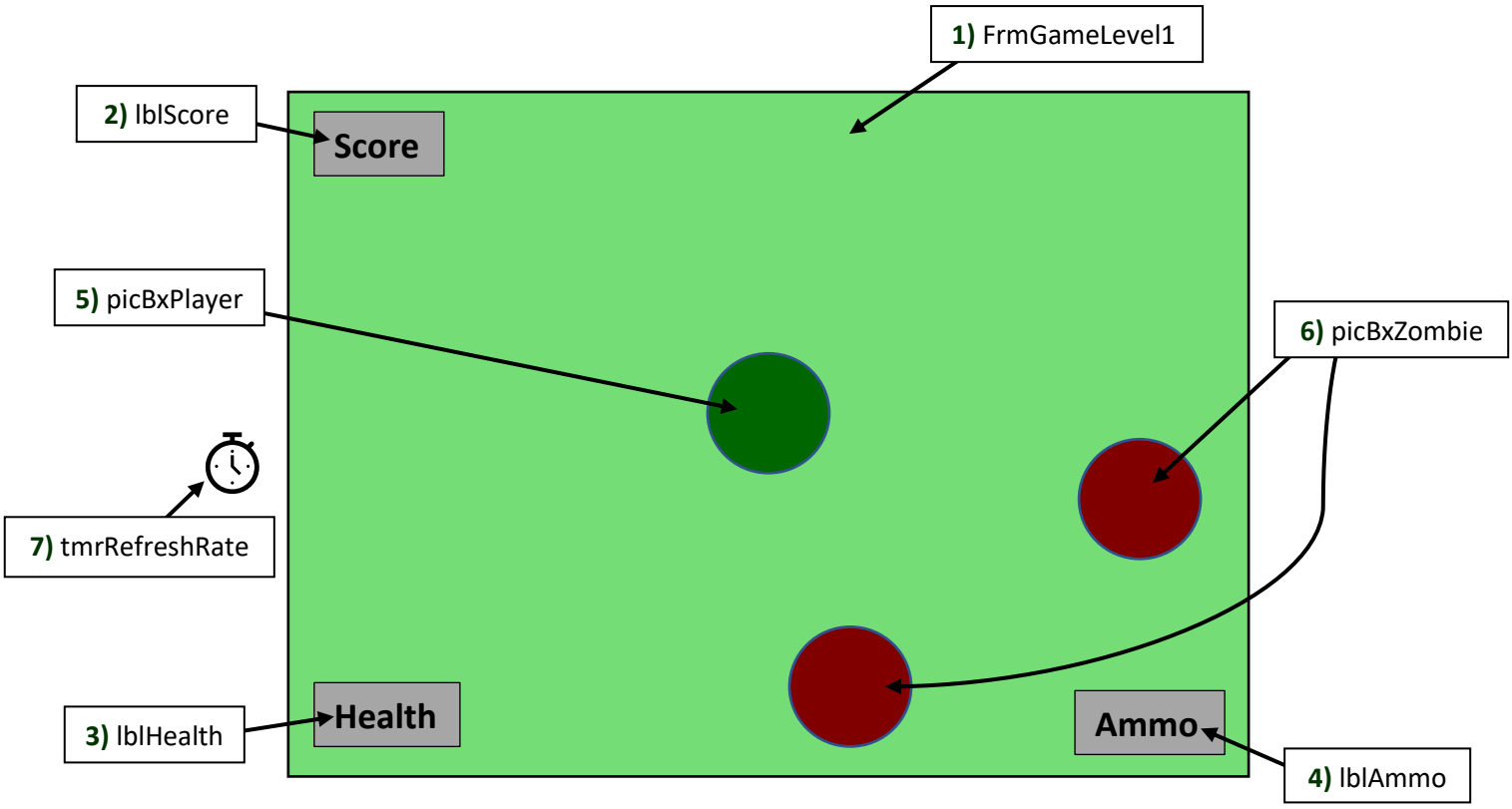


The label and rich text box will use the font 'Verdana' and size 12pt.

1) **FrmHelp:** The Help Guide will be the form that can be accessed from anywhere in the application to help the user when they are have difficulty getting through the application.
2) **lblHelpTitle:** This label is present to indicate that the user is on the Help Guide Form.
3) **rTxtHelpGuide:** This rich text box will hold the help guide text, so the user can read it. The text box will be scrollable in the vertical axis as an easy to use interface for the user.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmHelpGuide Help Guide FixedSingle 700 x 600(px) CentreScreen	N/A
Label	Name Text	lblHelpTitle Help Guide	N/A
Rich Text Box	Name Cursor ScrollBars ReadOnly WordWrap	rTxtHelpGuide IBeam Vertical True True	N/A

Game Level One

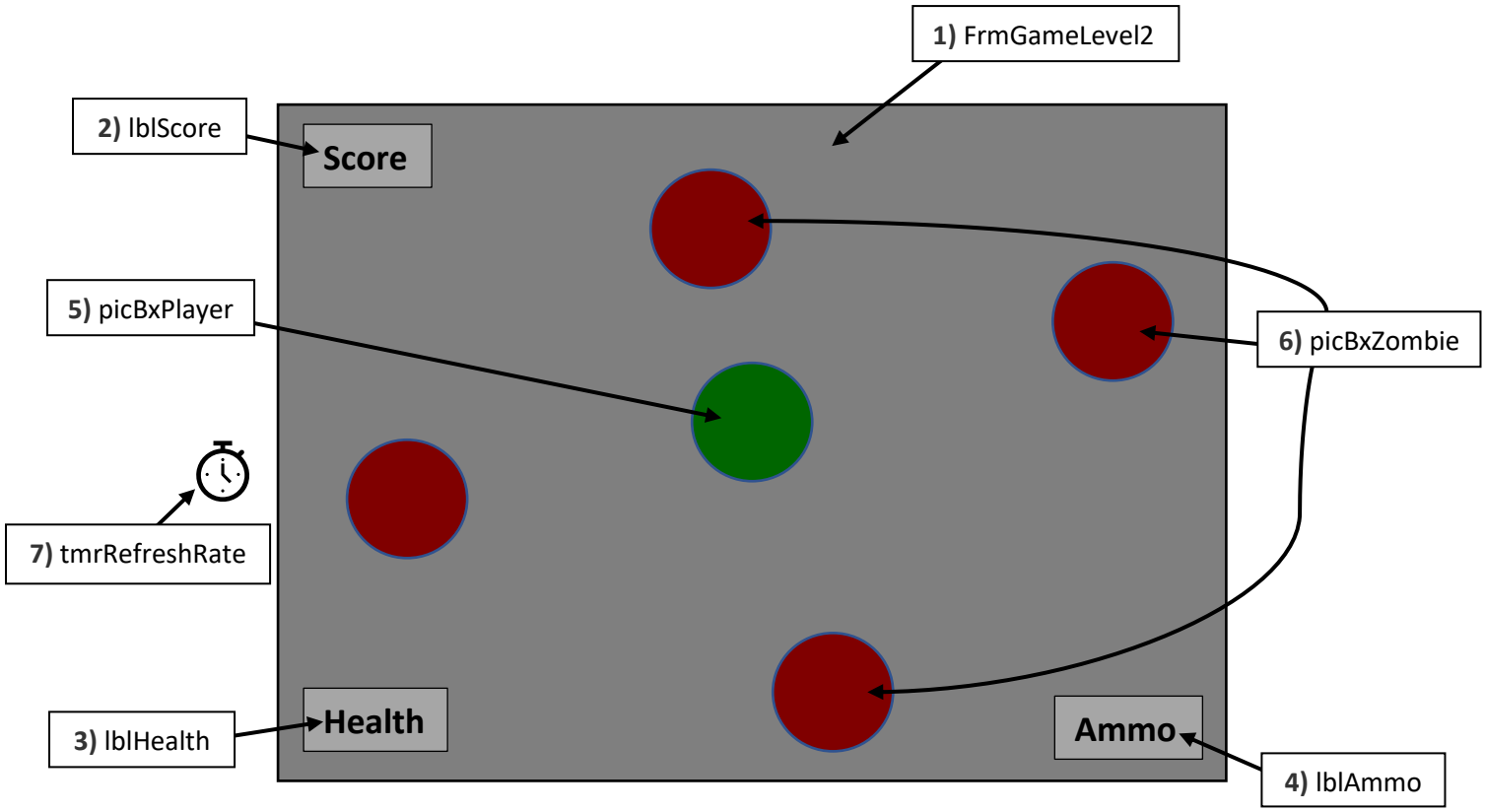


All text in the labels will use the font 'True Crimes' and size 15pt.

1) **FrmGameLevel1:** This will be the first level of the game. The level's background will act as the virtual play area that the player can move around in.
2) **lblScore:** This label will display the user's current score and will update as the user plays the game.
3) **lblHealth:** This label will display the user's current health and will update as the game progresses.
4) **lblAmmo:** This label will display the amount of ammo the player has in their disposal. It will be displayed in the format "'Ammo in the gun magazine' / 'Ammo in the ammo pouch'".
5) **picBxPlayer:** This picture box will display the player image that the user is in control of.
6) **picBxZombie:** This picture box will display the zombie images that will move towards the player.
7) **tmrRefreshRate:** This timer will act as the 'game engine' as every control on the form will update every time the timer ticks. The timer will be hidden from the user's view.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmGameLevel1 Alone | Level One FixedSingle 755 x 690(px) CentreScreen	N/A
Label	Name Text	lblScore Score: 0	N/A
Label	Name Text	lblHealth Health: 100%	N/A
Progress Bar	Name Value Maximum Minimum Style	prgsBrHealth 120 120 0 Continuous	N/A
Label	Name Text	lblAmmo Ammo: 7 / 3	N/A
Label	Name Text	lblPassword Password	N/A
Picture Box	Name	picBxPlayer	N/A
Picture Box	Name Tag	picBxZombie Zombie	N/A
Timer	Name Interval	tmrRefreshRate 1 (ms)	TmrRefreshRate_Tick

Game Level Two



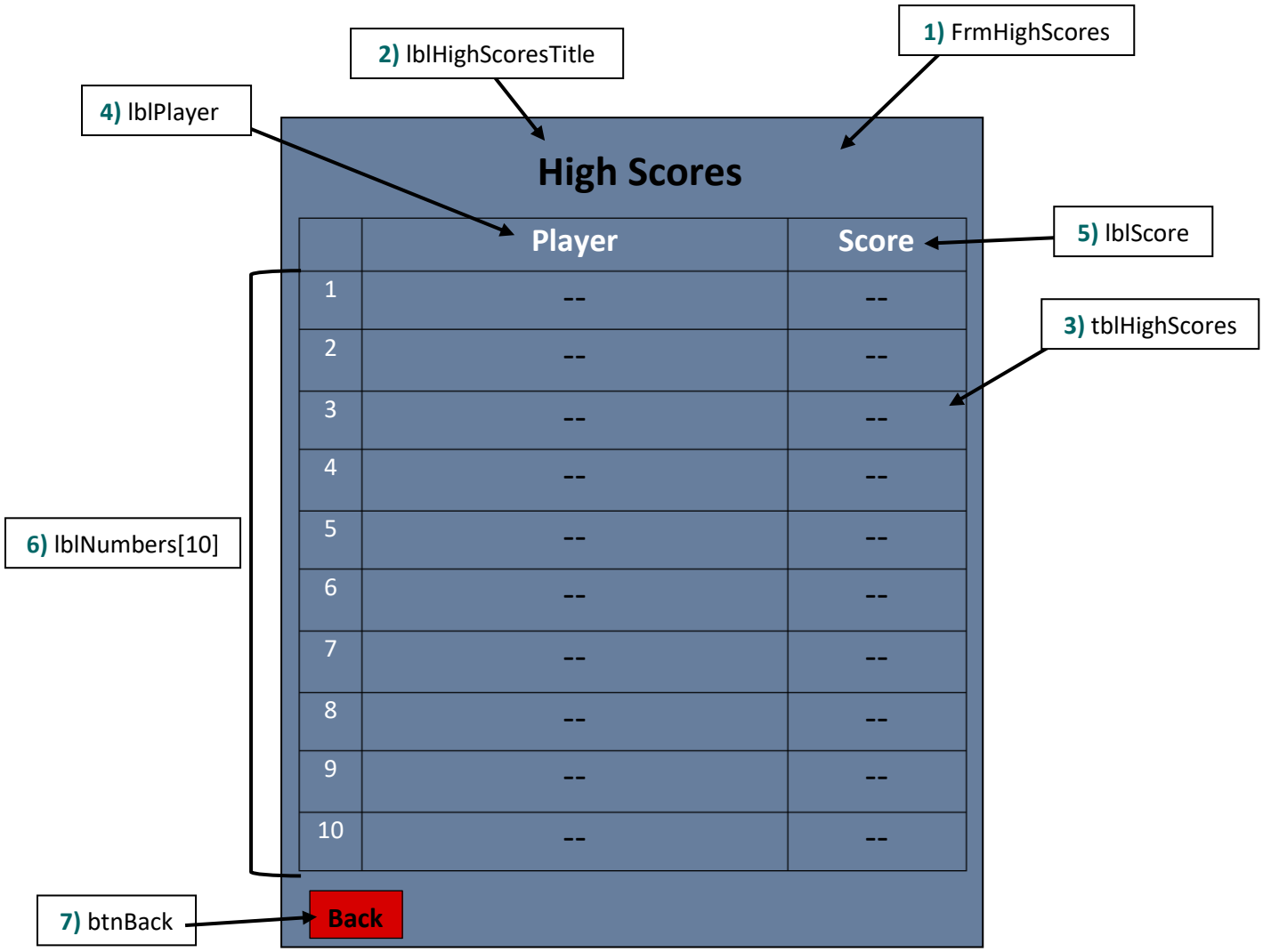
All text in the labels will use the font 'True Crimes' and size 15pt.

1) **FrmGameLevel2:** This will be the second level of the game. The level's background will act as the virtual play area that the player can move around in. The screen for this level will be bigger in size than the first level to offer more space for the player to utilise and strategize.
2) **lblScore:** This label will display the user's current score and will update as the user plays the game.
3) **lblHealth:** This label will display the user's current health and will update as the game progresses.
4) **lblAmmo:** This label will display the amount of ammo the player has in their disposal. It will be displayed in the format "'Ammo in the gun magazine' / 'Ammo in the ammo pouch'".
5) **picBxPlayer:** This picture box will display the player image that the user is in control of.
6) **picBxZombie:** This picture box will display the zombie images that will move towards the player.
7) **tmrRefreshRate:** This timer will act as the 'game engine' as every control on the form will update every time the timer ticks. The timer will be hidden from the user's view.

The second level will offer a different weapon compared to the first level that can hold more bullets in its magazine. This will encourage new ways to use the gun for the user making the gameplay refreshing and different.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmGameLevel2 Alone | Level Two FixedSingle 850 x 750(px) CentreScreen	N/A
Label	Name Text	lblScore Score: 0	N/A
Label	Name Text	lblHealth Health: 100%	N/A
Progress Bar	Name Value Maximum Minimum Style	prgsBrHealth 120 120 0 Continuous	N/A
Label	Name Text	lblAmmo Ammo: 12 / 8	N/A
Label	Name Text	lblPassword Password	N/A
Picture Box	Name	picBxPlayer	N/A
Picture Box	Name Tag	picBxZombie Zombie	N/A
Timer	Name Interval	tmrRefreshRate 1 (ms)	TmrRefreshRate_Tick

High Scores



All the labels will use the font 'Palatino Linotype' and size 18pt.

1) **FrmHighScores:** The High Scores Form will show the leader board where the top ten players with the highest scores are displayed in descending order down the list.
2) **lblHighScoresTitle:** This label is present to indicate to the user that they are on the High Scores Table.
3) **tblHighScores:** This table layout panel will be a 3 by 11 table which will hold all the labels (bar lblHIghScoresTitle).
4) **lblPlayer:** This label will specify the column holding all the usernames of the top ten players.
5) **lblScore:** This label will specify the column holding all the high scores in descending order.
6) **lblNumbers[10]:** This will be a label array to hold each of the number labels in the first column.
7) **btnBack:** Pressing this button will return the user back to the Game Level Select screen.

Control	Properties	Value	Events
Form	Name Text FormBorderStyle Size StartPosition	FrmHighScores High Scores FixedSingle 600 x 695(px) CentreScreen	N/A
Label	Name Text	lblHighScoresTitle High Scores	N/A
Table Layout Panel	Name CellBorderStyle ColumnCount RowCount Size	tblHighScores OutsetPartial 3 11 510 x 520(px)	N/A
Label	Name Text Cell Anchor	lblPlayer Player 1, 0 None	N/A
Label	Name Text Cell Anchor	lblScore Score 2, 0 None	N/A
Label[10] (Array)	Text (Each Label) Anchor	(Numbers 1 – 10) None	N/A
Button	Name Text Cursor FlatStyle	btnBack Back Hand Flat	BtnBack_Click

Pseudo Code

Pseudo Code

Main Menu

Set a public static Boolean variable called 'loggedIn' to false. This will be a global variable that will determine if the user is logged in.

//Initialise Form:
If loggedIn equals true:
    Then enable the play button (i.e. Change its forecolour to green and back-colour to black).

//Register Button Click Event (BtnRegister_Click):
Go to Register Form, closing current form.

//Login Button Click Event (BtnLogin_Click):
Go to Login Form, closing current form

//Play Button Click Event (BtnPlay_Click):
If loggedIn equals true:
    Clear the controls from the main menu form and create and add three buttons (btnLevel1, btnLevel2 & btnBack) to the form.
    The buttons will take the user to the two levels, while the back button takes the user back to the main menu screen. (Level Select Screen replaces main menu).

//Play Button Hover Event (BtnPlay_MouseHover):
If loggedIn is equal to false:
    Place a tool tip over the button with a caption telling the user to log in/register first to start playing the game.

//Level One Button Click Event (BtnLevel1_Click):
Go to Game Level One Form, closing current form.

//Level One Button Hover Event (BtnLevel1_MouseHover):
Place a tool tip over the button with a caption showing the user information about level one.

//Level Two Button Click Event (BtnLevel2_Click):
Go to Game Level Two Form, closing current form.

//Level Two Button Hover Event (BtnLevel2_MouseHover):
Place a tool tip over the button with a caption showing the user information about level two.

//Back Button Click Event (BtnBack_Click):
Go back to the Main Menu Form with the enabled (Green) play button.

//Help Button Click Event (BtnHelp_Click):
Open Help Form, keeping current form active.

//Exit Button Click Event (BtnExit_Click):
Exit application.

Registration

Instantiate a public static array of type Member called 'members' with 50 spaces.
Instantiate a 'member' object.

//Register Button Click Event (BtnRegister_Click):
If the current user input in the input fields matches the latest valid input assigned to the member:
    Then call member's 'ReadFromFile' method to populate array if any already registered users are present in the binary file.
    Integer 'i' equals 0.
    While i is less than members array length:
        If member object at index i of the array is null:
            Then make this element equal to the current member and break. (i.e. Adds the new registered member to the first null element in the members array).
        Else
            Increment i.
    Integer 'numOfMembers' is equal to i + 1.
    Write the member to the file by calling the class's 'WriteToFile' function with numOfMembers as the parameter.

    For each input field, place a tick icon as the error provider with caption "Valid".
    Show message box to welcome the new member and notify the user that they have successfully registered an account and can proceed to the login page.
    Call login button click event to go to login form.
Else:
    Check each field and place error providers where the fields are invalid. (Call Check methods for each input field)

//Home Button Click Event (BtnHome_Click):
Go to Main Menu Form, closing current form.

//Login Button Click Event (BtnLogin_Click):
Go to Login Form, closing current form.

//Help Button Click Event (BtnHelp_Click):
Open Help Form, keeping current form open.

//First Name Textbox Text Changed Event (TxtFirstName_TextChanged):
Call Check first name method.

//Last Name Textbox Text Changed Event (TxtLastName_TextChanged):
Call Check last name method.

//Username Textbox Text Changed Event (TxtUsername_TextChanged):
Call Check username method.

//Email Textbox Text Changed Event (TxtEmail_TextChanged):
Call Check e-mail method.

//Date of Birth Value Changed Event (DtpDateOfBirth_ValueChanged):
Call Check date of birth method.

//Password Textbox Text Changed Event (TxtPassword_TextChanged):
Call Check password method.

//Check for valid input methods:
Each method will try to assign the input in each field to the member class properties. If it fails (i.e. it is invalid), then an exception will be thrown and when caught as a CustomException, error providers will be assigned where necessary with the error message from the CustomException.
Finally:

If the input matches the latest valid input:
Then nullify the any error provider present for the that field.

Login

Instantiate a public static 'currentMember' object that will hold the currently logged in user details.

//Login Button Click Event (BtnLogin_Click):
If username or password textboxes are not empty:
Then read from the file using the class function.
For each member object in the members list in the Registration Form:
If the current member object is null:
Then break;
If the username of the member equals the text in the username textbox:
Then set the 'currentMember' equal to that member object in list. (Save matching member).
Nullify the error on the username textbox and break loop.
Else:
Set error on the username textbox telling the user that the username entered does not exist in the database.

If the password from the (saved) 'currentMember' equals the text in the password textbox:

Then change the error provider's icon to a tick.
For each textbox control on the form, set error (tick) to each of them.
Show a message box notifying the user that they have successfully logged in.
Close current form and set loggedIn from the Main Menu Form to true.
Call the BtnHome_Click event (Go to Main Menu).

Else:

Then set error on the control telling the user that it is incorrect.

//Home Button Click Event (BtnHome_Click):
Go to Main Menu Form, closing current form.

//Register Button Click Event (BtnRegister_Click):
Go to Registration Form, closing current form.

//Help Button Click Event (BtnHelp_Click):
Open Help Form, keeping current form active in background.

Help Guide

//Initialise Form:
Instantiate help guide array of type string to hold all lines from the help guide text file.
For loop to loop through the number of lines in the text file (or the length of the array) and set the rich text box's text to the text in the array followed by a new line.

Game Level One

//Initialise Form:
Instantiate a 'player' object with a speed of 12px/ms, a clip with 7 bullets, 3 in ammo pouch, a maximum clip size of 7, a point at the centre of the form, game level 1 and the form as parameters.

Set the current logged in member's score to 0.

Instantiate a 'zombie' object with a speed of 3px/ms, can deal damage of 1hp and appropriate size for form.
Call the zombie GenerateZombie function to generate 2 zombies and on the form and, as parameter, call the Zombie_RndSpawnPoint method.

Integer 'x' and 'y' to store the player's x and y co-ordinate to make it easily accessible.
Point 'gunPosition' to store the point where the bullet must be fired from according to where the player is facing.

//Paused and Game Over Controls

Instantiate button 'btnHome' with text "Home", FlatStyle as Flat, Cursor as Hand and visible as false.
Instantiate button 'btnHelp' with text "?", FlatStyle as Flat, Cursor as Hand and visible as false.
Instantiate label 'lblGameOver' with text "Game Over", BackColour as Maroon and visible as false.
Instantiate button 'btnPlayAgain' with text "Play Again?", FlatStyle as Flat, Cursor as Hand and visible as false.

//Zombie_RndSpawnPoint Method:

Two integers are randomly generated within the range of the size of the form.
While always true:
    If they do not follow the rules (i.e. if on top of objects in background/controls):
        Then generate new random number.
    Else:
        Break from loop.
Place the two integers in an array with 2 spaces and return the array.

//Form Key Down Event (FrmGameLevel1_KeyDown):

//Movement

If up arrow or 'W' key is pressed:
    Then call player's 'Up' method and set new gunPosition.
If left arrow or 'A' key is pressed:
    Then call player's 'Left' method and set new gunPosition.
If down arrow or 'S' key is pressed:
    Then call player's 'Down' method and set new gunPosition.
If right arrow or 'D' key is pressed:
    Then call player's 'Right' method and set new gunPosition.

//Reload

If 'R' key is pressed and player's clip is not full and the ammo in player's ammo pouch is not equal to 0 and player's reload Boolean is false (Check to see player is not already reloading):
    Then call player's Reload method and play reload sound effect.

//Form Key Up Event (FrmGameLevel1_KeyReleased):

If up arrow or 'W' key is pressed:
    Then set player's 'up' Boolean to false.
If left arrow or 'A' key is pressed:
    Then set player's 'left' Boolean to false.
If down arrow or 'S' key is pressed:
    Then set player's 'down' Boolean to false.
If right arrow or 'D' key is pressed:
    Then set player's 'right' Boolean to false.

//Pause

If 'P' key is pressed:

Then pause game (Stop any timers and freeze the game) and make visible true for main menu and help buttons.

Else (Any other key is pressed):

Then un-pause/unfreeze game (Restart any timers) and make visible property to false for the buttons.

//Fire Weapon (In key release so user cannot hold space bar down to continuously fire)

If 'Space Bar' Key is pressed and player reload Boolean is false:

If player's gun clip is not empty:

Then call 'Fire' method with player's direction as parameter.

If player's clip and ammo pouch is empty (No ammo):

Then call 'AmmoDrop' method.

Else (Player's clip is empty):

Play empty gunshot sound effect.

//Home Button Click Event (BtnHome_Click):

Save the logged in currentMember's score by calling the WriteToFile method with currentMember's 'CountMembers' method as parameter.

Go to Main Menu Form, closing current form.

//Help Button Click Event (BtnHelp_Click):

Open Help Form, keeping the current form open.

//Restart Button Click Event (BtnPlayAgain_Click):

Open Game Level One Form, closing current form.

//Timer Tick Event (TmrRefreshRate_Tick):

//Score

Update the score label to the logged in currentMember's current score.

//Health

Update the health progress bar to player's current health.

If player's health is less than 20:

Then change colour of progress bar to dark red.

Else:

Then default colour of progress bar is forest green.

If player's health is less than or equal to 0:

Then GAME OVER (Stop timer, make visible property true for game over controls to show).

Save the logged in currentMember's score by calling the WriteToFile method with currentMember's 'CountMembers' method as parameter.

Else:

Make the progress bar's value equal to player's current health.

//Ammunition

Update the ammo information label to player's current amount of ammo left in the gun clip and ammo pouch.
If ammo in gun clip is not equal to 0 and less than or equal to 2:
    Then change label colour to dark orange.
Else if ammo in gun clip is equal to 0:
    Then change label colour to dark red.
Else:
    Label default colour is white.

//Movement
If player's up is true and no invisible wall above player (InvisibleWall_Above method returns false):
    Then move player's y co-ordinate up by player speed.
If player's left is true and no invisible wall to the left of player (InvisibleWall_Left method returns false):
    Then move player's x co-ordinate left by player speed.
If player's down is true and no invisible wall below player (InvisibleWall_Below method returns false):
    Then move player's y co-ordinate down by player speed.
If player's right is true and no invisible wall to the right of player (InvisibleWall_Right method returns false):
    Then move player's x co-ordinate right by player speed.

Set the player picture box's location to a new point with the x and y.

//Reloading
If player's reload Boolean is true:
    Place a progress bar above the player (relative to x and y so it will move with player) and follow its movement displaying reload time delay.

//Collisions and Zombie Movement
Foreach loop to loop through each control in the form:
    //Ammunition Refill Box Collisions
    If control is a picture box and its tag is "Ammo Box" and this control collides with the player picture box:
        Then player's ammo pouch will increase by the maximum clip size.

    //Zombie Collisions and Movement
    If control is a picture box and its tag is "Zombie":
        If this control collides with the player picture box:
            Then zombie will damage the player (Decreases their health by zombie's damage amount).

        If bullet object is not null and this control collides with the bullet picture box.
            Then kill zombie and remove bullet and increment logged in member's score. Integer 'numOfZ' is set to 1.

            If player's score equals 50:

Then increase difficulty (Increase zombie speed, damage, numOfZ is set to 3 [Only for one instance], load second zombie picture) and increase player's health by 60hp.

Call zombie's 'GenerateZombie' method with the form, numOfZ and Zombie_RndSpawnPoint method as parameters.
Play zombie sound effect.

Call zombie's 'ZombieMovement' method with the current control in loop, the player object and all 4 InvisibleWall methods as parameters.

//InvisibleWall Methods:

Methods to check if player's location crosses the co-ordinates of the background objects and edges of the form. The methods will return a Boolean value. The invisible boundary methods should be separated into the 4 directions (Above, Left, Below, Right). Takes two integer parameters (x, y).

//Fire Method:

Decrement player's clip.
Instantiate bullet object with a speed of 30px/ms, can deal damage of 100hp. The direction the player is facing is passed into the constructor.
Call bullet's 'GenerateBullet' method with the form, size of bullet picture box, gunPosition and 'BulletCollisions' method as parameters.
Play the gun shot sound effect for the Desert Eagle pistol.

//BulletCollisions Method:

If the co-ordinates of the bullet picture box reaches the edges of the form or objects/controls on the form:

Then dispose the bullet picture box and bullet's timer.

//AmmoDrop Method:

While always true:

Two integers are randomly generated within the range of the size of the form.
While always true:

If they do not follow the rules (i.e. if on top of objects in background/controls):

Then generate new random number.

Else:

Break from loop.

Instantiate picture box 'picBxAmmoBox' with its image as the first ammunition refill box image, its tag as "Ammo Box", visible as false, its location as the random co-ordinate.

Integer 'bonus_ammoBox' equal to a random integer between 1 and 10.
If bonus_ammoBox is not equal to 1:

Then break from outer loop. (1 in 10 chance of receiving bonus ammo boxes).

Game Level Two

//Initialise Form:

Instantiate a 'player' object with a speed of 13px/ms, a clip with 12 bullets, 8 in ammo pouch, a maximum clip size of 12, a point at the centre of the form, game level 2 and the form as parameters.

Set the current logged in currentMember's score to 0.

Instantiate a 'zombie' object with a speed of 4px/ms, can deal damage of 2hp and appropriate size for form.
Call the zombie GenerateZombie function to generate 4 zombies and on the form and, as parameter, call the Zombie_RndSpawnPoint method.

Integer 'x' and 'y' to store the player's x and y co-ordinate to make it easily accessible.
Point 'gunPosition' to store the point where the bullet must be fired from according to where the player is facing.

//Paused and Game Over Controls

Instantiate button 'btnHome' with text "Home", FlatStyle as Flat, Cursor as Hand and visible as false.
Instantiate button 'btnHelp' with text "?", FlatStyle as Flat, Cursor as Hand and visible as false.
Instantiate label 'lblGameOver' with text "Game Over", BackColour as Maroon and visible as false.
Instantiate button 'btnPlayAgain' with text "Play Again?", FlatStyle as Flat, Cursor as Hand and visible as false.

//Zombie_RndSpawnPoint Method:

Two integers are randomly generated within the range of the size of the form.
While always true:
    If they do not follow the rules (i.e. if on top of objects in background/controls):
        Then generate new random number.
    Else:
        Break from loop.
Place the two integers in an array with 2 spaces and return the array.

//Form Key Down Event (FrmGameLevel2_KeyDown):

//Movement

If up arrow or 'W' key is pressed:
    Then call player's 'Up' method and set new gunPosition.
left arrow or 'A' key is pressed:
    Then call player's 'Left' method and set new gunPosition.
If down arrow or 'S' key is pressed:
    Then call player's 'Down' method and set new gunPosition.
If right arrow or 'D' key is pressed:
    Then call player's 'Right' method and set new gunPosition.

//Reload

If 'R' key is pressed and player's clip is not full and the ammo in player's ammo pouch is not equal to 0 and player's reload Boolean is false (Check to see player is not already reloading):

Then call player's Reload method and play reload sound effect.

//Form Key Up Event (FrmGameLevel2_KeyReleased):

If up arrow or 'W' key is pressed:

Then set player's 'up' Boolean to false.

If left arrow or 'A' key is pressed:

Then set player's 'left' Boolean to false.

If down arrow or 'S' key is pressed:

Then set player's 'down' Boolean to false.

If right arrow or 'D' key is pressed:

Then set player's 'right' Boolean to false.

//Pause

If 'P' key is pressed:

Then pause game (Stop any timers and freeze the game) and make visible true for main menu and help buttons.

Else (Any other key is pressed):

Then un-pause/unfreeze game (Restart any timers) and make visible property to false for the buttons.

//Fire Weapon (In key release so user cannot hold space bar down to continuously fire)

If 'Space Bar' Key is pressed and player reload Boolean is false:

If player's gun clip is not empty:

Then call 'Fire' method with player's direction as parameter.

If player's clip and ammo pouch is empty (No ammo):

Then call 'AmmoDrop' method.

Else (Player's clip is empty):

Play empty gunshot sound effect.

//Home Button Click Event (BtnHome_Click):

Save the logged in currentMember's score by calling the WriteToFile method with currentMember's 'CountMembers' method as parameter.

Go to Main Menu Form, closing current form.

//Help Button Click Event (BtnHelp_Click):

Open Help Form, keeping the current form open.

//Restart Button Click Event (BtnPlayAgain_Click):

Open Game Level Two Form, closing current form.

//Timer Tick Event (tmrRefreshRate_Tick):

//Score

Update the score label to the currentMember's current score.

//Health

Update the health progress bar to player's current health.
If player's health is less than 20:
    Then change colour of progress bar to red.
Else:
    Then default colour of progress bar is forest green.

If player's health is less than or equal to 0:
    Then GAME OVER (Stop timer, make visible property true for game over controls to show).
    Save the logged in currentMember's score by calling the WriteToFile method with currentMember's 'CountMembers' method as parameter.
Else:
    Make the progress bar's value equal to player's current health.

//Ammunition

Update the ammo information label to player's current amount of ammo left in the gun clip and ammo pouch.
If ammo in gun clip is not equal to 0 and less than or equal to 5:
    Then change label colour to orange.
Else if ammo in gun clip is equal to 0:
    Then change label colour to red.
Else:
    Label default colour is white.

//Movement

If player's up is true and no invisible wall above player (InvisibleWall_Above method returns false):
    Then move player's y co-ordinate up by player speed.
If player's left is true and no invisible wall to the left of player (InvisibleWall_Left method returns false):
    Then move player's x co-ordinate left by player speed.
If player's down is true and no invisible wall below player (InvisibleWall_Below method returns false):
    Then move player's y co-ordinate down by player speed.
If player's right is true and no invisible wall to the right of player (InvisibleWall_Right method returns false):
    Then move player's x co-ordinate right by player speed.

Set the player picture box's location to a new point with the x and y.

//Reloading

If player's reload Boolean is true:

Place a progress bar above the player (relative to x and y so it will move with player) and follow its movement displaying reload time delay.

//Collisions and Zombie Movement

Foreach loop to loop through each control in the form:

//Ammunition Refill Box Collisions

If control is a picture box and its tag is "Ammo Box" and this control collides with the player picture box:

Then player will collect the amount of ammo in the box (Transfer to their ammo pouch).

//Zombie Collisions and Movement

If control is a picture box and its tag is "Zombie":

If bullet object is not null and this control collides with the player picture box:

Then zombie will damage the player (Decreases their health by damage amount).

If this control collides with the bullet picture box.

Then decrease zombie's health by the damage the bullet deals and dispose of the bullet picture box.
Integer 'numOfZ' is set to 1.

If player's score reaches 20:

Increase level difficulty (Increase zombie speed, numOfZ is set to 2 [Only for one instance]) and increases player's health by 80hp.

If zombie's health reaches 0:

Then increment player's score and remove zombie from the screen.

(Player will get small health boost for every kill, will depend on score).
If player's score is greater than or equal to 20:

Give health boost to player by 4hp.

Else:

Give health boost to player by 2hp.

Call zombie's 'GenerateZombie' method with the form, numOfZ and Zombie_RndSpawnPoint method as parameters.
Play zombie sound effect.

Call zombie's 'ZombieMovement' method with the current control in loop, the player object and all 4 InvisibleWall methods as parameters.

//InvisibleWall Methods:

Methods to check if player's location crosses the co-ordinates of the background objects and edges of the form. The methods will return a Boolean value. The invisible boundary methods should be separated into the 4 directions (Above, Left, Below, Right). Takes two integer parameters (x, y).

//Fire Method:

Decrement player's clip.
Instantiate bullet object with a speed of 50px/ms, can deal damage of 50hp. The direction the player is facing is passed into the constructor.
Call bullet's 'GenerateBullet' method with the form, size of bullet picture box, gunPosition and 'BulletCollisions' method as parameters.
Play the gun shot sound effect for the Nova shotgun.

//BulletCollisions Method:

If the co-ordinates of the bullet picture box reaches the edges of the form or objects/controls on the form:
    Then dispose the bullet picture box and bullet's timer.

//AmmoDrop Method:

While always true:
    Two integers are randomly generated within the range of the size of the form.
    While always true:
        If they do not follow the rules (i.e. if on top of objects in background/controls):
            Then generate new random number.
        Else:
            Break from loop.
    Instantiate picture box 'picBxAmmoBox' with its image as the first ammunition refill box image, its tag as "Ammo Box", visible as false, its location as the random co-ordinate.

    Integer 'bonus_ammoBox' equal to a random integer between 1 and 30.
    If bonus_ammoBox is not equal to 1:
        Then break from outer loop. (1 in 30 chance of receiving bonus ammo boxes).

High Scores

//Initialise Form:

Instantiate a 'highScores' integer array with 50 spaces to hold all 50 members' high score.
For loop to loop through highScores array (int 'i'):
    If the member object at index i is null:
        Then break from loop.

    Set element 'i' of highScores array to the member object's high score property.

Sort highScores in ascending order then reverse it to put it in descending order.
Instantiate a 'topTenHighScores' array with 10 spaces and place a copy of the first 10 elements in highScores array into the topTenHighScores array.

Instantiate a two-dimensional 10 by 10 label array called 'lblInfo'.

Instantiate a string array 'usernames' with 10 spaces.
For loop to loop through usernames array (int 'i'):
    For loop to loop through FrmRegistration's member's array (int 'j'):
        If the member object at index j is null:
            Then break from inner loop.

        If the member object's high score at index j matches the integer in the topTenHighScores array at index i:
            If usernames array already contains the member's username at index j:
                Then break from loop.
            Set element i of usernames array to the member's username at index j.

Call PopulateLblInfo method to instantiate label objects in the lblInfo array.

For loop to loop through the first element of the lblInfo 2D array (int i):
    If the logged in currentMember's username matches the text of the label in lblInfo at index 0, i:
        Then change the colour of the label forecolour (Text colour) to white and make the text bold. (Highlight the currently logged in user's username if present.

Call DisplayInfo function to add the labels to the table layout panel on the form.

//Back Button Click Event (BtnBack_Click):
Show the Main Menu form while closing the current form and call the main menu's play button click event method to transport user back to Game Level Select screen.

//PopulateLblInfo Method:
For loop to loop 10 times (int i):
    Instantiate the 10 labels in the first element of the lblInfo array and set its text equal to the username string in the usernames array at index i. Set each of their anchor to the left of the table cell.
    Instantiate the 10 labels in the second element of the lblInfo array and set its text equal to the high score integer in the topTenHighScores array at index i. Set each of their anchor to none.

//DisplayInfo Method:
Integer 'n' is set to 0.
For loop to loop 10 times (int i starts as 1):
    If the element at index of the username array is null:
        Then break from loop.
    Add the labels at index 0, n of the lblInfo array to second column of the table, row by row.
    Add the labels at index 1, n of the lblInfo array to the last column of the table, row by row.

Classes:

CustomException

//Attributes:
String ‘exceptionMessage’.

//Constructors:
Default constructor inheriting from the base Exception class.
Parametrised constructor with a string ‘message’ as the parameter, also inheriting from the base class and parameter.
{
    The exceptionMessage is set to the ‘message’.
}

//Properties:
String ‘ExceptionMessage’:
    Get { return exceptionMessage }
    Set { exceptionMessage = value }

Member

//Attributes:
String ‘firstName’, ‘lastName’, ‘username’, ‘email’, ‘password’, ‘errorMessage’.
DateTime ‘dateOfBirth’.
Integer ‘score’, ‘highScore’.

//Constructors:
Default constructor.

//Properties:
String ‘FirstName’:
    Get { return firstName }
    Set
    {
        If Validate_Name method, with value and “first” as parameters, returns true:
            firstName = value

        Else:
            Throw a new CustomException with errorMessage as the parameter.

    }

String ‘LastName’:

Get { return lastName }

Set

{

If Validate_Name method, with value and “last” as parameters, returns true:

lastName = value

Else:

Throw a new CustomException with errorMessage as the parameter.

}

String ‘Username’:

Get { return username }

Set

{

If Validate_Username method, with value as parameter, returns true:

username = value

Else:

Throw a new CustomException with errorMessage as the parameter.

}

String ‘Email’:

Get { return email }

Set

{

If Validate_Email method, with value as parameter, returns true:

email = value

Else:

Throw a new CustomException with errorMessage as the parameter.

}

DateTime ‘DateOfBirth’:

Get { return dateOfBirth }

Set

{

If Validate_DateOfBirth method, with value as parameter, returns true:

dateOfBirth = value

Else:

Throw a new CustomException with errorMessage as the parameter.

}

String ‘Password’:

Get { return password }
Set
{
If Validate_Password method, with value as parameter, returns true:
password = value

Else:
Throw a new CustomException with errorMessage as the parameter.

}

Integer 'Score':

Get { return score }
Set
{
score = value.
If score is greater than highScore:
Then set highScore to the score integer.
}

Integer 'HighScore':
Get { return highScore }

//Methods:

//Validate_Name:

Method takes 2 string parameters, one for the value 'n' we are trying to set to the variable and the second for the 'type' (E.g. First name or Last name).
If n is null or empty:
Then set the error message to a string telling the user to provide a 'type' name and return false.

For each loop to loop through every character 'c' in n:
If c is number or white space or punctuation:
Then set the error message to a string telling the user that this 'type' name is invalid and return false.

Return true by default.

//Validate_Username:

Method takes one string parameter 'u' for the value we are trying to set to the variable.
If u is null or empty:
Then set the error message to a string telling the user to provide a username and return false.

Call the ReadFromFile method with u as the parameter.

For each Member m in the Registration Form's members array:

If m is null:

Then break from loop.

If u matches the username of m:

Then set the error message to a string telling the user that the username entered already exists and return false.

Return true by default.

//Validate_Email:

Method takes one string parameter 'e' for the value we are trying to set to the variable.
If e is null or empty:

Then set the error message to a string telling the user to provide an e-mail address and return false.

If e does not contain '@' and '.':

Then set the error message to a string telling the user that the e-mail address provided is invalid and return false.

Return true by default.

//Validate_DateOfBirth:

Method takes one DateTime parameter 'dob' for the value we are trying to set to the variable.
Integer 'age' is set to the difference between the current year and the dob year.
If dob is greater than the result of current date subtract the age:

Then decrement age as an extra year has been taken into account.

If age is less than 7:

Then set the error message to a string telling the user that they cannot register as they are under-age and return false.

Return true by default.

//Validate_Password:

Method takes one string parameter 'p' for the value we are trying to set to the variable.
If p is null or empty:

Then set the error message to a string telling the user to provide a password and return false.

If the length of p is less than 6:

Then set the error message to a string telling the user that the password is too short and return false.

Integer 'digits' is set to 0.

For each character ‘c’ in p:

If c is a number:

Then increment digits.

If digits is less than 1:

Then set the error message to a string telling the user that the password must contain at least 1 digit and return false.

Return true by default.

//WriteToFile:

Method takes one integer parameter ‘numOfMembers’.
Instantiate a stream that opens the MemberData binary file in create mode to overwrite any existing members.
Instantiate a binary formatter and serialize the first element in the Registration Form’s members array to the file and close the stream.

Instantiate a stream that opens the MemberData binary file in append mode.
For loop to loop through the Registration Form’s members array (int i starts at 1):

Use the binary formatter to serialize the member at index i in the members array to the file.

Close the stream.

//ReadFromFile:

Instantiate a stream that opens the MemberData binary file to read.
Instantiate a binary formatter.
Integer i is set 0.
While the stream pointer position value is less than the length of the stream:

Set the member object at index i in the members array to the de-serialised stream using the binary formatter.

Close the stream.

//CountMembers:

Integer n is set to 0.
For each member object in the Registration Form’s members array:

If the member object is null:

Then break from the loop.

Else:

Increment n.

Return n. (i.e. Total number of members (That are not null))

Player

//Attributes:
Integer ‘speed’, ‘health’, ‘clip’, ‘ammoPouch’ and ‘max_clipSize’.
Character ‘direction’.

//Variables:
Public PictureBox ‘picBxPlayer’.
Private Integer ‘gameLevel’.
Public Boolean ‘up’, ‘left’, ‘down’, ‘right’.

Public ProgressBar ‘prgsBrReloadBar’.
Public Timer ‘tmrReloadTime’.
Public Boolean ‘reload’.

//Constructors:
Default constructor.
Parameterised constructor with integer ‘s’, ‘c’, ‘aP’, ‘m_cS’, Point ‘start’, integer ‘gL’ and Form ‘frm’ as the parameters.
{
    speed is set to s.
    health is set to 120.
    clip is set to c.
    ammoPouch is set to aP.
    max_clipSize is set to m_cS.
    Direction is set to character ‘r’.

    Instantiate picBxPlayer picture box and set its location to start.
    gameLevel is set gL.
    If gameLevel is equal to 1:
        Then set picBxPlayer’s image to level one (Right) image.
    Else:
        Set picBxPlayer’s image to level two (Right) image.
    Add picBxPlayer to frm.
}

//Properties:
Integer ‘Speed’:
    Get { return speed }

Integer ‘Health’:
    Get { return health }
    Set
    {
        If value is greater than 120:

Then health is set to 120.
Else:
health is set to value.
}

Integer 'Clip':
Get { return clip }
Set
{
If value is greater than max_clipSize:
Then add the difference between value and max_clipSize to ammoPouch.
Set clip to max_clipSize.
Else:
clip is set to value.
}

Integer 'AmmoPouch':
Get { return ammoPouch }
Set { ammoPouch is set to value }

Integer 'Max_ClipSize':
Get { return max_clipSize }

Character 'Direction':
Get { return direction }

//Methods:

//Up:

Set up to true and direction to the character 'u'.
If gameLevel is equal to 1:
Then set picBxPlayer's image to level one (Up) image.
Else:
Set picBxPlayer's image to level two (Up) image.

//Left:

Set left to true and direction to the character 'l'.
If gameLevel is equal to 1:
Then set picBxPlayer's image to level one (Left) image.
Else:
Set picBxPlayer's image to level two (Left) image.

//Down:

Set down to true and direction to the character 'd'.
If gameLevel is equal to 1:
Then set picBxPlayer's image to level one (Down) image.

Else:

Set picBxPlayer's image to level two (Down) image.

//Right:

Set right to true and direction to the character 'r'.

If gameLevel is equal to 1:

Then set picBxPlayer's image to level one (Right) image.

Else:

Set picBxPlayer's image to level two (Right) image.

//Reload:

Instantiate progress bar prgsBrReloadBar and set its value to 0 and its maximum value to 100. Set its style to continuous then add to frm.

Instantiate timer tmrReloadTime and set its interval to 300 milliseconds then add to frm

Set reload to true and start tmrReloadTime.

//Reload Timer Tick Event (TmrReloadTime_Tick):

If the value of prgsBrReloadBar is equal to its maximum value:

Dispose prgsBrReloadBar and set reload to false.

Integer 'reloadAmount' is set to the difference between the max_clipSize and clip.

If the reloadAmount is greater than ammoPouch:

Add the ammo in the ammoPouch to the clip and set ammoPouch back to 0.

Else:

Set clip to the max_clipSize and subtract the reloadAmount from ammoPouch.

Dispose the tmrReloadTime timer.

Else:

Add 20 to the value of prgsBrReloadBar.

Zombie

//Attributes:
Integer ‘speed’, ‘health’ and ‘damage’.
Size ‘zombieDimensions’.

//Variables:
Public PictureBox ‘picBxZombie’.
Public integer ‘zImage’.
Public integer ‘zombie_x’ and ‘zombie_y’.

//Constructors:
Parameterised constructor with parameters integer ‘s’, ‘d’ and Size ‘zD’.
{
    speed is set to s.
    health is set to 100.
    damage is set to d.
    zombieDimensions is set to zD.
    zImage is set to 1.
}

//Properties:
Integer ‘Speed’:
    Get { return speed }
    Set { speed is set to value }

Integer ‘Health’:
    Get { return health }
    Set { health is set to value }

Integer ‘Damage’:
    Get { return damage }
    Set { damage is set to value. }

//Methods:
//GenerateZombie:
Method takes parameters Form ‘frm’, integer ‘numOfZombies’ and method ‘rnd’.
health is set to 100.
While numOfZombies is not equal to 0:
    Decrement numOfZombies.
    Integer array ‘rndCoord’ is set to the call of the method ‘rnd’ which returns an integer array.
    zombie_x is set to the first element of rndCoord and zombie_y is set to the second element of rndCoord.

    Instantiate Picture Box picBxZombie and set its tag to a string “Zombie”.

If zImage is equal to 1:
Then set picBxZombie's image to zombie1 (Up) image.
Else:
Set picBxZombie's image to zombie2 (Up) image.
Set picBxZombie's size to zombieDimensions then add to frm.

//ZombieMovement:

Method takes parameters Control 'z', Player 'p' and then the InvisibleWall methods ('iW_A', 'iW_L', 'iW_B' and 'iW_R').
Integer 'z_x' is set to the x-coordinate location of z and 'z_y' is set to the y-coordinate location of z.

//Up

If z_y is more than the y-coordinate of p's picBxPlayer and iW_A returns false:
Then move z's y-coordinate up by speed.
If zImage is equal to 1:
Then set picBxZombie's image to zombie1 (Up) image.
Else:
Set picBxZombie's image to zombie2 (Up) image.

//Left

If z_x is more than the x-coordinate of p's picBxPlayer and iW_L returns false:
Then move z's x-coordinate up by speed.
If zImage is equal to 1:
Then set picBxZombie's image to zombie1 (Left) image.
Else:
Set picBxZombie's image to zombie2 (Left) image.

//Down

If z_y is less than the y-coordinate of p's picBxPlayer and iW_B returns false:
Then move z's y-coordinate up by speed.
If zImage is equal to 1:
Then set picBxZombie's image to zombie1 (Down) image.
Else:
Set picBxZombie's image to zombie2 (Down) image.

//Right

If z_x is less than the x-coordinate of p's picBxPlayer and iW_R returns false:
Then move z's x-coordinate up by speed.
If zImage is equal to 1:
Then set picBxZombie's image to zombie1 (Right) image.
Else:
Set picBxZombie's image to zombie2 (Right) image.

Set the z's location to a new point with the z_x and z_y.

Bullet

//Attributes:
Integer ‘speed’ and ‘damage’.
Character ‘direction’.

//Variables:
Public Picture Box ‘picBxBullet’.
Public Timer ‘tmrRefreshRate_B’.
Private Bullet ‘b’.
Private (Method) ‘CollisionsWithSurroundings’.

//Constructors:
Default constructor.
Parameterised constructor with parameters integer ‘s’, ‘d’ and character ‘dir’.
{
    speed is set to s.
    damage is set to d.
    direction is set to dir.
}

//Properties:
Integer ‘Speed’:
    Get { return speed }

Integer ‘Damage’:
    Get { return damage }

Character ‘Direction’:
    Get { return direction }

//Methods:
//GenerateBullet:
Method takes parameters Form ‘frm’, Size ‘bulletSize’, Point ‘bulletLocation’ and method ‘c’.
Instantiate Picture box ‘picBxBullet’ and set its tag to a string “Bullet”, its back colour to golden.
Set its size to bulletSize and its location to bulletLocation and then add to frm.

Instantiate Timer ‘tmrRefreshRate_B’ and set its interval to 30 milliseconds and then start the timer.

Instantiate Bullet b and then set the method c to CollisionsWithSurroundings.

//Bullet Timer Tick Event (TmrRefreshRate_B_Tick):
Integer x and y are set to the picBxBullet’s location co-ordinates.
If direction is character ‘u’:
    Then move bullet’s y-coordinate up by the speed amount.

If direction is character ‘l’:

Then move bullet’s x-coordinate left by the speed amount.

If direction is character ‘d’:

Then move bullet’s y-coordinate down by the speed amount.

If direction is character ‘r’:

Then move bullet’s x-coordinate right by the speed amount.

Set the picBxBullet’s location to a new point with the new x and y coordinates.

Call the CollisionsWithSurrounding method with Bullet object b as parameter.

Test Plan

Main Menu

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
1	Test to see if the main menu page loads correctly.	The main menu should load with the appropriate controls and the main background music (e.g. Buttons, Titles).	4
2	Test to see if the register button functions correctly.	The register button, once clicked, should transfer the user to the Registration Page.	4
3	Test to ensure the login button functions correctly.	The login button, when clicked, should transfer the user to the Login Page.	4
4	Test to check if the help button functions correctly.	The help button, when clicked, should open the help guide whilst keeping the main menu open in the background.	4
5	Test to see if the play button is enabled when user is not logged in.	The play button should only be enabled and green once the user has logged in. (Play button should not be clickable).	5
6	Test to see if the help tip appears when play button is hovered over.	When hovered over with the mouse, a tip should appear telling the user to log in to play.	2/5
7	Test to check if the play button functions correctly.	The enabled (Green) play button, when clicked, should change the main menu screen to a game level select screen.	6
8	Test to check if the appropriate buttons appear in game level select screen.	There should be 4 buttons in total that should draw on to the form, the buttons for the 2 levels, a back button and a high scores button.	6
9	Test to see if the level one button functions correctly. (Level Select)	The level one button, should transfer the user to game level one, once clicked.	6
10	Test to see if the level one button functions correctly. (Level Select)	The level one button once hovered over, should display text describing what the level entails.	2/6
11	Test to see if the level two button functions correctly. (Level Select)	The level two button should transfer the user to game level two, once clicked.	6

12	Test to see if the level two button functions correctly. (Level Select)	The level two button, once hovered over with the mouse, should display a description tip describing the level.	2/6
13	Test to test if the back button functions appropriately. (Level Select)	The back button should return the user back to the main menu screen, when clicked.	6
14	Test to check if the high scores button functions correctly. (Level Select)	The high scores button should transport the user to the high scores table.	6
15	Test to see if the exit button functions correctly.	The exit/quit button should close the application entirely, when clicked by the user.	7

Registration

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
16	Test to see if the registration page loads appropriately.	The registration page should load with the appropriate controls. (E.g. Titles, input fields, buttons, taskbar).	8
17	Test to see if the register button functions correctly.	The register button, when clicked, should check if the all the user input is valid or not.	8
18	Test to see if the message box appears.	A message box should be shown after the user has successfully registered an account.	2/8
19	Test to see if the home button functions correctly.	The home button should take the user to the main menu screen, when clicked.	8
20	Test to check if the login button functions correctly.	The login button should take the user to the login page, when clicked.	8
21	Test to check if the help button functions correctly.	The help button, once clicked, should open the help guide while keeping the registration page open in the background.	8
22	Test to check if the first name text box displays errors when necessary.	The first name text box should show error providers with appropriate captions when the user leaves it empty or inputs digits or whitespaces or punctuation/symbols.	2/9
23	Test to check if the last name text box displays errors when necessary.	The last name text box should show error providers with appropriate captions when the user leaves it empty or inputs digits or whitespaces or punctuation/symbols.	2/9
24	Test to check if the username text box displays errors when necessary.	The username text box should show error providers with appropriate captions when the user leaves it blank or inputs a username that already exists or if they enter whitespaces into the username.	2/10
25	Test to check if the e-mail text field shows errors when appropriate.	The e-mail text field should display errors with appropriate captions when the user leaves it blank or inputs an e-mail address without a “@” character and a full stop or with whitespaces.	2/11

26	Test to see if the date of birth displays errors when necessary.	The date of birth input field should show an error with caption if the user's age is less than 7 years.	2/12
27	Test to see if the password text field shows errors when necessary.	The password text box should display errors with the appropriate captions when the user leaves it empty or if they enter a password with under 6 characters or without at least 1 digit or if they enter a password with whitespaces.	2/13

Login

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
28	Test to see if the login page loads appropriately.	The login page should load with all the appropriate controls. (E.g. Titles, text fields, taskbar).	14
29	Test to check if the login button functions.	The login button should check if what the user entered matches with existing member properties.	14
30	Test to see if the message box appears.	A message box should appear when the user successfully logs in.	2/14
31	Test to see if the home button functions.	The home button, when clicked, should transport the user to the home screen.	14
32	Test to check if the register button functions.	The register button, once clicked, should change the screen to the registration page.	14
33	Test to see if the help button functions.	The help button, when clicked, should open the user help guide while keeping login page open in background.	14
34	Test to see if the username text field shows errors when necessary.	The username text box should display errors when the user does not fill in the field or enters a username that does not exist in the database.	2/15
35	Test to check if the password text field displays errors when necessary.	The password text field should place errors when the user leaves it blank or inputs a password that does not match with the corresponding username.	2/15

Help Guide

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
36	Test to see if the help page loads correctly.	The help guide should open with a text box filled with the game information.	16
37	Test to see if the scroll bar functions correctly.	The user should be able to use the scroll bar to scroll the text both up and down.	16

Game Level One

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
38	Test to see if the appropriate controls are drawn dynamically onto the form at runtime.	All the controls (e.g. Player picture box, Zombie picture boxes) should be drawn onto the form at runtime.	17
39	Test to see if the player moves up when the correct key/s is pressed.	Player will move up the form when the user presses the up-arrow key or 'W' key.	18
40	Test to see if the player moves right when the correct key/s is pressed.	Player will move to the right on the form when the user presses the right-arrow key or 'D' key.	18
41	Test to see if the player moves down when the correct key/s is pressed.	Player will move down the form when the user presses the down-arrow key or 'S' key.	18
42	Test to see if the player moves left when the correct key/s is pressed.	Player will move to the left on the form when the user presses the left-arrow key or 'A' key.	18
43	Test to see if the invisible boundaries are being set up and functioning correctly.	The player should not be able to walk over the labels and objects in the background. The player should stop moving when they reach an invisible boundary.	19
44	Test to see if the user can fire their weapon with a key press.	A bullet picture box should appear and travel from the gun in the direction that the player is facing after the user has pressed the space bar key. A gun shot sound effect should also play.	20
45	Test to see if the ammo in the gun clip decrements when weapon is fired.	The ammo in the gun clip should decrement when the user presses the space bar key.	20/24
46	Test to see if the zombies get disposed of when the bullet makes contact with their picture boxes.	When the bullet fired from the player's gun collides with the zombies, they are expected to disappear from the form and new zombies should appear at a random point.	20
47	Test to see if the score label gets updated.	The score label should increment when a zombie gets eliminated.	20/24
48	Test to check if the user can reload their weapon using a key press.	When the 'R' key is pressed, a progress bar should appear above the player and should fill with time. The reload sound effect should also play.	21

49	Test to see if the ammo label gets updated when player reloads.	When the player reloads the ammo from the ammo pouch should be relocated to the gun clip.	21/24
50	Test to see if the zombies are spawning on to the form.	The zombies should be generated and appear at random locations on the form when the form is loaded.	22
51	Test to see if the ammo refill box is spawning when the player runs out of ammo.	The ammo refill box should appear at a random location when the player has no ammo in their utility.	22
52	Test to see if the player collects the ammo when they collide with the ammo refill box.	The player should collect the ammo when they collide with the refill box. The ammo should be stored in the ammo pouch.	22/24
53	Test to see if the zombies move by themselves towards the player picture box.	All zombie picture boxes on the screen should move towards the player, regardless if the player is moving or not.	23
54	Test to see if the player loses health points when the zombies makes contact with it.	When the zombies collide with the player picture box, their health bar should decrement by the amount of damage dealt by the zombie.	23
55	Test to see if the health progress bar changes colour to warn user.	The health progress bar is expected to change to red when reaching critical levels (e.g. <20hp).	25
56	Test to see if the ammo label changes colour to warn user.	The ammo label should change to orange when at critical levels and then to red when it reaches 0.	25
57	Test to see if the user can pause and un-pause the game at any time.	The user should be able to freeze the game when the 'P' key is pressed. Main menu and help buttons should appear and function appropriately and the user should be able to return back to the game by pressing any other key.	26
58	Test to see if the main menu button click event functions correctly.	The main menu button, when clicked by the user, should take them to the main menu form.	26/28
59	Test to see if the help button click event functions correctly.	The help/guide button, when clicked, should open the help page while keeping the current form open in the background.	26/28
60	Test to see if the 'game over' screen appears when the player is eliminated.	The game over label, retry, main menu and help buttons should appear along the centre of the form, when the player's health reaches 0hp.	28

61	Test to check if the retry button click event during 'game over' event functions correctly.	The 'Play Again?' button, when clicked, should restart the level again.	28
62	Test to see if the game's difficulty steps up when the user's score reaches 50.	The game's difficulty is increase when the user's score reaches 50. (More zombies appear on the form, the zombie's speed and strength increases).	30

Game Level Two

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
63	Test to see if the appropriate controls are drawn dynamically onto the form at runtime.	All the controls (e.g. Player picture box, Zombie picture boxes) should be drawn onto the form at runtime.	17
64	Test to see if the player moves up when the correct key/s is pressed.	Player will move up the form when the user presses the up-arrow key or 'W' key.	18
65	Test to see if the player moves right when the correct key/s is pressed.	Player will move to the right on the form when the user presses the right-arrow key or 'D' key.	18
66	Test to see if the player moves down when the correct key/s is pressed.	Player will move down the form when the user presses the down-arrow key or 'S' key.	18
67	Test to see if the player moves left when the correct key/s is pressed.	Player will move to the left on the form when the user presses the left-arrow key or 'A' key.	18
68	Test to see if the invisible boundaries are being set up and functioning correctly.	The player should not be able to walk over the labels and objects in the background. The player should stop moving when they reach an invisible boundary.	19
69	Test to see if the user can fire their weapon with a key press.	A bullet picture box should appear and travel from the gun in the direction that the player is facing after the user has pressed the space bar key. A gun shot sound effect should also play.	20
70	Test to see if the ammo in the gun clip decrements when weapon is fired.	The ammo in the gun clip should decrement when the user presses the space bar key.	20/24
71	Test to see if the zombies' health decreases when the bullet makes contact with their picture boxes.	When the bullet fired from the player's gun collides with the zombies, their health should decrease by the bullet damage amount and new zombies should appear at a random point only when the zombie is eliminated.	20
72	Test to see if the score label gets updated.	The score label should increment when a zombie gets eliminated.	20/24
	Test to check if the user can reload their	When the 'R' key is pressed, a progress bar should appear above the player	

73	weapon using a key press.	and should fill with time. The reload sound effect should also play, as well as, relocating the ammo from the ammo pouch to the gun clip, ready to be fired.	21
74	Test to see if the ammo label gets updated when player reloads.	When the player reloads the ammo from the ammo pouch should be relocated to the gun clip.	21/24
75	Test to see if the zombies are spawning on to the form.	The zombies should be generated and appear at random locations on the form when the form is loaded.	22
76	Test to see if the ammo refill box is spawning when the player runs out of ammo.	The ammo refill box should appear at a random location when the player has no ammo in their utility.	22
77	Test to see if the player collects the ammo when they collide with the ammo refill box.	The player should collect the ammo when they collide with the refill box. The ammo should be stored in the ammo pouch.	22/24
78	Test to see if the zombies move by themselves towards the player picture box.	All zombie picture boxes on the screen should move towards the player, regardless if the player is moving or not.	23
79	Test to see if the player loses health points when the zombies makes contact with it.	When the zombies collide with the player picture box, their health bar should decrement by the amount of damage dealt by the zombie.	23
80	Test to see if the health progress bar changes colour to warn user.	The health progress bar is expected to change to red when reaching critical levels (e.g. <20hp).	25
81	Test to see if the ammo label changes colour to warn user.	The ammo label should change to orange when at critical levels and then to red when it reaches 0.	25
82	Test to see if the user can pause and un-pause the game at any time.	The user should be able to freeze the game when the ‘P’ key is pressed. Main menu and help buttons should appear and function appropriately and the user should be able to return back to the game by pressing any other key.	26
83	Test to see if the main menu button click event functions correctly.	The main menu button, when clicked by the user, should take them to the main menu form.	26/28
84	Test to see if the help button click event functions correctly.	The help/guide button, when clicked, should open the help page while keeping the current form open in the background.	26/28

85	Test to see if the appropriate appear at the specified locations when the player is eliminated.	The game over label, retry, main menu and help buttons should appear along the centre of the form, when the player's health reaches 0hp.	28
86	Test to check if the retry button click event during 'game over' event functions correctly.	The 'Play Again?' button, when clicked, should restart the level again.	28
87	Test to see if the game's difficulty steps up when the user's score reaches 20.	The game's difficulty is increase when the user's score reaches 20. (More zombies appear on the form, the zombie's speed and strength increases).	32

High Scores

Test No.	Test Data/Reasons for Test	Expected Outcome	User Requirement
88	Test to see if the high scores screen loads correctly.	The High Scores table should load with the appropriate controls (e.g. Title, Table and button).	34
89	Test to see if the high scores are displayed.	The high scores should be displayed in the last column in descending order.	34
90	Test to check if the corresponding username are displayed.	The usernames should be displayed in the second column, beside their corresponding high score.	34
91	Test to see if the high scores updates when loaded.	The high scores should update when the user achieves a new high score.	34
92	Test to check if the current user's username is highlighted.	The currently logged in user's username should be highlighted (Bold) if present on the leader board.	35
93	Test to check if the back button functions correctly.	The back button, once clicked, should transfer the user back to the game level select screen.	36

TESTING

Testing

Main Menu

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
1	The main menu should load with the appropriate controls and the main background music (e.g. Buttons, Titles).	The main menu form loads correctly with all appropriate controls and the background music starts to play. **[PASSED]**	N/A	p71	4
2	The register button, once clicked, should transfer the user to the Registration Page.	When the register button is clicked, the user is transported to the Registration page. **[PASSED]**	N/A	p72	4
3	The login button, when clicked, should transfer the user to the Login Page.	When the login button is clicked, the user is transported to the Login page. **[PASSED]**	N/A	p72	4
4	The help button, when clicked, should open the help guide whilst keeping the main menu open in the background.	When the help button is clicked, the help guide opens while keeping the main menu open in the background. **[PASSED]**	N/A	p73	4
5	The play button should only be enabled and green once the user has logged in. (Play button should not be clickable).	The play button is only enabled when the user is logged in. **[PASSED]**	N/A	p73	5
6	When hovered over with the mouse, a tip should appear telling the user to log in to play.	When the mouse is hovered over the disabled play button, a tool tip appears. **[PASSED]**	N/A	p73	2/5

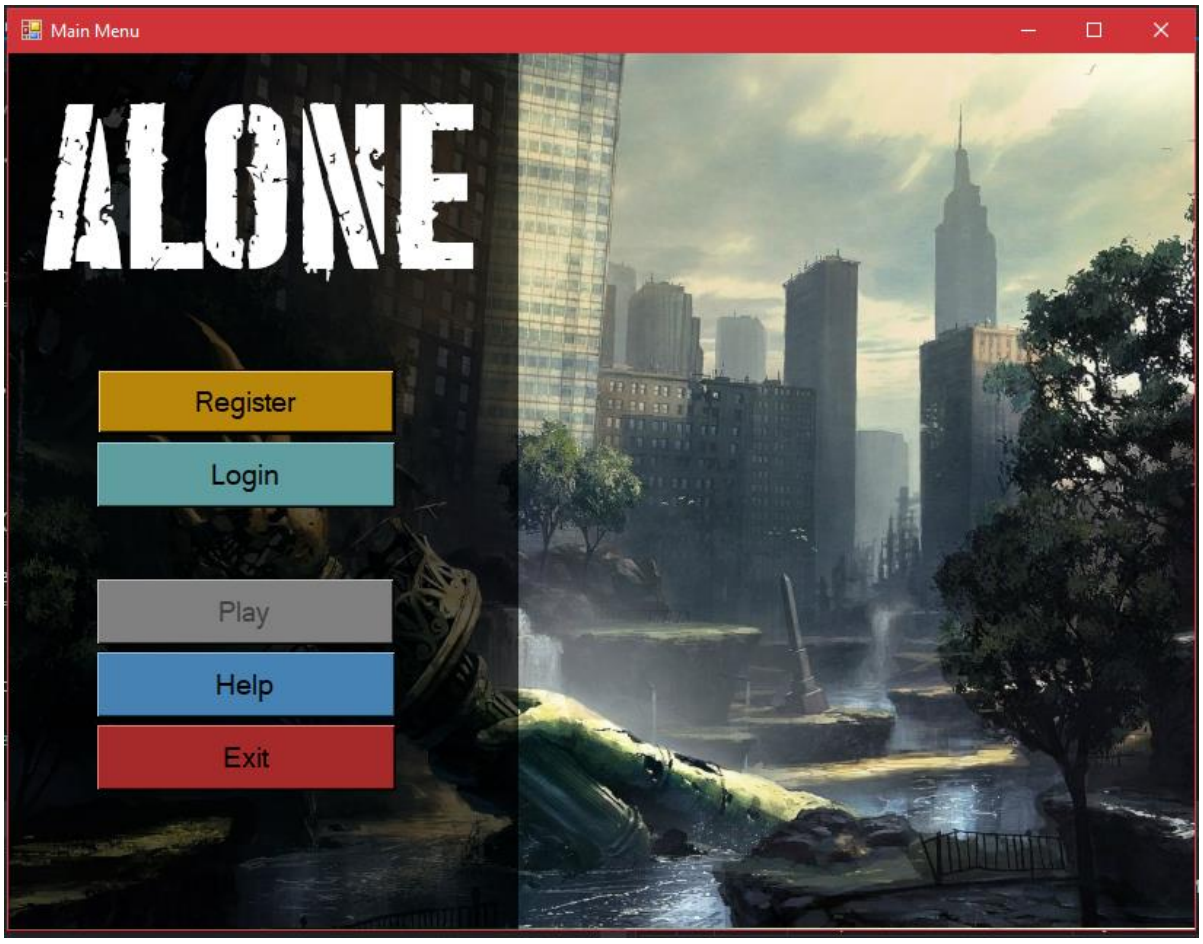
7	The enabled (Green) play button, when clicked, should change the main menu screen to a game level select screen.	When the enabled play button is clicked, the main menu fails to change to the game level select form. **[FAILED]**	Instead of showing another form for the game level select screen. The screen and controls will be dynamically drawn on the main menu form.	p74	5
8	There should be 3 buttons in total that should draw on to the form, the buttons for the 2 levels and a back button.	The game level screen draws on dynamically with the desired controls on to the main menu form. **(After Corrective Measure for Test 7).** **[PASSED]**	N/A	p74	6
9	The level one button, should transfer the user to game level one, once clicked.	The level one button successfully opens and runs the level one form. **[PASSED]**	N/A	p75	6
10	The level one button once hovered over, should display text describing what the level entails.	The level one button successfully displays the tool tip when hovered over with the mouse. **[PASSED]**	N/A	p75	2/6
11	The level two button should transfer the user to game level two, once clicked.	The level two button successfully opens and runs the level two form. **[PASSED]**	N/A	p76	6
12	The level two button, once hovered over with the mouse, should display a description tip describing the level.	The level two button successfully displays the tool tip once hovered over. **[PASSED]**	N/A	p76	2/6
13	The back button should return the user back to the main menu screen, when clicked.	The back button successfully returns the user back to the main menu form. **[PASSED]**	N/A	p77	6

14	The high scores button should transport the user to the high scores table.	The high scores button successfully transfers the user to the leader board. **[PASSED]**	N/A	p77	6
15	The exit/quit button should close the application entirely, when clicked by the user.	The exit button successfully closes and stops the whole application. **[PASSED]**	N/A	p78	7

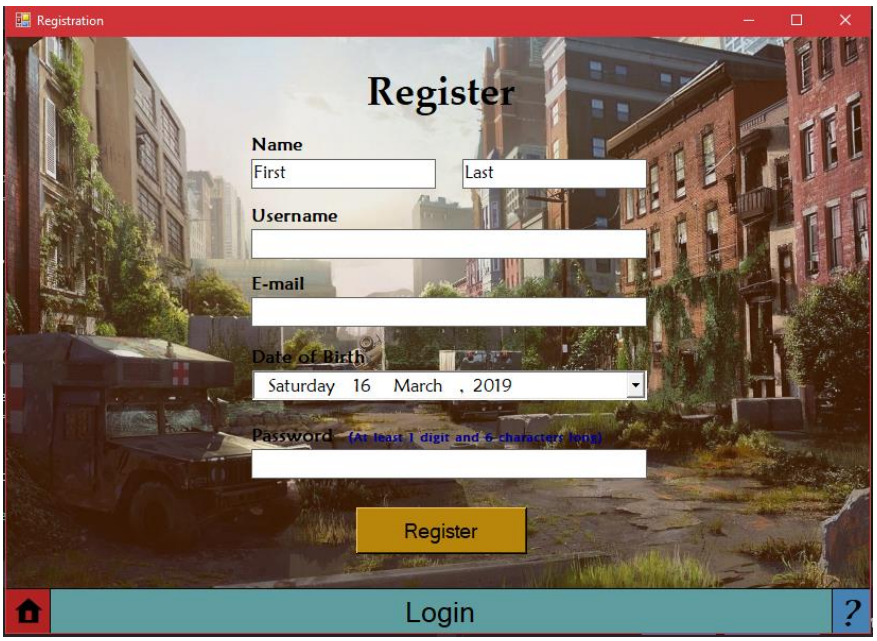
Test Evidence:

1.

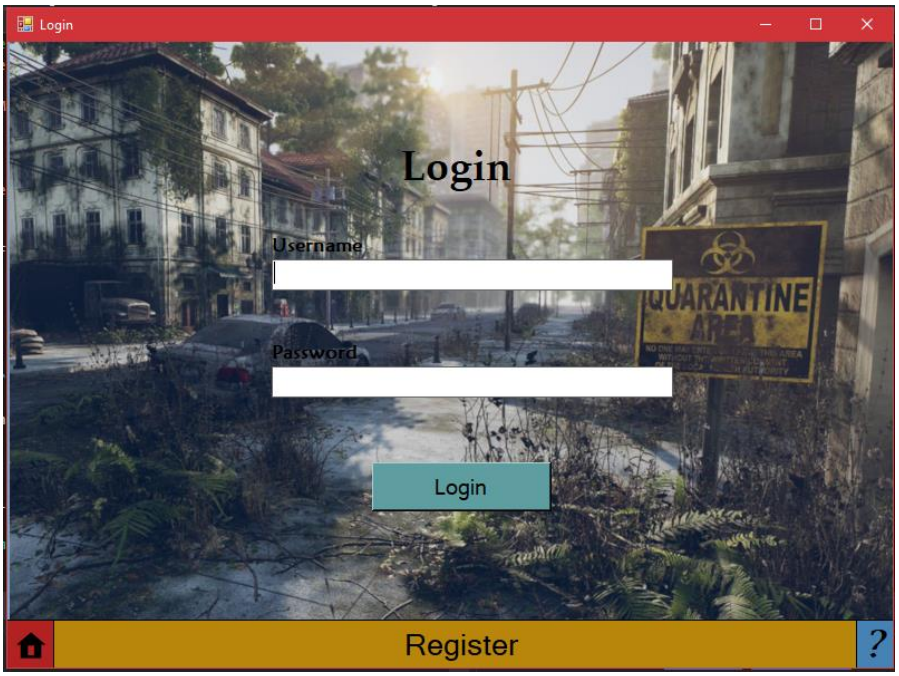
The Main Menu Form successfully loads with the appropriate controls with the background music.



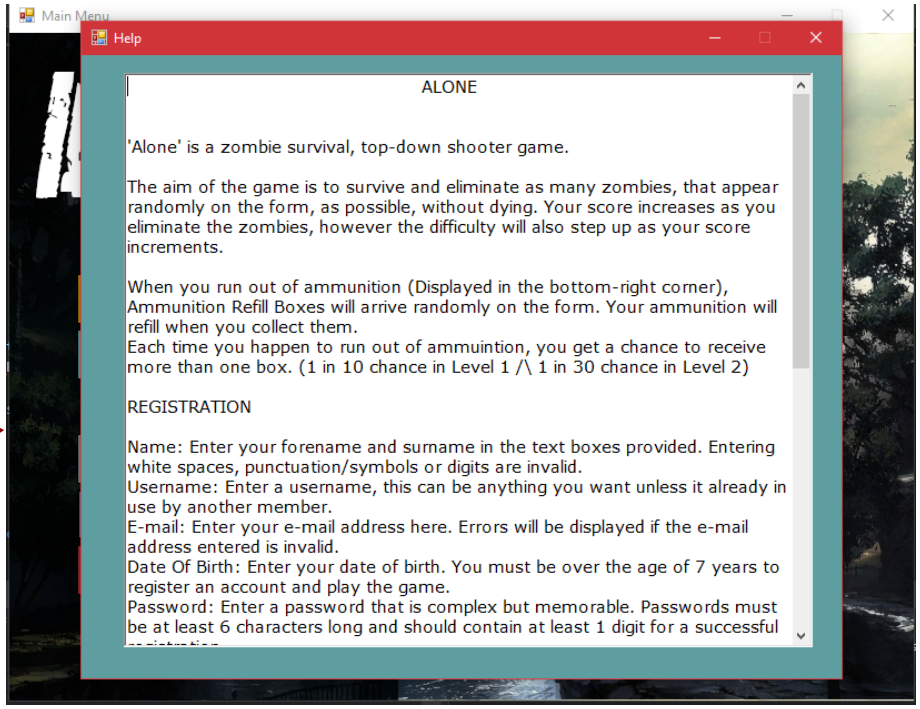
2. The BtnRegister_Click Event works correctly by switching to the Registration Form.



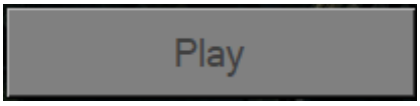
3. The BtnLogin_Click Event works correctly by switching to the Login Form.



4. The BtnHelp_Click Event works correctly by opening the Help Guide Form.

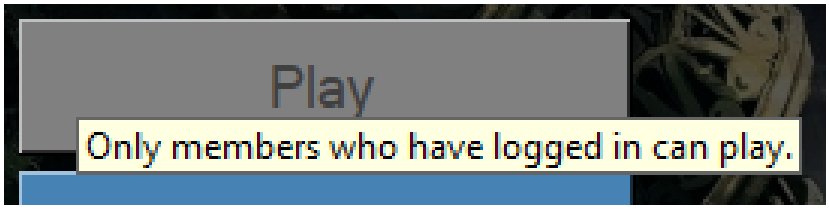


5. Play button is disabled (Grey) as user has not logged in.



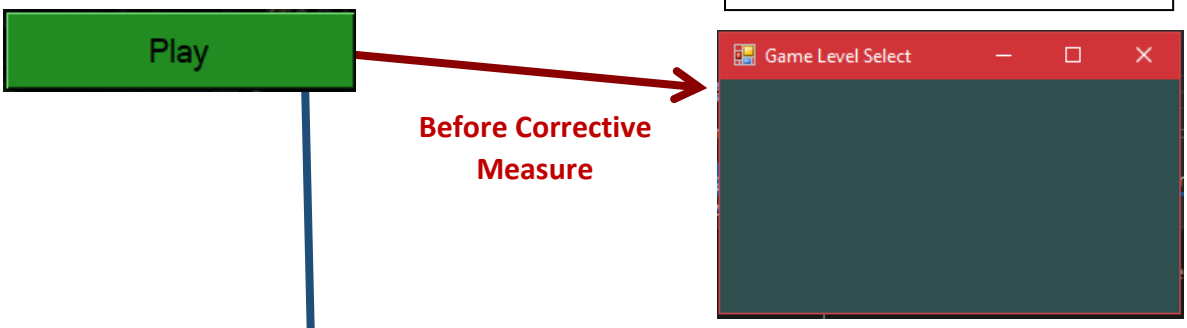
Play button is enabled (Green) when user has successfully logged in.

6. The tool tip is displayed when the level one button is hovered over in the BtnPlay_MouseHover Event.



7.

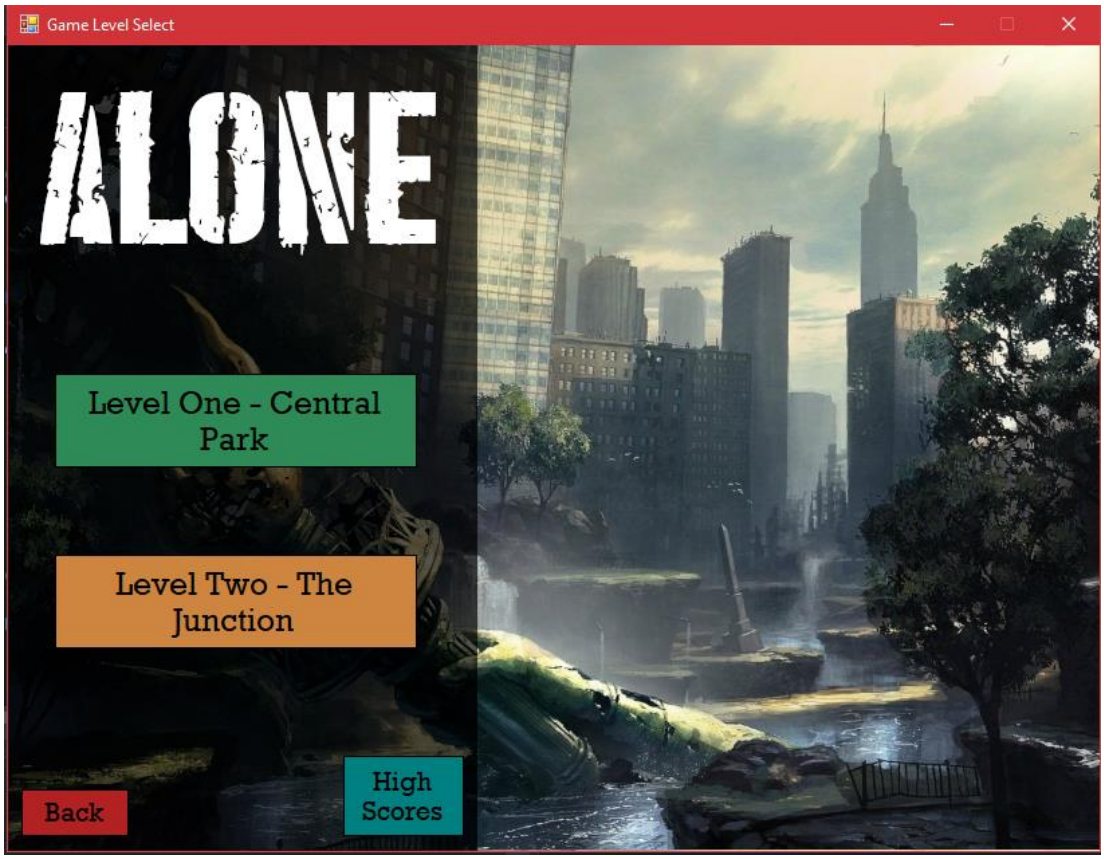
The Game Level Select Form was not loading correctly with the correct contols.



After Corrective Measure

8.

The Game Level Select screen now successfully is drawn onto the cleared main menu form in the BtnPlay_Click Event.

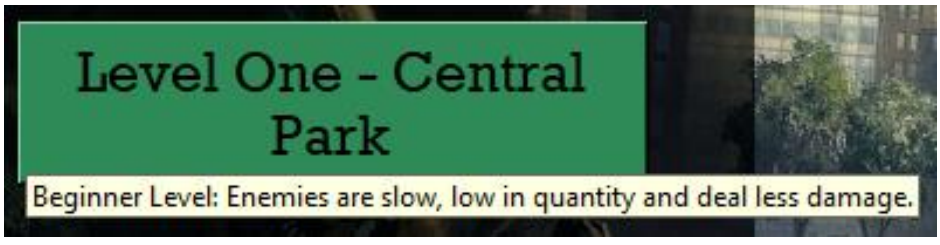


9.

The Game Level One Form loads correctly from the BtnLevelOne_Click Event.

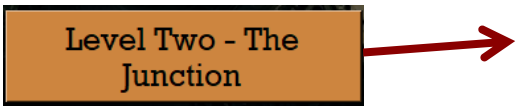
10.

The tool tip is displayed when the level one button is hovered over in the BtnLevelOne_MouseHover Event.



11.

The Game Level Two Form loads correctly from the BtnLevelTwo_Click Event.



12.

The tool tip is displayed when the level two button is hovered over in the BtnLevelTwo_MouseHover Event.

13.

The Main Menu Form loads correctly from the BtnBack_Click Event in the Game Level Select screen.

Back

14.

The High Scores Form loads correctly from the BtnHighScores_Click Event in the Game Level Select screen.

High Scores

15.

The BtnExit_Click Event occurs successfully, closing the application from the Main Menu Form

Registration

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
16	The registration page should load with the appropriate controls. (E.g. Titles, input fields, buttons, taskbar).	All the appropriate controls load correctly on the registration form. **[PASSED]**	N/A	p80	8
17	The register button, when clicked, should check if the all the user input is valid or not.	When clicked, the register button successfully does a final check on all input fields and assigns errors appropriately. **[PASSED]**	N/A	p81	8
18	A message box should be shown after the user has successfully registered an account.	When the register button is clicked, and the user has successfully registered an account, a message box appears welcoming the user. **[PASSED]**	N/A	p81	2/8
19	The home button should take the user to the main menu screen, when clicked.	When clicked, the home button takes the user to the main menu form. **[PASSED]**	N/A	p82	8
20	The login button should take the user to the login page, when clicked.	When the login button is clicked, the user is transported to the login page. **[PASSED]**	N/A	p82	8

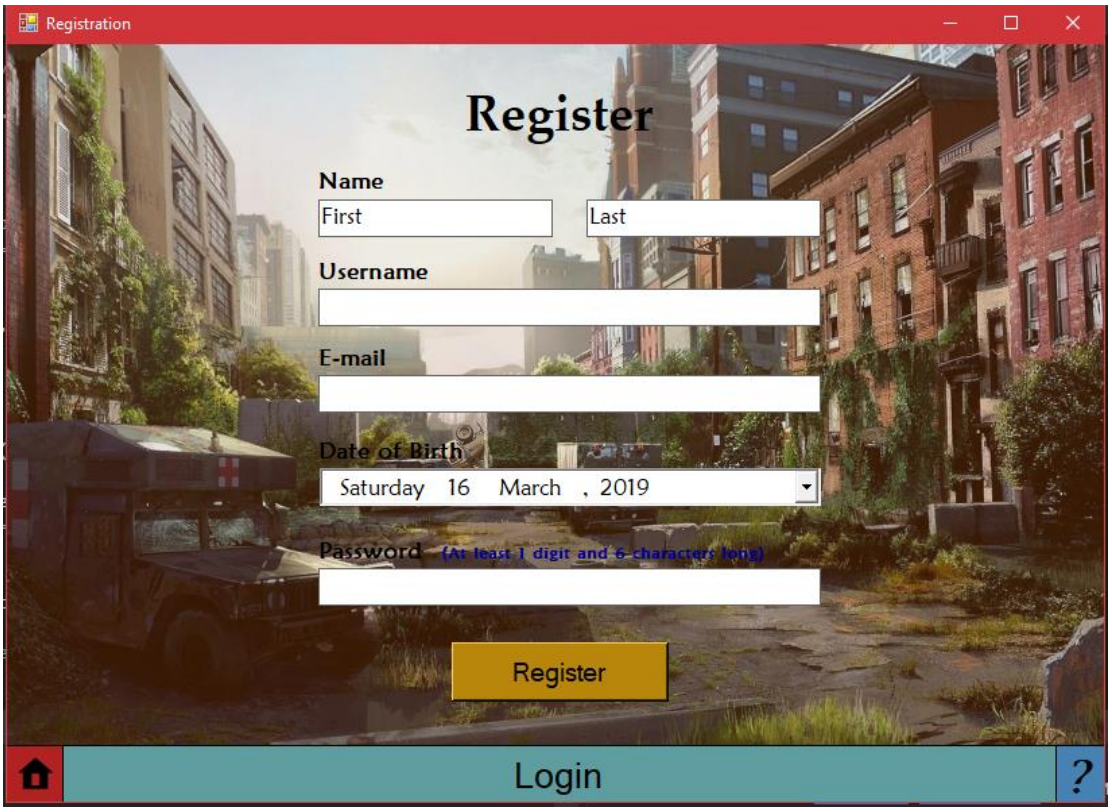
21	The help button, once clicked, should open the help guide while keeping the registration page open in the background.	When the help button is clicked, the help guide opens successfully in front of the registration page. **[PASSED]**	N/A	p83	8
22	The first name text box should show error providers with appropriate captions when the user leaves it empty or inputs digits or whitespaces or punctuation/symbols.	The appropriate error is shown on the first name text box, as the text is changed. **[PASSED]**	N/A	p83	2/9
23	The last name text box should show error providers with appropriate captions when the user leaves it empty or inputs digits or whitespaces or punctuation/symbols.	The appropriate errors are shown on the last name text box, as the text is changed. **[PASSED]**	N/A	p83	2/9
24	The username text box should show error providers with appropriate captions when the user leaves it blank or inputs a username that already exists or if they enter whitespaces into the username.	The appropriate errors are shown on the username text field, as the text is entered. **[PASSED]**	N/A	p84	2/10
25	The e-mail text field should display errors with appropriate captions when the user leaves it blank or inputs an e-mail address without a “@” character and a full stop or with whitespaces.	The correct errors are shown on the e-mail text field, when the text is entered. **[PASSED]**	N/A	p84	2/11
26	The date of birth input field should show an error with caption if the user’s age is less than 7 years.	The correct error is displayed on the date time picker field, when the date is changed by the user.	N/A	p84	2/12

		[PASSED]			
27	The password text box should display errors with the appropriate captions when the user leaves it empty or if they enter a password with under 6 characters or without at least 1 digit or if they enter a password with whitespaces.	The correct errors are displayed on the password text field, when the text is changed. **[PASSED]**	N/A	p85	2/13

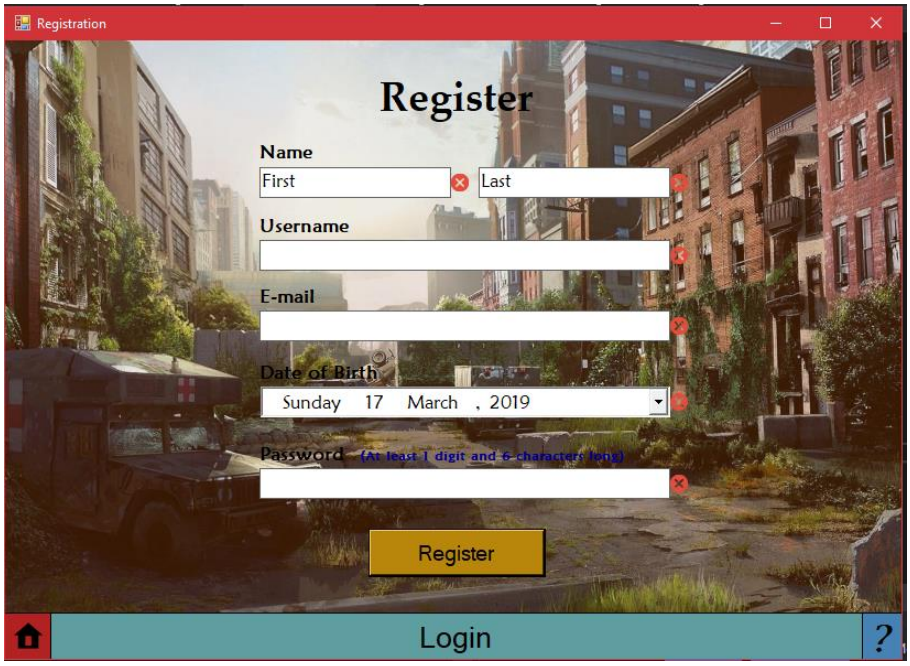
Test Evidence:

16.

The Registration Form successfully loads with the appropriate controls.

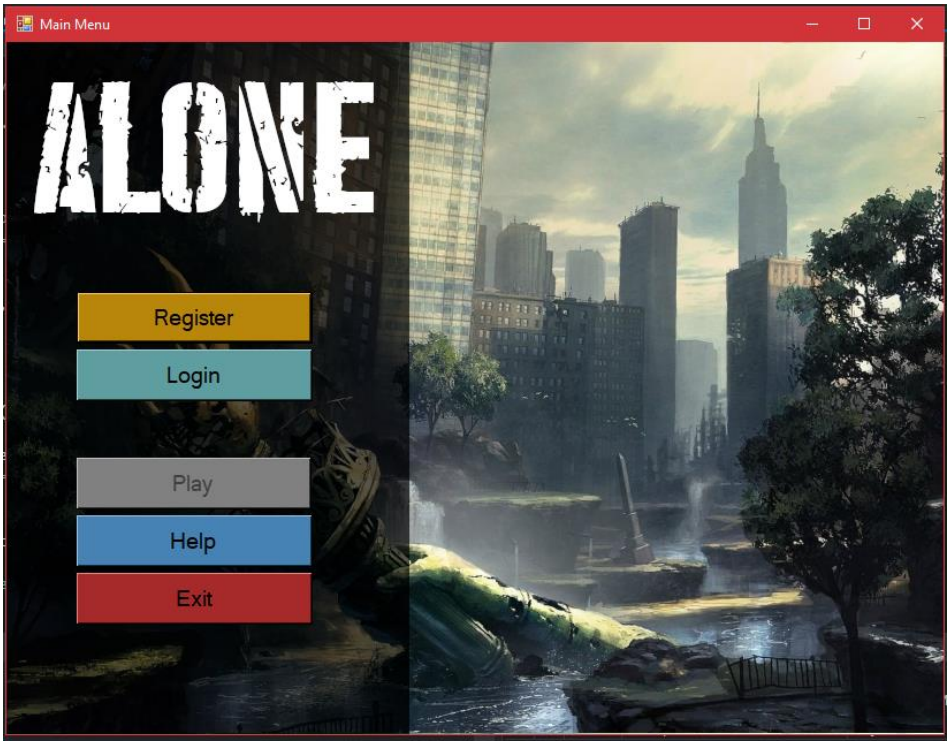


17. Unsuccessful user registration example, where the errors are displayed appropriately from the BtnRegister_Click Event.

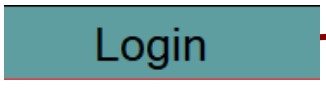


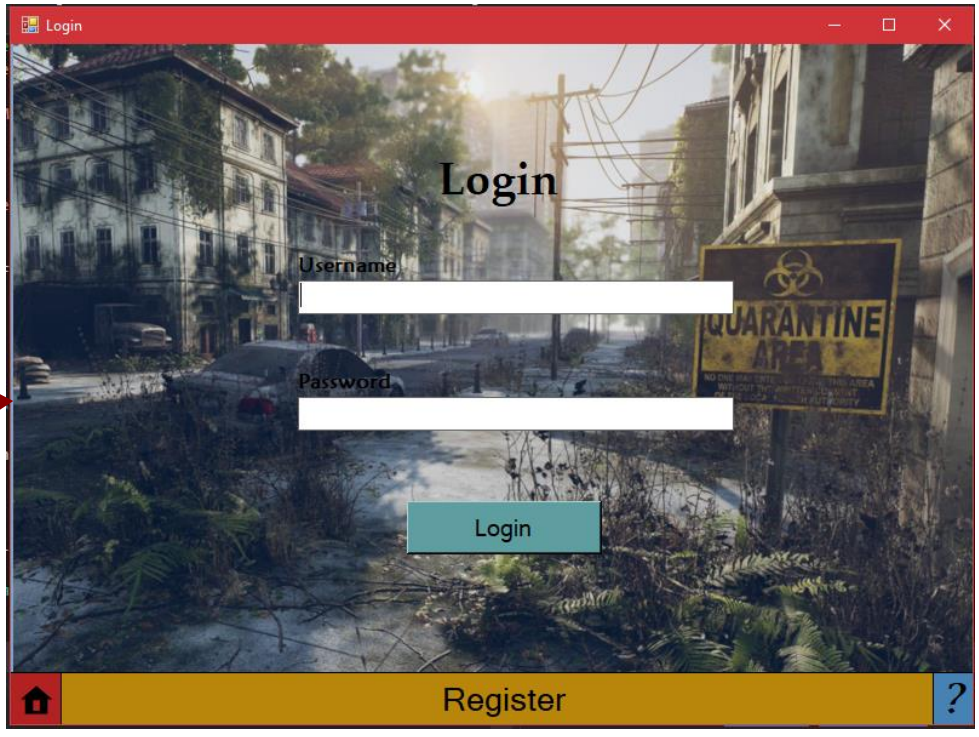
18. A successful user registration results in a message box successfully notifying the user they have registered. (From the BtnRegister_Click Event.)

19. The Main Menu Form successfully loads in the BtnHome_Click Event. Disabled Play button example when user is not logged in.



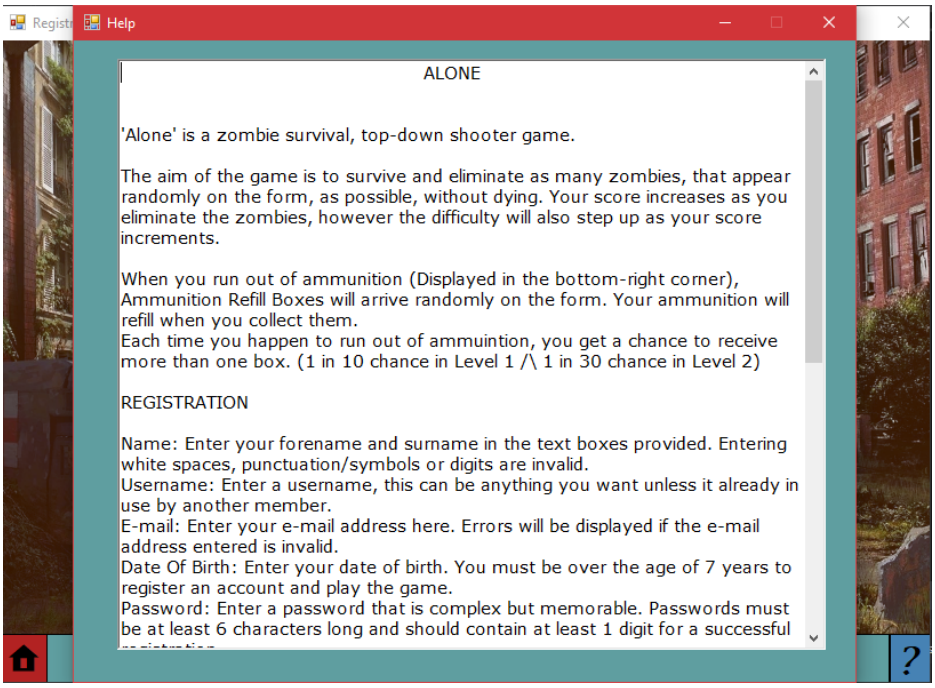
20. The Login Form loads correctly in the BtnLogin_Click Event.





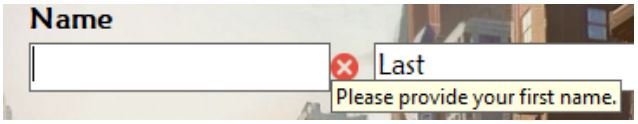
21.

The Help Guide Form opens correctly in the BtnHelp_Click Event.

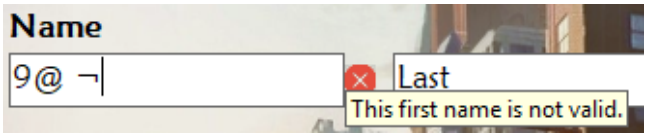


22.

When text field is empty.



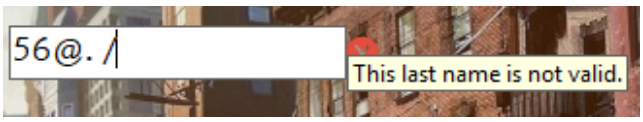
When user enters numbers, punctuation/symbols or whitespaces.



23.

When text field is empty.

When user enters numbers, punctuation/symbols or whitespaces.

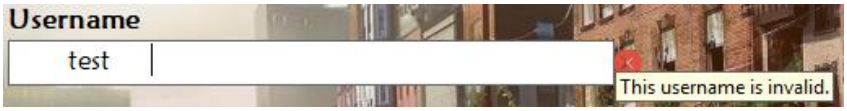


24.

When text field is empty.

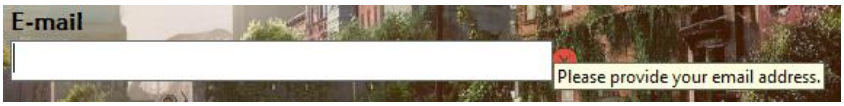
When user enters a username that exists already in the database.

When the user enters a username with whitespaces.



25.

When text field is empty.

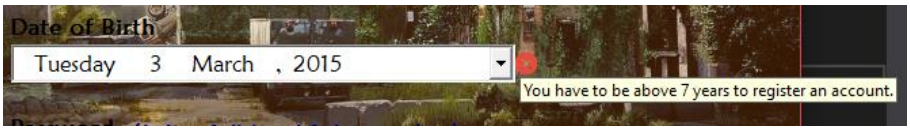


When user enters an invalid e-mail address (Without '@' and '.')

When the user enters a username with whitespaces.

26.

When the user enters a date of birth and they are under 7 years old.



27.

When text field is empty.

When the user enters a password that is under 6 characters.

When the user enters a password that does not contain at least 1 digit.

When the user enters a username with whitespaces.

Login

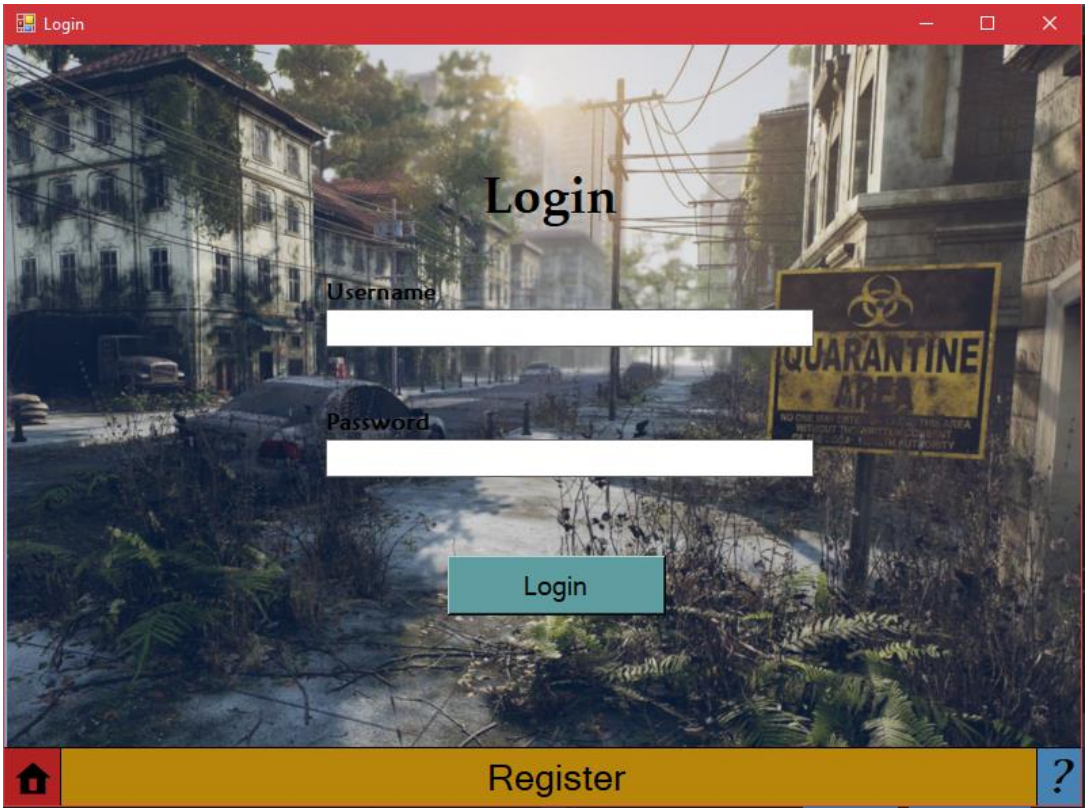
Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
28	The login page should load with all the appropriate controls. (E.g. Titles, text fields, taskbar).	All the appropriate controls load on to the page when initialised. **[PASSED]**	N/A	p87	14
29	The login button should check if what the user entered matches with existing member properties.	When clicked, the login button successfully checks all text fields and assigns errors appropriately. **[PASSED]**	N/A	p88	14
30	A message box should appear when the user successfully logs in.	When the login button is clicked, and the user's inputs successfully matches existing usernames and passwords, a message box appears notifying the user that they are logged in. **[PASSED]**	N/A	p88	2/14
31	The home button, when clicked, should transport the user to the home screen.	The home button successfully transports the user to the main menu form, when clicked. **[PASSED]**	N/A	p89	14
32	The register button, once clicked, should change the screen to the registration page.	The register button, when pressed, changes the form to the registration page. **[PASSED]**	N/A	p89	14
33	The help button, when clicked, should open the user help guide while keeping login page open in background.	The help button opens the help guide as expected. **[PASSED]**	N/A	p90	14

34	The username text box should display errors when the user does not fill in the field or enters a username that does not exist in the database.	The correct errors are shown on the username text field, when the login button is pressed. **[PASSED]**	N/A	p90	2/15
35	The password text field should place errors when the user leaves it blank or inputs a password that does not match with the corresponding username.	The correct errors are displayed on the password text field, when the login button is pressed. **[PASSED]**	N/A	p90	2/15

Test Evidence:

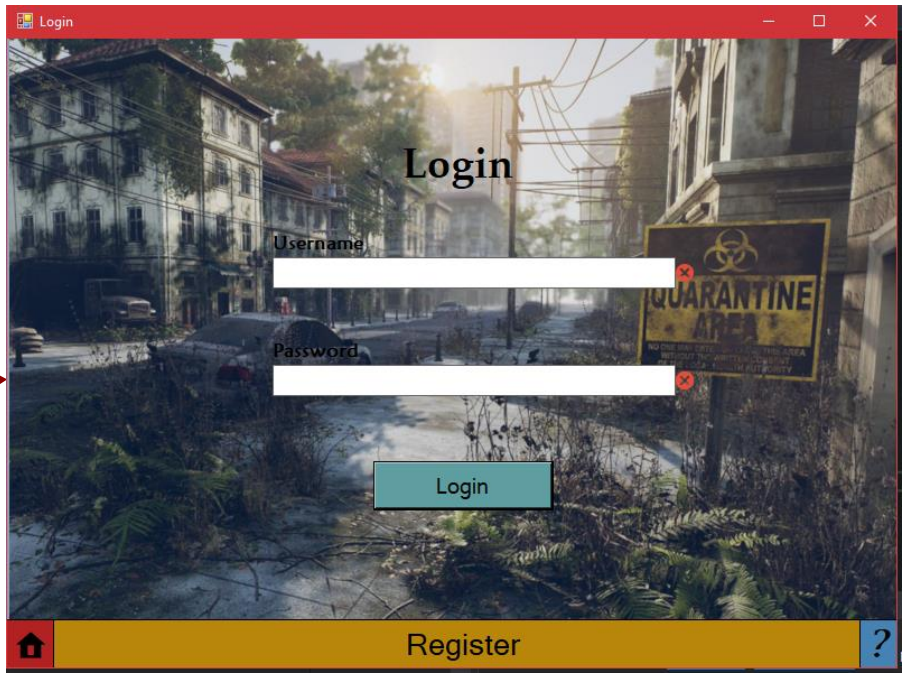
28.

The Login Form successfully loads with the appropriate controls.



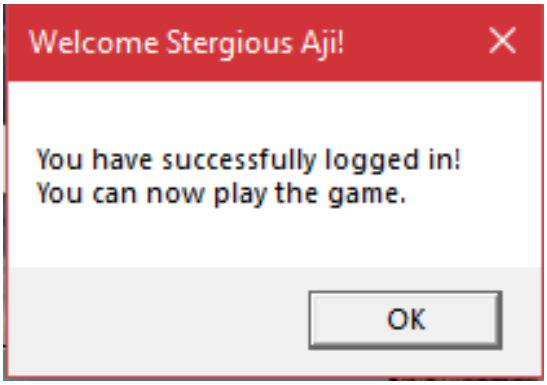
29.

Unsuccessful user login example, where the errors are displayed appropriately from the BtnLogin_Click Event.



30.

A successful user login results in a message box successfully notifying the user they have logged in. (From the BtnRegister_Click Event.)

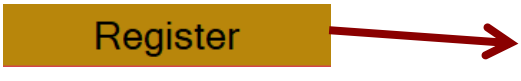


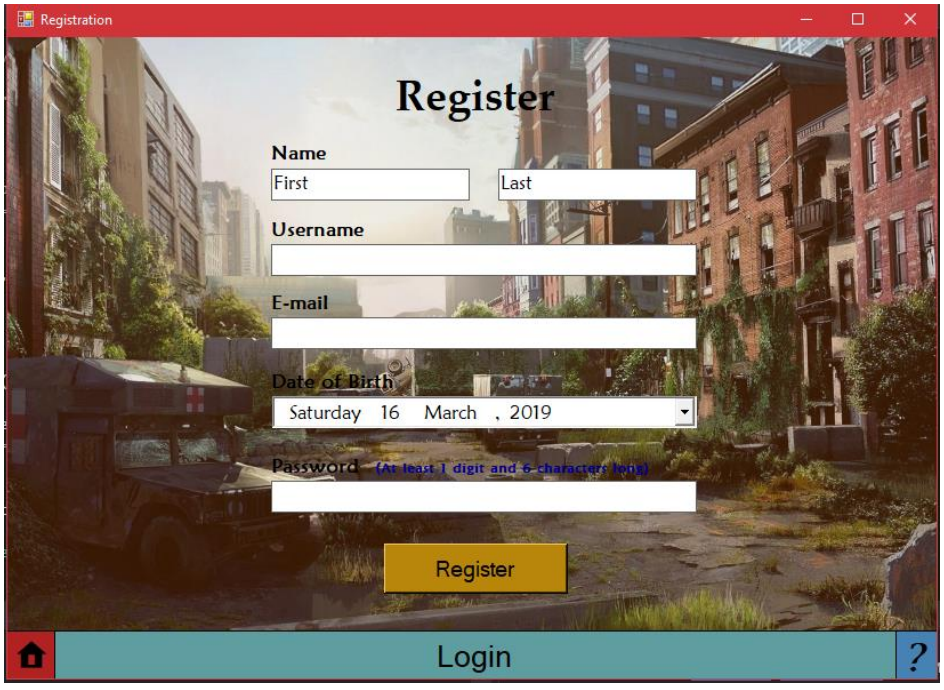
31.

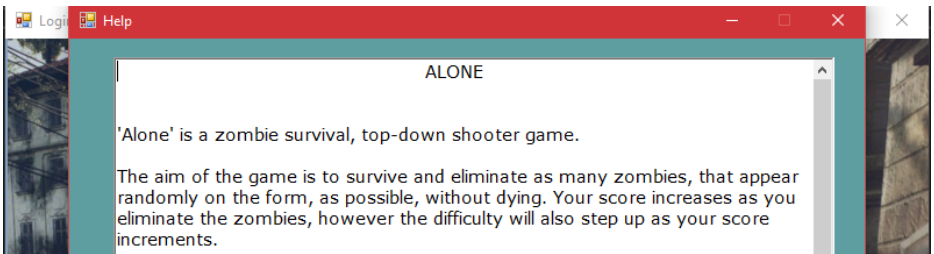
The Main Menu Form successfully loads in the BtnHome_Click Event. Disabled Play button example when user is not logged in.

32.

The Registration Form loads correctly in the BtnRegister_Click Event.







33.

The Help Guide Form opens correctly in the BtnHelp_Click Event.

34.

When text field is empty.

When the user enters a username that is not in the database.

35.

When text field is empty.

When the user inputs a password that does not match the corresponding username

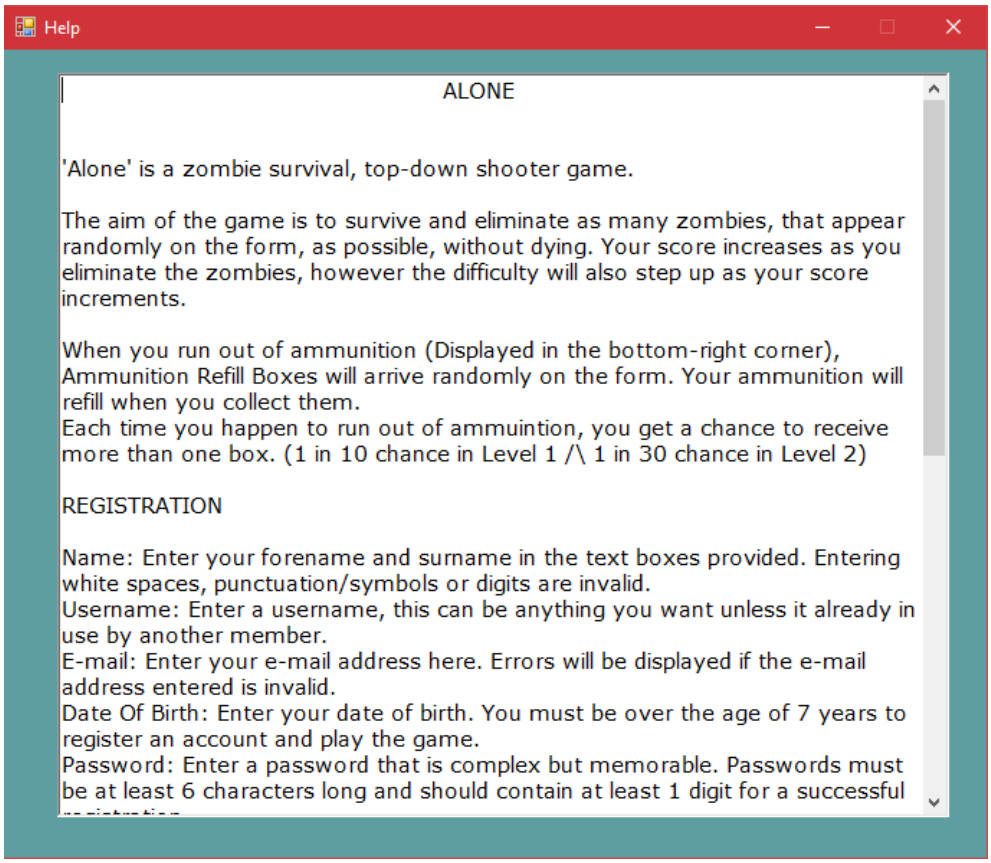
Help Guide

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
36	The help guide should open with a text box filled with the game information.	The help guide opens on top of the previous form with the text file successfully displayed on the text box. **[PASSED]**	N/A	p91	16
37	The user should be able to use the scroll bar to scroll the text both up and down.	The scroll bar functions correctly scrolling both up and down. **[PASSED]**	N/A	p92	16

Test Evidence:

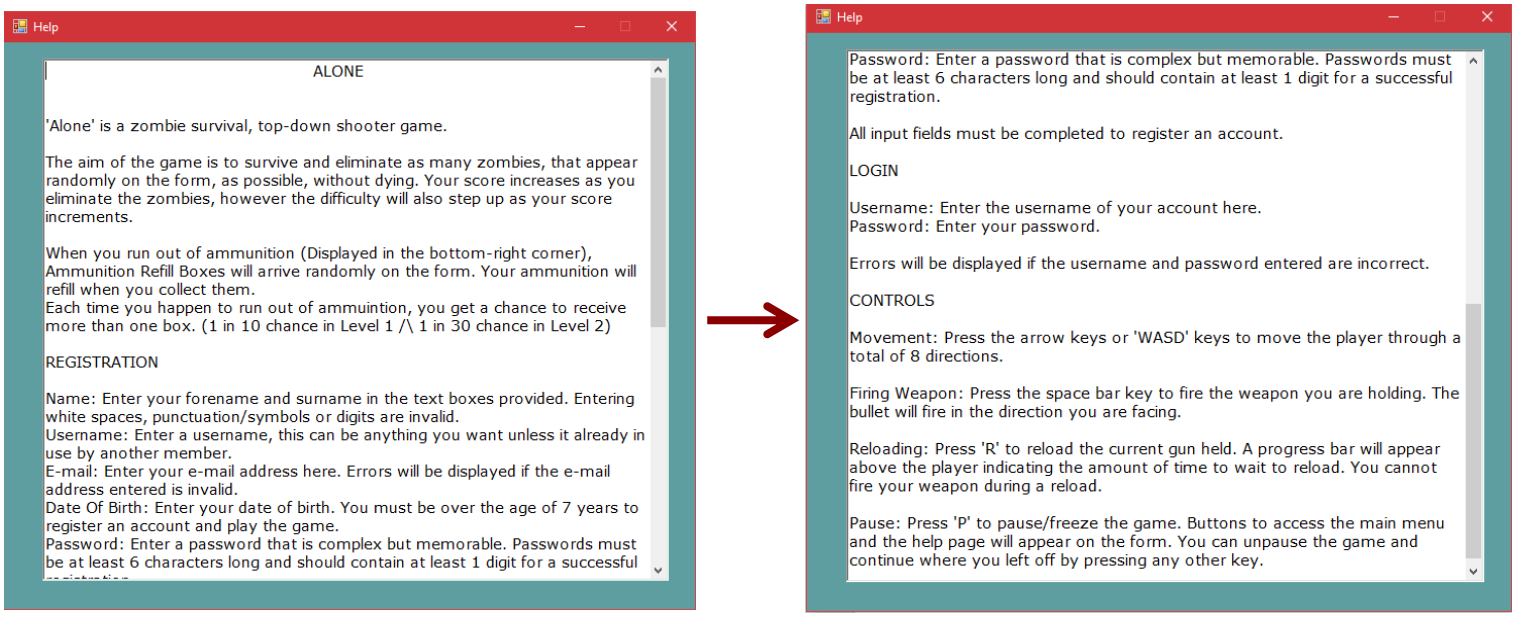
36.

The Help Guide Form successfully loads with the appropriate controls.



37.

The user can successfully scroll vertically through the rich text box with the text file.



Game Level One

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
38	All the controls (e.g. Player picture box, Zombie picture boxes) should be drawn onto the form at runtime.	All the appropriate controls are drawn at the set locations correctly. **[PASSED]**	N/A	p98	17
39	Player will move up the form when the user presses the up-arrow key or 'W' key.	Player picture box successfully moves up the form and player image faces up when the up-arrow key or 'W' key is pressed. **[PASSED]**	N/A	p98	18
40	Player will move to the right on the form when the user presses the right-arrow key or 'D' key.	Player picture box successfully moves to the right on the form and the player image faces right when the right-arrow key or 'D' key is pressed. **[PASSED]**	N/A	p99	18
41	Player will move down the form when the user presses the down-arrow key or 'S' key.	Player picture box successfully moves down the form and the player image faces down when the down-arrow key or 'S' key is pressed. **[PASSED]**	N/A	p99	18
42	Player will move to the left on the form when the user presses the left-arrow key or 'A' key.	Player picture box successfully moves to the left on the form and the player image faces left when the left-arrow key or 'A' key is pressed. **[PASSED]**	N/A	p99	18

43	The player should not be able to walk over the labels and objects in the background. The player should stop moving when they reach an invisible boundary.	Player is able to move across the invisible boundaries. **[FAILED]**	The invisible walls may not be working as the player moves multiple pixels at a time, so the boundaries should be thick enough to encompass a range of co-ordinates.	p100	19
44	A bullet picture box should appear and travel from the gun in the direction that the player is facing after the user has pressed the space bar key. A gun shot sound effect should also play.	A bullet generates correctly when the space bar key is pressed. The gun shot sound effect also plays in time. **[PASSED]**	N/A	p101	20
45	The ammo in the gun clip should decrement when the user presses the space bar key.	The gun clip ammo label decrements when the weapon is fired. **[PASSED]**	N/A	p101	20/24
46	When the bullet fired from the player's gun collides with the zombies, they are expected to disappear from the form and new zombies should appear at a random point.	When the bullet collides with the zombies, they are disposed of from the form appropriately. Simultaneously, another zombie spawns at a random point outside of the form. **[PASSED]**	N/A	p101	20
47	The score label should increment when a zombie gets eliminated.	When a zombie gets eliminated, the score label increments. **[PASSED]**	N/A	p102	20/24
48	When the 'R' key is pressed, a progress bar should appear above the player and should fill with time. The reload sound effect should also play, as well as, relocating the ammo from the ammo pouch to	The reload progress bar appears above the player and reload sound effect for the desert eagle pistol plays when the 'R' key is pressed. The ammo successfully transfers	N/A	p102	21

	the gun clip, ready to be fired.	into the gun clip ready to fire. **[PASSED]**			
49	When the player reloads the ammo from the ammo pouch should be relocated to the gun clip.	The ammo label successfully gets updated. **[PASSED]**	N/A	p102	21/24
50	The zombies should be generated and appear at random locations on the form when the form is loaded.	The zombies are placed at a random point on the form, however, sometimes, they spawn on top of objects or even the player. **[FAILED]**	I will change where the zombies spawn to outside the form and they will move into the play area, ensuring they do not spawn on the objects in the background.	p103	22
51	The ammo refill box should appear at a random location when the player has no ammo in their utility.	The ammo refill box spawns at a random point when the player's ammo reaches 0. **[PASSED]**	N/A	p103	22
52	The player should collect the ammo when they collide with the refill box. The ammo should be stored in the ammo pouch.	When the player collides with the ammo refill box, the ammo label updates successfully. **[PASSED]**	N/A	p103	22/24
53	All zombie picture boxes on the screen should move towards the player, regardless if the player is moving or not.	All the zombies move simultaneously towards the player. **[PASSED]**	N/A	p104	23
54	When the zombies collides with the player picture box, the player's health bar should decrement by the amount of damage dealt by the zombie.	The player's health bar decreases by the damage amount dealt by the zombie when they collide. **[PASSED]**	N/A	p104	23
55	The health progress bar is expected to change to red when reaching critical levels (e.g. <20hp).	The health bar changes to red when under 20hp. **[PASSED]**	N/A	p105	25

56	The ammo label should change to orange when at critical levels and then to red when it reaches 0.	The ammo label changes colour to orange when ammo in clip is under 2. It changes to red when clip ammo reaches 0. **[PASSED]**	N/A	p105	25
57	The user should be able to freeze the game when the 'P' key is pressed. Main menu and help buttons should appear and function appropriately and the user should be able to return to the game by pressing any other key.	The game freezes when the 'P' key is pressed, and the user is able to return back to the game by pressing any other key. The main menu button takes the user back to the main menu, however, after the help button is pressed the user cannot go back and resume the game. **[FAILED]**	In order to fix this issue, I must find a way to return the focus back to the Game Level One Form after the user closes the Help Guide so the user can press any key from there and resume the game. Alternatively, I can add another button with the functionality of pausing and unpausing the game.	p105	26
58	The main menu button, when clicked by the user, should take them to the main menu form.	When the main menu button is clicked, when game is paused or from game over screen, takes the user to the main menu form. **[PASSED]**	N/A	p106	26/28
59	The help/guide button, when clicked, should open the help page while keeping the current form open in the background.	When the help button is clicked, when game is paused or from game over screen, opens the help form while game level active in background. **[PASSED]**	N/A	p106	26/28
60	The game over label, retry, main menu and help buttons should appear along the centre of the form, when the	The play again, main menu and help buttons appear successfully during the game over event, however only a part of the game over	The size property of the label must be adjusted so all the text can be displayed.	p107	28

	player's health reaches 0hp.	label appears. Additionally, the user can still move the player and fire the weapon using key inputs. **[FAILED]**	Find a way to stop the Form Key Down and Key Up events entirely so the user's actions don't execute any parts of the code.		
61	The 'Play Again?' button, when clicked, should restart the level again.	When the retry button is clicked, the game level restarts as desired. **[PASSED]**	N/A	p108	28
62	The game's difficulty is increase when the user's score reaches 50. (More zombies appear on the form, the zombie's speed and strength increases).	When the player's score reaches 50, the stronger and faster zombies spawn on to the form successfully, 4 at a time. **[PASSED]**	N/A	p108	29

Test Evidence:

38.

The Game Level One Form successfully loads with the appropriate controls.

39.

The player's picture changes successfully and moves up the form when the up-arrow key or 'W' key is pressed in the FrmGameLevelOne_KeyDown Event.

40.

The player's picture changes successfully and moves to the right of the form when the right-arrow key or 'D' key is pressed in the FrmGameLevelOne_KeyDown Event.

41.

The player's picture changes successfully and moves down the form when the down-arrow key or 'S' key is pressed in the FrmGameLevelOne_KeyDown Event.

42.

The player's picture changes successfully and moves to the left of the form when the left-arrow key or 'A' key is pressed in the FrmGameLevelOne_KeyDown Event.

43.

The user is able to move the player through the invisible boundaries onto the background objects on the form. **[FAILED TEST]**

Corrective Measure Now the invisible walls will encompass a range of co-ordinates to catch the player picture box.

```
//Make top of form, bottom of Score box, Tree #1 and Bin/Bench, invisible walls.
if (y <= 0 || (x <= 185 && (y >= 60 && y <= 70)) || ((x >= 335 && x <= 505) && (y >= 95 && y <= 105)) || (x >= 495 && y <= 85))
{
    invisibleWall = true;
}
```

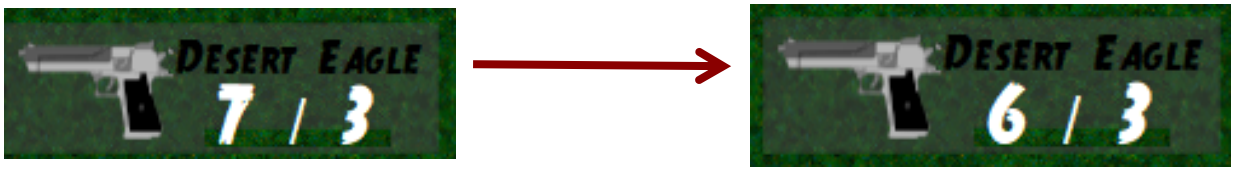
The player picture box cannot pass the invisible wall above it as desired. **[RESOLVED]**

44.

The bullet appears from the player's gun successfully and travels in the same direction the player is facing, as well as the Desert Eagle sound effect playing, when the 'Space Bar' key is pressed in the FrmGameLevelOne_KeyReleased Event.

45.

The ammo in the gun clip decrements successfully when the weapon is fired.



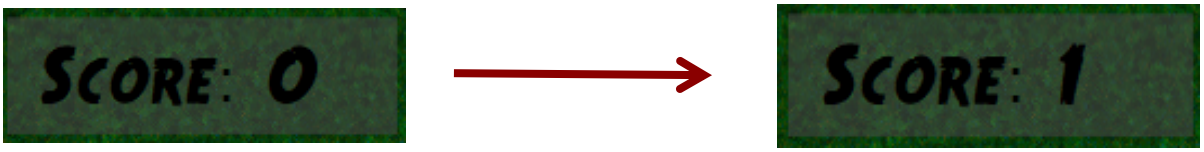
46.

Zombie and bullet dissappears when the bullet collides with it.

New zombie spawns at random point at the same time with the zombie sound effect playing.

47.

The player's score successfully increments every time the user eliminates a zombie.



48.

The Reload progress bar appears above the player, moving with the player, when the 'R' key is pressed in the FrmGameLevelOne_KeyDown Event.

49.

The ammo from the ammo pouch gets successfully transferred to the gun clip when the Reload event occurs.

50.

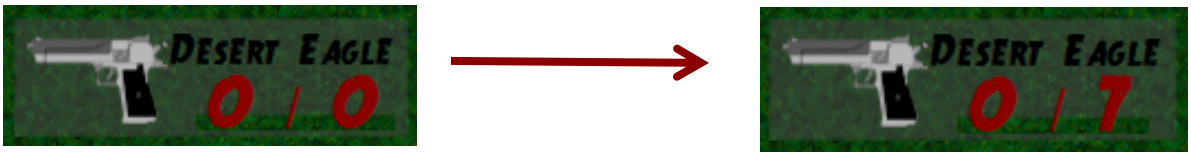
Just like the player, the zombies can move through the invisible boundaries and also can randomly spawn on top of the objects in the background.
[FAILED TEST]

51.

The Ammunition Refill Box successfully, spawns at a random point on the form (Within the play area) when the player's ammo in the gun clip and ammo pouch reaches zero.

52.

When the player collides with the Ammunition Refill Box, the ammo gets collected and is stored in the ammo pouch. The ammo box disappears as well.

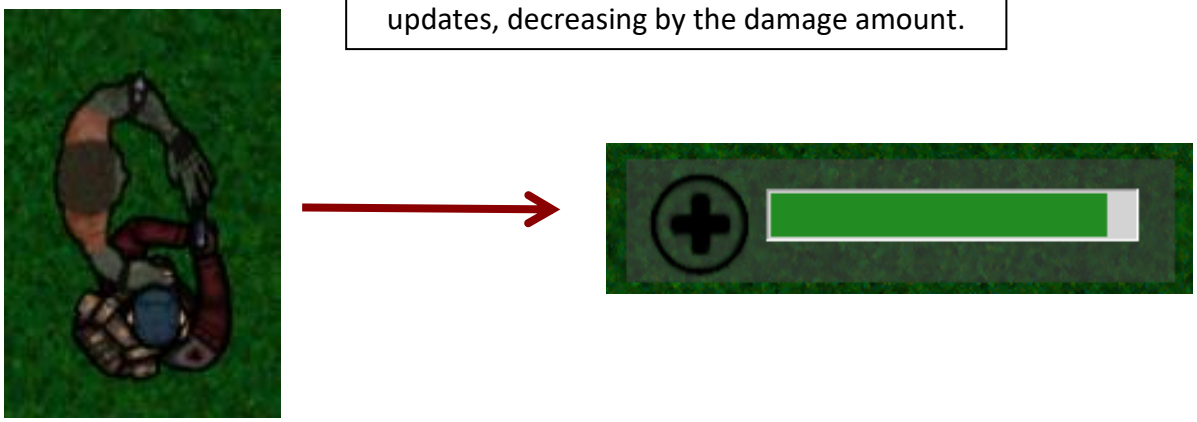


53.

As time passes on, all the zombies successfully move towards the player to attack them.

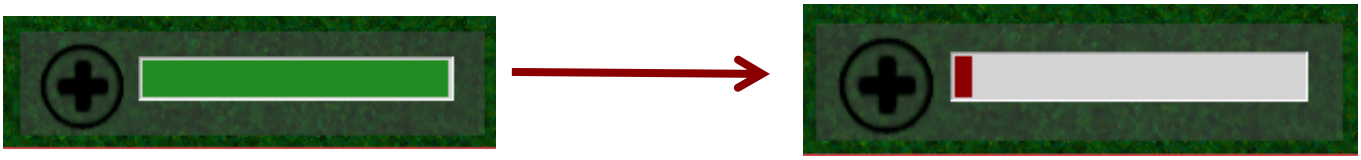
54.

When the zombies make contact with the player, the health progress bar successfully updates, decreasing by the damage amount.



55.

When the player's health drops below 20, the progress bar's fore colour successfully changes to red.



56.

When the ammo in the player's gun magazine drops below 2, the label's fore colour successfully changes to orange.

When the ammo in the player's gun magazine drops to zero, the label's fore colour successfully changes to red.

57.

The Pause Menu (Home and Help buttons) is successfully drawn dynamically on to the form when the 'P' key is pressed in the FrmGameLevelOne_KeyDown Event.

All controls are frozen when the game is paused. The user must press another key to resume the game.

However, after the help button is pressed, the user cannot resume the game and the form continues to show the Pause Menu.

[FAILED TEST]

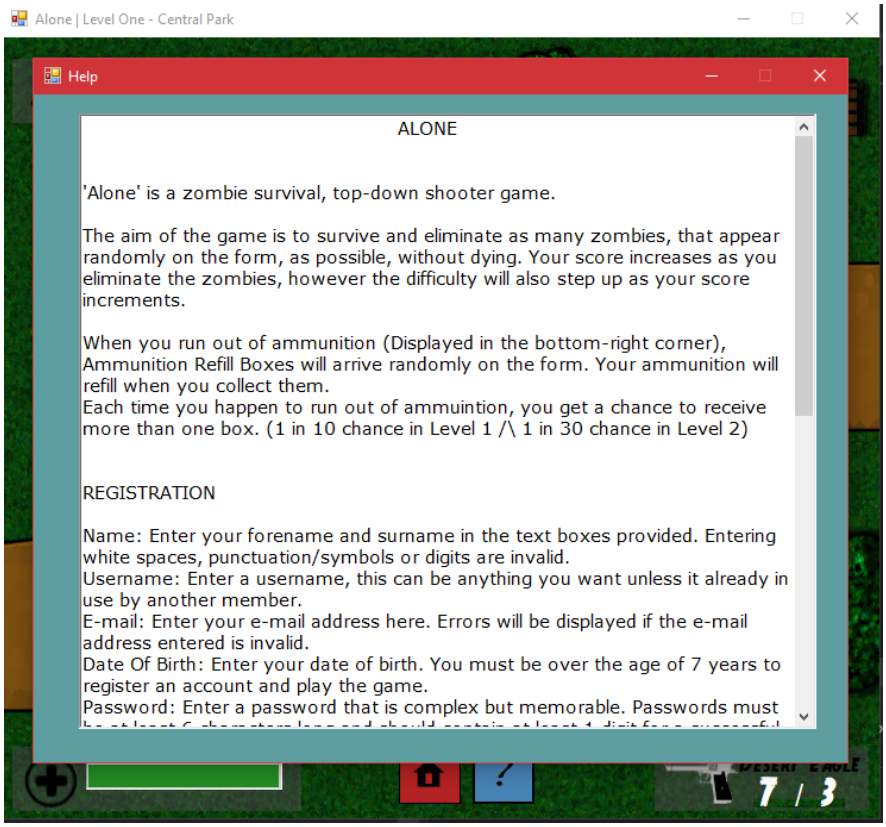
[UNRESOLVED]

58.

The Main Menu Form successfully loads in the BtnHome_Click Event. Enabled Play button as user is logged in.

59.

The Help Guide Form opens correctly in the BtnHelp_Click Event.



60.

The Game Over Menu (Home, Help and Play Again buttons) is successfully drawn dynamically on to the form when the player's health reaches zero.
All controls are frozen (like the Pause event but the user cannot resume the game).
The Game Over label is not drawn correctly.
[FAILED TEST]

Corrective Measure
Now the invisible walls will encompass a range of co-ordinates to catch the player picture box.

```
//Game Over label and retry button will appear when the player gets eliminated.
lblGameOver = new Label();
lblGameOver.Text = "GAME OVER";
lblGameOver.Font = new Font("OCR A Extended", 60, FontStyle.Bold);
lblGameOver.BackColor = Color.Maroon;
lblGameOver.Location = new Point(150, 200);
lblGameOver.Size = new Size(476, 83);
lblGameOver.Visible = false;
this.Controls.Add(lblGameOver);
lblGameOver.BringToFront();
```

The Game Over Label now draws on to the Form correctly.
[RESOLVED]

61.

The BtnPlayAgain_Click Event works correctly by restarting the Game Level One Form.

Play Again?

62.

Once the player's score reaches 50, the difficulty steps up. More zombies spawn in the game who are faster and deal more damage.

Game Level Two

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
63	All the controls (e.g. Player picture box, Zombie picture boxes) should be drawn onto the form at runtime.	All the appropriate controls are drawn at the set locations correctly. **[PASSED]**	N/A	p113	17
64	Player will move up the form when the user presses the up-arrow key or 'W' key.	Player picture box successfully moves up the form and player image faces up when the up-arrow key or 'W' key is pressed. **[PASSED]**	N/A	p114	18
65	Player will move to the right on the form when the user presses the right-arrow key or 'D' key.	Player picture box successfully moves to the right on the form and the player image faces right when the right-arrow key or 'D' key is pressed. **[PASSED]**	N/A	p114	18
66	Player will move down the form when the user presses the down-arrow key or 'S' key.	Player picture box successfully moves down the form and the player image faces down when the down-arrow key or 'S' key is pressed. **[PASSED]**	N/A	p114	18
67	Player will move to the left on the form when the user presses the left-arrow key or 'A' key.	Player picture box successfully moves to the left on the form and the player image faces left when the left-arrow key or 'A' key is pressed. **[PASSED]**	N/A	p115	18

68	The player should not be able to walk over the labels and objects in the background. The player should stop moving when they reach an invisible boundary.	Player is not able to move across the invisible boundaries. **[PASSED]**	N/A	p115	19
69	A bullet picture box should appear and travel from the gun in the direction that the player is facing after the user has pressed the space bar key. A gun shot sound effect should also play.	A bullet generates correctly when the space bar key is pressed. The gun shot sound effect also plays in time. **[PASSED]**	N/A	p116	20
70	The ammo in the gun clip should decrement when the user presses the space bar key.	The gun clip ammo label decrements when the weapon is fired. **[PASSED]**	N/A	p116	20/24
71	When the bullet fired from the player's gun collides with the zombies, their health should decrease by the damage dealt by the bullet.	When the bullet collides with the zombies, the health for all the zombies on the form decreases by the damage dealt by the bullet. This forces to shoot again to finally kill the zombies. **[PASSED]**	N/A	p116	20
72	The score label should increment when a zombie gets eliminated.	When a zombie gets eliminated, the score label increments. **[PASSED]**	N/A	p117	20/24
73	When the 'R' key is pressed, a progress bar should appear above the player and should fill with time. The reload sound effect should also play, as well as, relocating the ammo from the ammo pouch to	The reload progress bar appears above the player and reload sound effect for the desert eagle pistol plays when the 'R' key is pressed. The ammo successfully transfers into the gun clip ready to fire.	N/A	p117	21

	the gun clip, ready to be fired.	**[PASSED]**			
74	When the player reloads the ammo from the ammo pouch should be relocated to the gun clip.	The ammo label successfully gets updated. **[PASSED]**	N/A	p117	21/24
75	The zombies should be generated and appear at random locations on the form when the form is loaded.	The zombies are placed at a random point on the form, however, sometimes, they spawn on top of objects or even the player. **[PASSED]**	N/A	p118	22
76	The ammo refill box should appear at a random location when the player has no ammo in their utility.	The ammo refill box spawns at a random point when the player's ammo reaches 0. **[PASSED]**	N/A	p118	22
77	The player should collect the ammo when they collide with the refill box. The ammo should be stored in the ammo pouch.	When the player collides with the ammo refill box, the ammo label updates successfully. **[PASSED]**	N/A	p119	22/24
78	All zombie picture boxes on the screen should move towards the player, regardless if the player is moving or not.	All the zombies move simultaneously towards the player. **[PASSED]**	N/A	p119	23
79	When the zombies collides with the player picture box, the player's health bar should decrement by the amount of damage dealt by the zombie.	The player's health bar decreases by the damage amount dealt by the zombie when they collide. **[PASSED]**	N/A	p120	23
80	The health progress bar is expected to change to red when reaching critical levels (e.g. <20hp).	The health bar changes to red when under 20hp. **[PASSED]**	N/A	p120	25
	The ammo label should change to orange when	The ammo label changes colour to			

81	at critical levels and then to red when it reaches 0.	orange when ammo in clip is under 5. It changes to red when clip ammo reaches 0. **[PASSED]**	N/A	p120	25
82	The user should be able to freeze the game when the 'P' key is pressed. Main menu and help buttons should appear and function appropriately and the user should be able to return back to the game by pressing any other key.	The game freezes when the 'P' key is pressed and the user is able to return back to the game by pressing any other key. The main menu button takes the user back to the main menu, however, after the help button is pressed the user cannot go back to the game. **[FAILED]**	In order to fix this issue, I must find a way to return the focus back to the Game Level Two Form after the user closes the Help Guide so the user can press any key from there and resume the game. Alternatively, I can add another button with the functionality of pausing and unpausing the game.	p121	26
83	The main menu button, when clicked by the user, should take them to the main menu form.	When the main menu button is clicked, when game is paused or from game over screen, takes the user to the main menu form. **[PASSED]**	N/A	p121	26/28
84	The help/guide button, when clicked, should open the help page while keeping the current form open in the background.	When the help button is clicked, when game is paused or from game over screen, opens the help form while game level active in background. **[PASSED]**	N/A	p122	26/28
85	The game over label, retry, main menu and help buttons should appear along the centre of the form, when the player's health reaches 0hp.	All the appropriates controls are drawn correctly during the game over event. However, the user can still move the player and fire the weapon using key inputs.	Find a way to stop the Form Key Down and Key Up events entirely so the user's actions don't execute any parts of the code.	p122	28

		[FAILED]			
86	The 'Play Again?' button, when clicked, should restart the level again.	When the retry button is clicked, the game level restarts as desired. [PASSED]	N/A	p124	28
87	The game's difficulty is increase when the user's score reaches 20. (More zombies appear on the form, the zombie's speed and strength increases).	When the player's score reaches 20, the stronger and faster zombies spawn on to the form successfully, 4 at a time. [PASSED]	N/A	p124	32

Test Evidence:

63.

The Game Level Two Form successfully loads with the appropriate controls.

64.

The player's picture changes successfully and moves up the form when the up-arrow key or 'W' key is pressed in the FrmGameLevelTwo_KeyDown Event.

65.

The player's picture changes successfully and moves to the right of the form when the right-arrow key or 'D' key is pressed in the FrmGameLevelTwo_KeyDown Event.

66.

The player's picture changes successfully and moves down the form when the down-arrow key or 'S' key is pressed in the FrmGameLevelTwo_KeyDown Event.

67.

The player's picture changes successfully and moves to the left of the form when the left-arrow key or 'A' key is pressed in the FrmGameLevelTwo_KeyDown Event.

68.

The user is successfully unable to move the player through the invisible boundaries onto the background objects on the form.

Bottom-Left Building

Abandoned Car

Trees

Top-Right Building

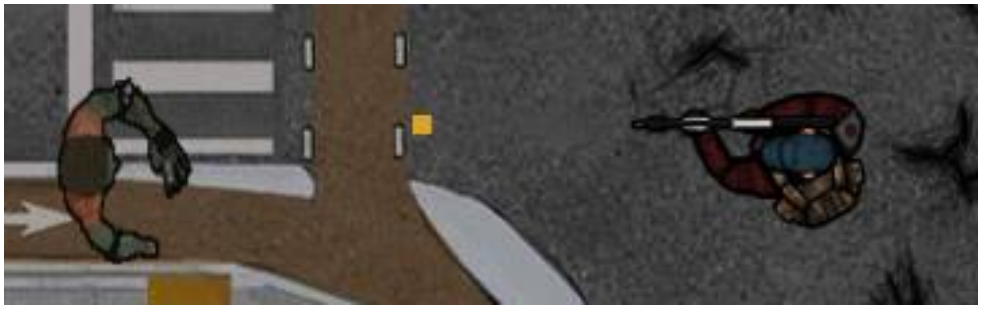
Top-Left Building

Weapon Slot Box

Health Box

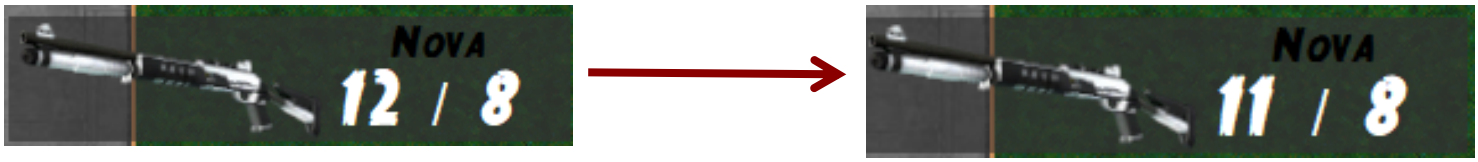
69.

The bullet appears from the player's gun successfully and travels in the same direction the player is facing, as well as the Nova shotgun sound effect playing, when the 'Space Bar' key is pressed in the FrmGameLevelTwo_KeyReleased Event.



70.

The ammo in the gun clip decrements successfully when the weapon is fired.



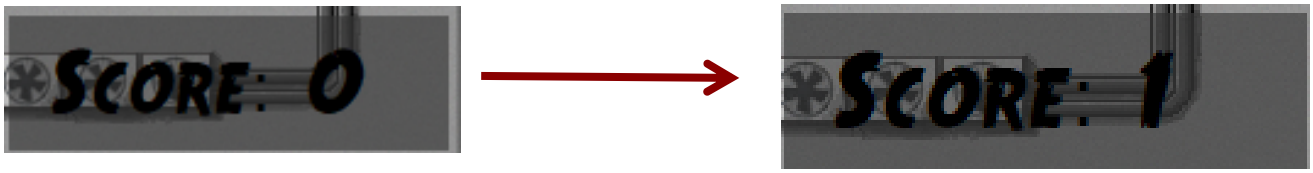
71.

The zombies have to be shot twice to eliminate them. They dissappear on collision with the second bullet. The bullet dissapperears when they collide with the zombies.

New zombie spawns at random point at the same time with the zombie sound effect playing.

72.

The player's score successfully increments every time the user eliminates a zombie.

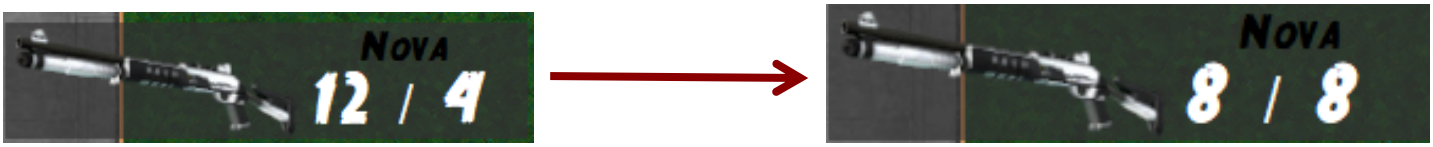


73.

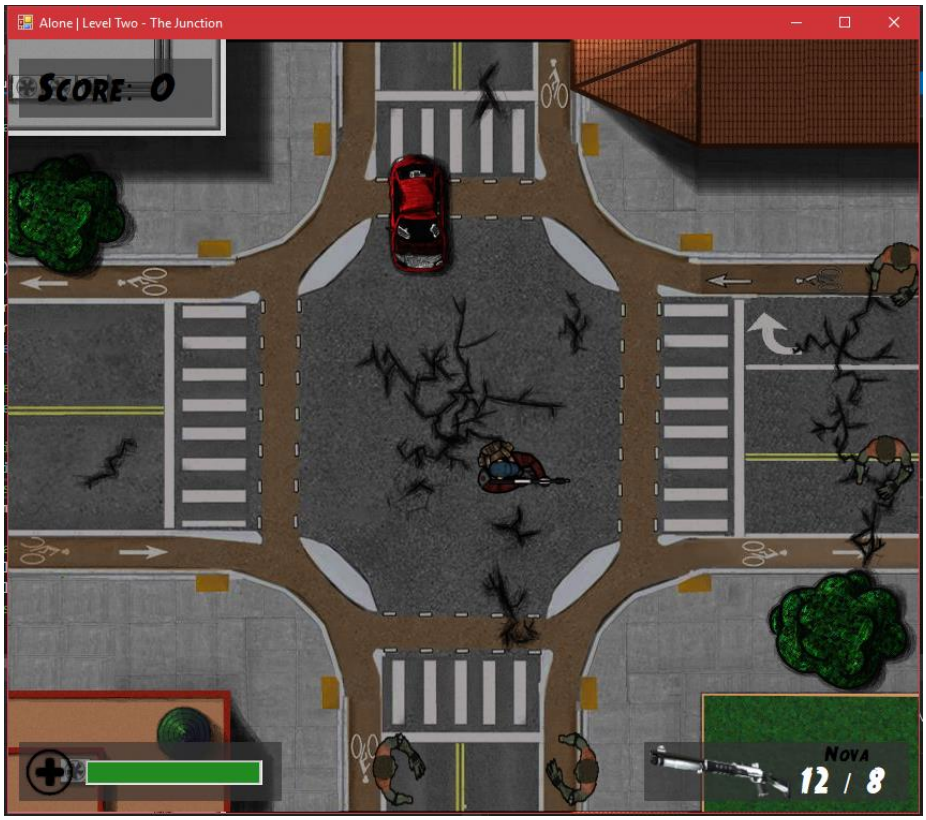
The Reload progress bar appears above the player, moving with the player, when the 'R' key is pressed in the FrmGameLevelTwo_KeyDown Event.

74.

The ammo from the ammo pouch gets successfully transferred to the weapon clip when the Reload event occurs.



75.



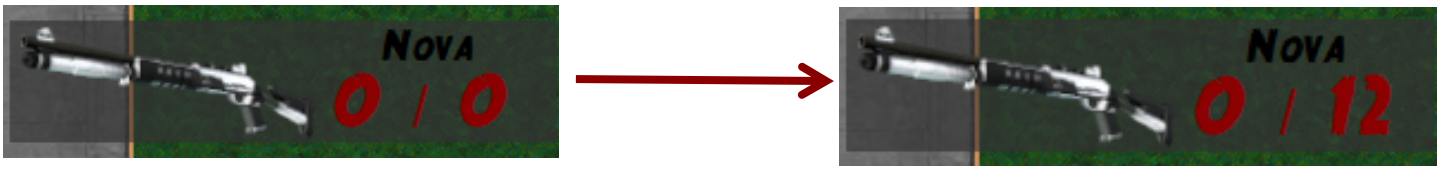
The zombies succesfully spawn on the form at random co-ordinates within the play area.

76.

The Ammunition Refill Box successfully, spawns at a random point on the form (Within the play area) when the player's ammo in the weapon clip and ammo pouch reaches zero.

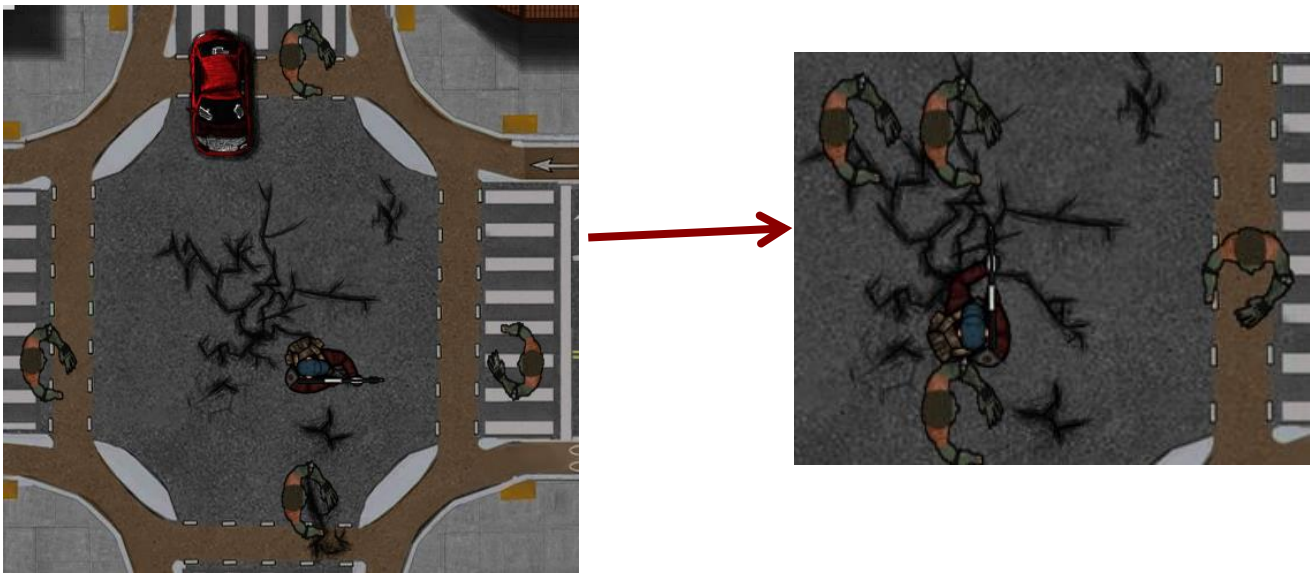
77.

When the player collides with the Ammunition Refill Box, the ammo gets collected and is stored in the ammo pouch. The ammo box disappears as well.



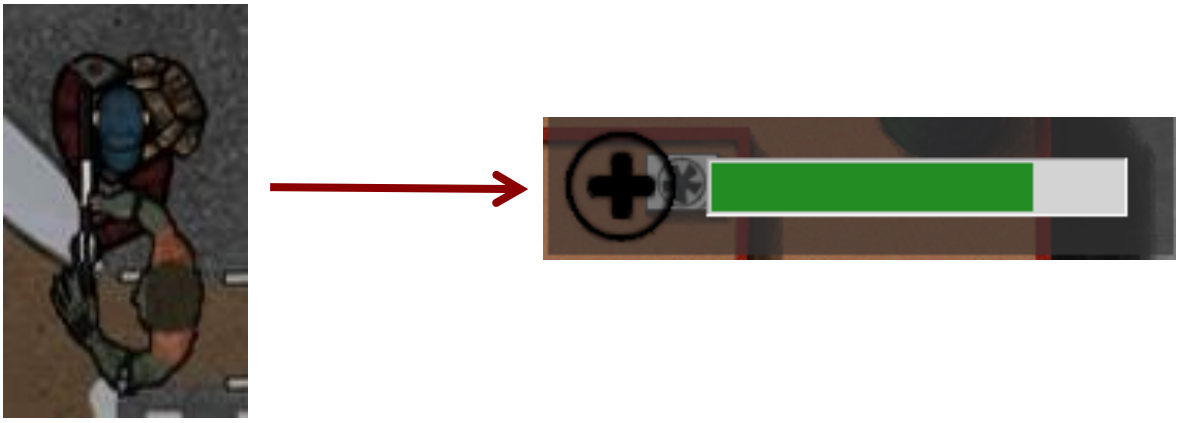
78.

As time passes on, all the zombies successfully move towards the player to attack them.



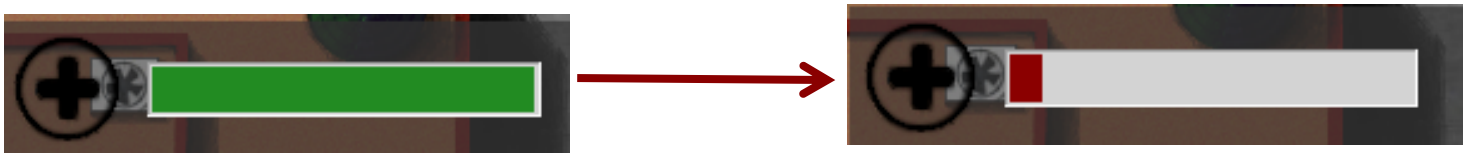
79.

When the zombies make contact with the player, the health progress bar successfully updates, decreasing by the damage amount.



80.

When the player's health drops below 20, the progress bar's fore colour successfully changes to red.



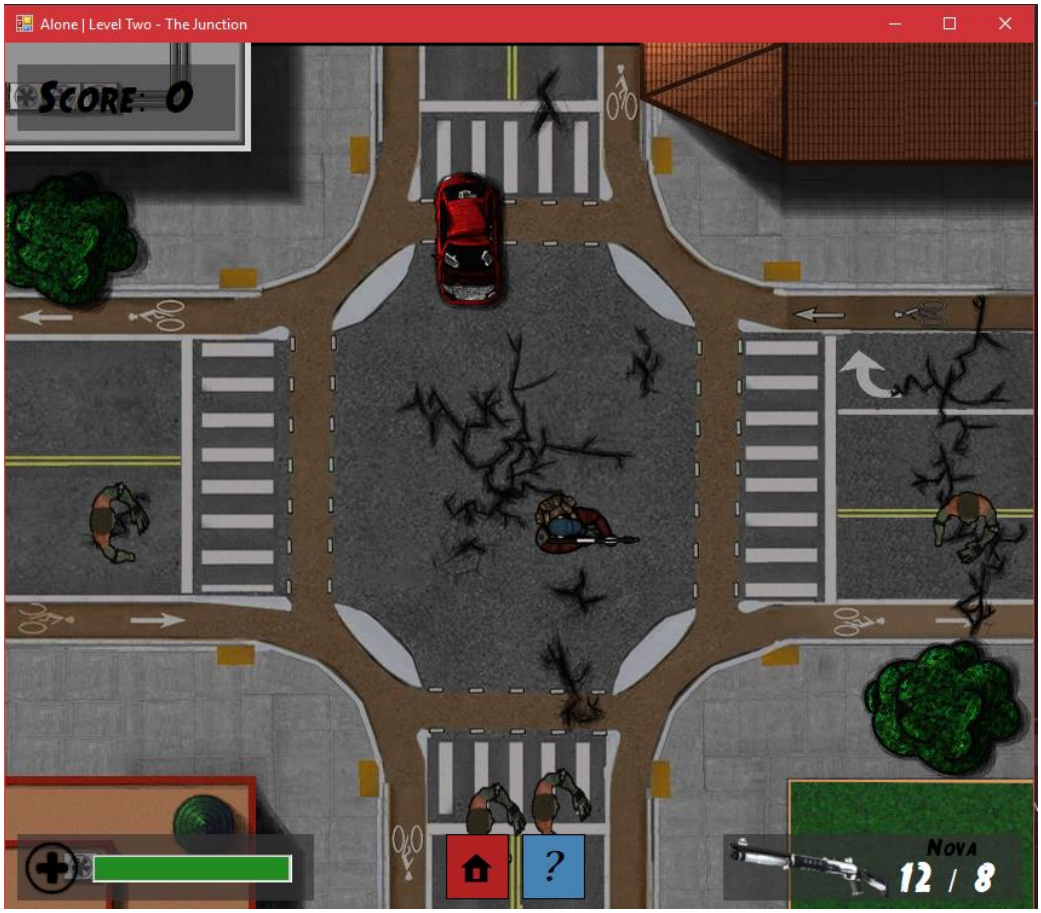
81.

When the ammo in the player's gun magazine drops below 5, the label's fore colour successfully changes to orange.

When the ammo in the player's gun magazine drops to zero, the label's fore colour successfully changes to red.

82.

The Pause Menu (Home and Help buttons) is successfully drawn dynamically on to the form when the ‘P’ key is pressed in the FrmGameLevelTwo_KeyDown Event.
All controls are frozen when the game is paused. The user must press another key to resume the game.
However, after the help button is pressed, the user cannot resume the game and the form continues to show the Pause Menu.
[FAILED TEST]
[UNRESOLVED]

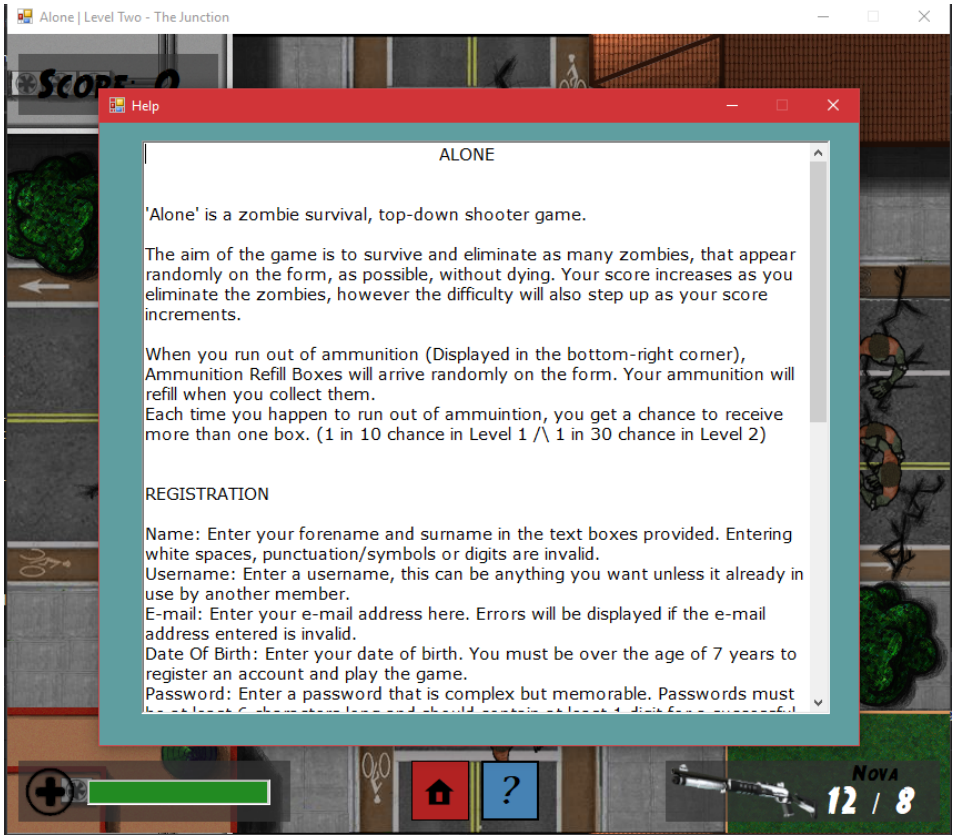


83.

The Main Menu Form successfully loads in the BtnHome_Click Event. Enabled Play button as user is logged in.

84.

The Help Guide Form opens correctly in the BtnHelp_Click Event.



85.

The Game Over Menu (Game Over label, Home, Help and Play Again buttons) are successfully drawn dynamically on to the form when the player's health reaches zero.
All controls are frozen until the user presses a movement key or the 'Space Bar' key. Then the player will move, and weapon is fired.
[FAILED TEST]

'Space Bar' key press results in the weapon being fired and bullet being created during the Game Over event.

Corrective Measure

During the Game Over event, the key down and key up events are removed so any user inputs are ignored.

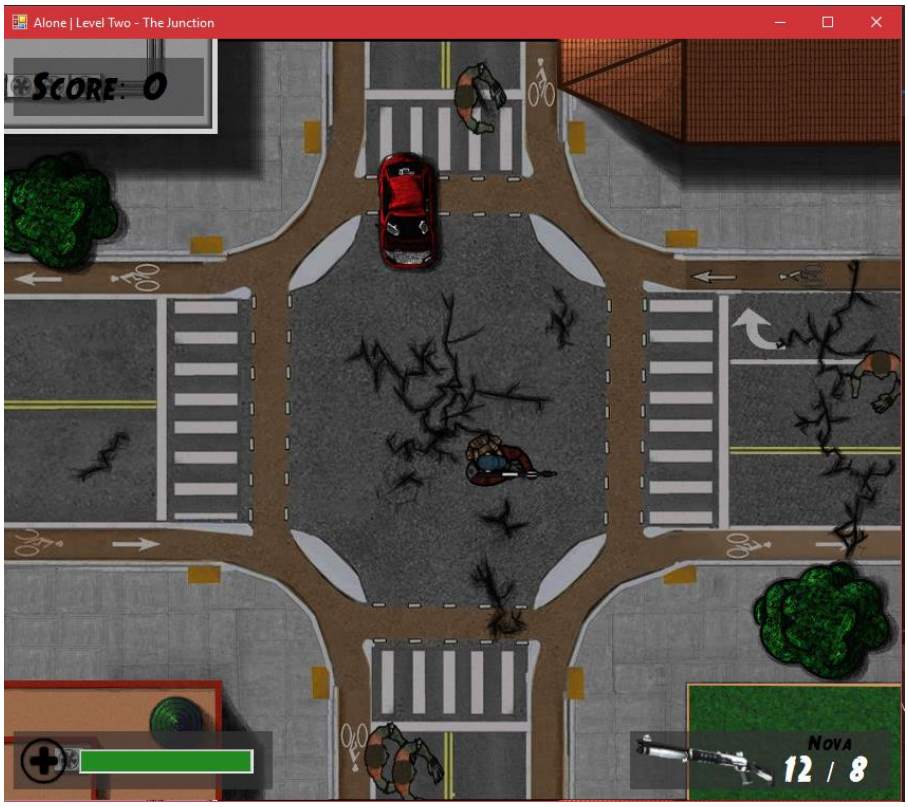
```
//DEAD - GAME OVER
//-Stop timer and remove key events then change to game over screen.
tmrRefreshRate.Stop();
lblGameOver.Visible = true;
btnPlayAgain.Visible = true;
btnHome.Visible = true;
btnHelp.Visible = true;

this.KeyDown -= FrmGameLevel2_KeyDown;
this.KeyUp -= FrmGameLevel2_KeyReleased;
```

The game is permanently frozen during the Game Over event and any key presses are ignored.

[RESOLVED]

86. The BtnPlayAgain_Click Event works correctly by restarting the Game Level Two Form.

Play Again?



87.

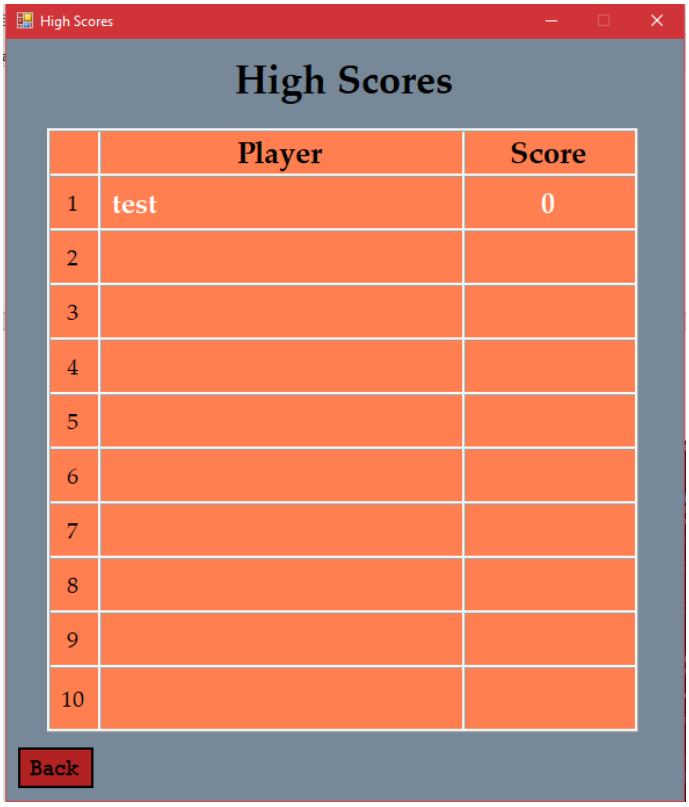
Once the player's score reaches 20, the difficulty steps up. More zombies spawn in the game who are faster and deal more damage.

High Scores

Test No.	Expected Outcome	Actual Outcome	Corrective Measures	Test Evidence	User Requirement
88	The High Scores table should load with the appropriate controls (e.g. Title, Table and button).	The High Scores Form loads correctly with all the appropriate controls. **[PASSED]**	N/A	p126	34
89	The high scores should be displayed in the last column in descending order.	The high scores are displayed in descending order in the last column. **[PASSED]**	N/A	p126	34
90	The usernames should be displayed in the second column, beside their corresponding high score.	The usernames are not successfully displayed in the second column. **[FAILED]**	Find where the error is occurring by using breakpoints in the code.	p127	34
91	The high scores should update when the user achieves a new high score.	The high scores update when the user achieves a new high score. **[PASSED]**	N/A	p128	34
92	The currently logged in user's username should be highlighted (Bold) if present on the leader board.	The currently logged in user's username is highlighted on the leader board. **[PASSED]**	N/A	p128	35
93	The back button, once clicked, should transfer the user back to the game level select screen	The back button successfully transports the user to the Game Level Select screen. **[PASSED]**	N/A	p129	36

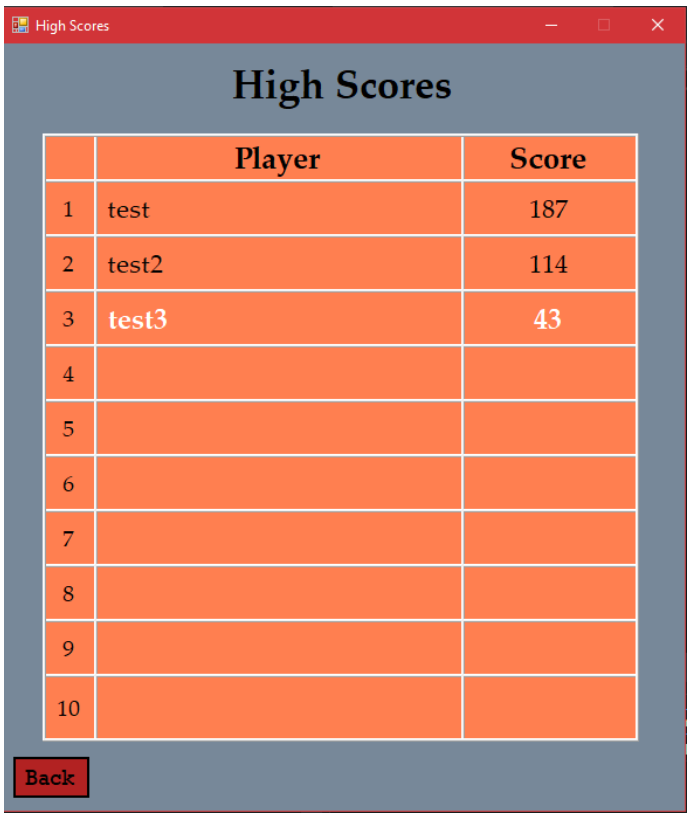
Test Evidence:

88.



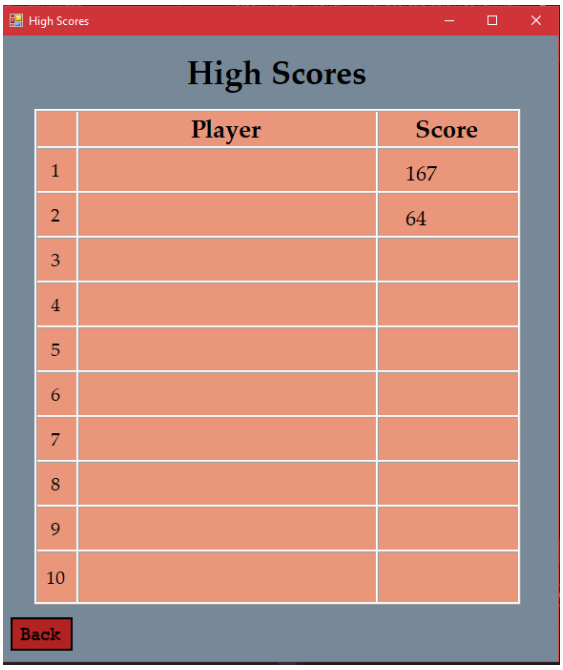
The High Scores Form successfully loads with the appropriate controls.

89.



The high scores of the players are displayed in descending order down the last column.

90.



The usernames of the players are not displayed down the second column.

[FAILED TEST]

Corrective Measure

Breakpoints helped me to pinpoint the problem to a logic error on my part where instead of breaking from the loop when one of the members "==" (Equals) null, I had "!=" (Not Equal to) null.

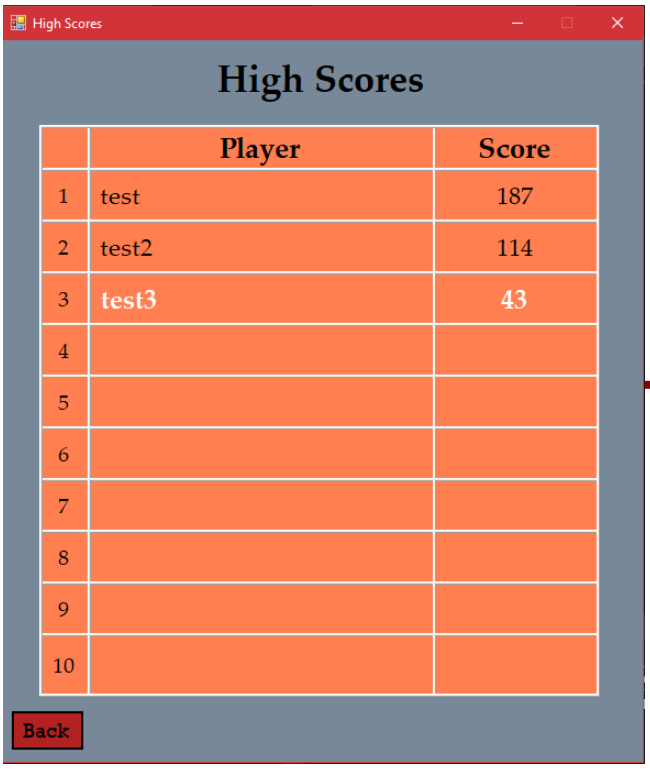
```
//Second for loop to loop through the elements of the members array.
for (int j = 0; j < FrmRegistration.members.Length; j++)
{
    //If a null space is reached in the members array then break as the rest of the array is also empty spaces.
    if (FrmRegistration.members[j] != null)
        break;
```

The usernames of the members now display on to the leader board as desired.

[RESOLVED]

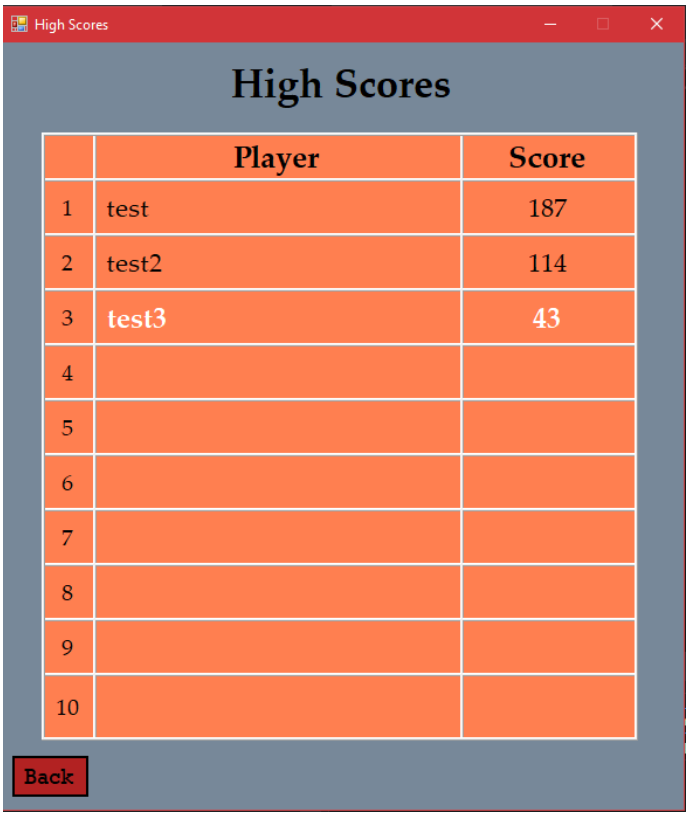
91.

High Scores Leader board updates when opened again.



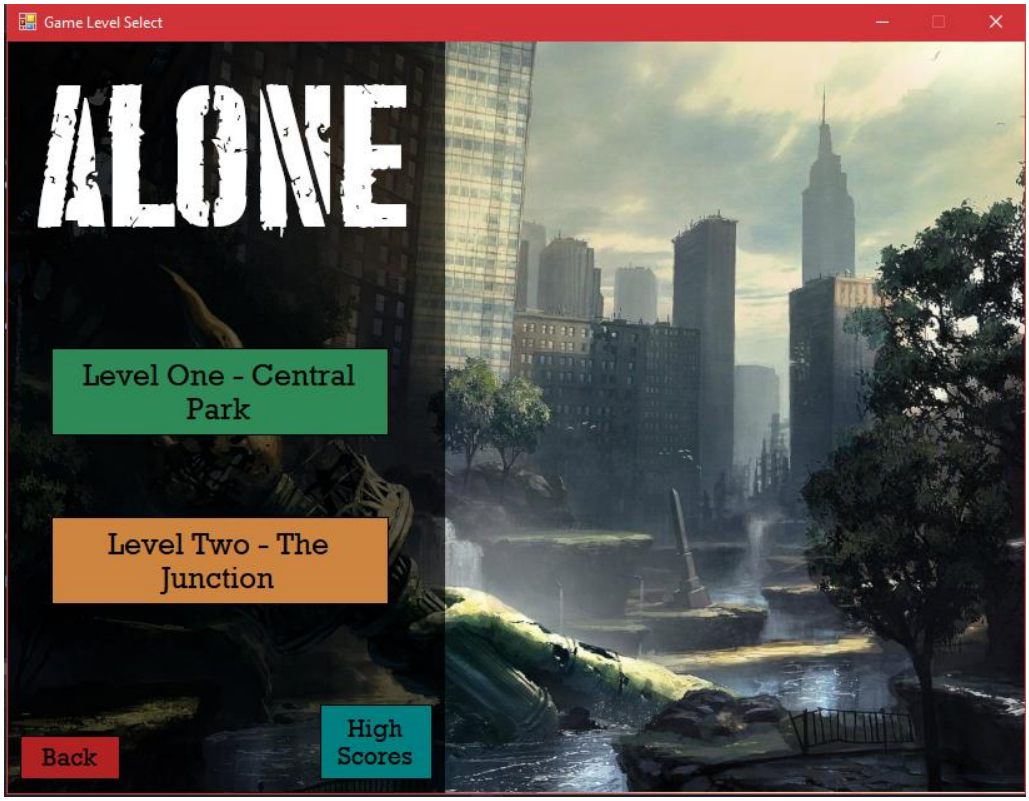
92.

Member 'test3' is currently logged in so their username and score are highlighted.



93.

The BtnBack_Click Event works correctly by returning the user back to Main Menu Form.

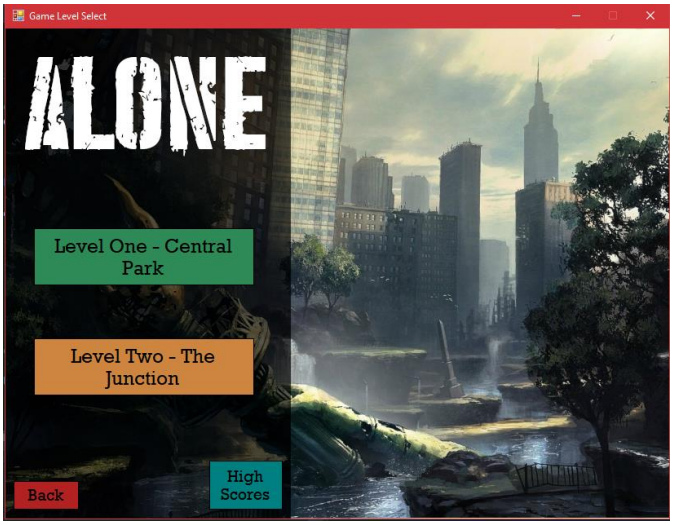


User Guide

Main Menu

The Main Menu is the core of the whole application, where the rest of the application can be accessed. Users are advised to return to the main menu at any time to find the part of the application they want to go to. There are various buttons on the main menu, that are clearly labelled, and colour coded to easily distinguish between them.

If the user has not already registered, they are advised to click the Register button to go to the Registration Page and create an account. Once registered or if already registered before, the user can login in the Login Page.



Once logged in, the user is now able to click the Play button, which will be enabled. Clicking this button, will change the main menu to a level select screen. Here, the user will be greeted with two buttons to transport the user to the 2 different levels of the game, a Back button, to return to the main menu, and a High Scores button, which will take the user to the High Scores table, displaying the top ten players who have achieved the highest score.

There is a Help button on the main menu that, when clicked by the user, will open the Help Guide which will explain to the user the mechanics of the game and controls. There is, finally, an Exit or Quit button that will exit the application when the user is done.

Registration

For a successful registration, all input fields must be completed without errors. Errors will be displayed and updated as the text is entered in text boxes or when the date is changed when entering the user's date of birth. To register, the user must click the Register button to do a final check.

The name text fields must not contain whitespaces, punctuation/symbols or digits. The user should enter a valid first name and surname.

The username text box must not contain a username that is already in use by another member. The user will have to input another username in this case.

The e-mail text box must contain a valid e-mail address (containing '@' and '.' symbols).

The user can display a date and time picker to select their date of birth in the date of birth field. If the user is under 7 years, they cannot register an account and play the game.

The password text box must contain a password with at least 1 digit and at least be 6 characters long. The user is advised to pick a complex but memorable password, so they can easily log in.

There is a taskbar present at the bottom of the Registration page, with the Home, Login, and Help buttons so the user can seamlessly switch between pages.

Login

For a successful log in, all input fields must be completed without errors. Errors will be displayed and when the log in button is clicked by the user.

The username text field should not contain a username that does not exist in the database.

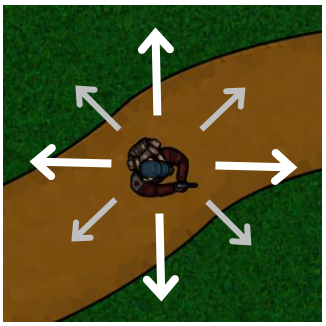
The password text box should contain the password that matches the corresponding username.

Just like the Registration Page, there is a taskbar at the bottom of the page, only with a Register button instead of the Login Button to take the user to the Registration Page.

Help Guide

All pages in the application will include a Help button that will open the Help Guide from anywhere, so the user can get help with any query or problem they encounter. The Help Guide contains a text box with a scroll bar so the user can access rest of the guide. The guide will explain how to register and login and explains what the game entails and how to play it.

Game Level One



To play the game, the user must avoid the zombies to survive, who will be always moving towards the player. The user can move the player up, down, left and right or diagonally by pressing a combination of the arrow keys or the 'WASD' keys.

The user can shoot their weapon by pressing the space bar key and they are advised to aim their player towards the zombie to successfully eliminate them. The user can reload their weapon by pressing the 'R' key. This action will transfer any ammunition stored in the ammo pouch to the magazine of the weapon to be fired. When the user runs out of ammunition, a Ammunition Refill Box will spawn at a random location on the form, collecting this will refill the ammo pouch.

The Desert Eagle handgun is the weapon given to the user in the first level, able to carry 7 bullets at once in the magazine. The enemies are weak in this first level and so one shot from the weapon will eliminate them entirely, making the first level fairly easy, ideal for beginners to learn and get used to the controls and mechanics of the game.

The user will be able to see their score increment every time they eliminate a zombie. Their health bar will decrease when the zombies make contact with the player.

The user can pause the game at any time by pressing the 'P' key and 2 buttons will display on to the screen. One to navigate to the Main Menu screen, the user's score will save when they leave the game levels. The other button will open the Help Guide, here the user can read how to play the game.

When the player's health reaches zero, the Game Over screen is displayed with the same 2 buttons in the pause menu as well as a third button to restart Game Level One. The user's score will be saved when they restart the level.

The difficulty substantially increases the moment the user's score reaches 50, more strong and agile zombies spawn at a time.

Game Level Two

The mechanics of the second level is more or less the same as the first level with the only difference being the second level will be more difficult for more experienced players. The enemies are faster and deal more damage and spawn in larger quantities at a time. There will be more space available in the second level for the user to experiment and strategize.

The Nova shotgun will be given as the weapon to use for the second level which carries very different mechanics compared to the Desert Eagle handgun in the first level. The Nova has a larger magazine capacity as opposed to the Desert Eagle, being able to hold 12 bullets instead of only 7 in level one. The bullets fired from the Nova will be bigger and faster making it easier to aim at the enemies and to compensate for the large play area. The bigger size of the bullet is to mimic the spread of real-life shotgun shells. However, even with the upgrades of the weapon and bullets in level 2, the enemies are twice as strong and so needs twice as much firepower to fully eliminate giving the second level that higher difficulty for experienced users to master.

The second level uses the same controls as mentioned in the first level and the user is able to pause the game any time. When their player's health reaches zero, the Game Over event occurs and the user is offered the chance to play again.

Another difference between Game Level One and Game Level Two is that the difficulty increases when the user's score reaches 20 instead of 50 making it harder to survive from early on.

The Game Level Two screen offers more space to move than the first level as well as increasing the player's speed and bullet speed and size. The design for the background of the second level is more intricate than the first for an aesthetically pleasing and different user experience.

There is an additionally game mechanic added to the second level to balance the increased difficulty, this being that the player will be rewarded bonus health points for every elimination of a zombie. This makes the gameplay more interesting and enjoyable giving the user a chance to survive longer if they strategize well in the game.

High Scores

This Windows Form displays the leader board table showing the top ten users with the top highest scores achieved on the two levels. The high scores can only be reached from the Game Level Select screen after the user has successfully logged in. The leader board will update every time a user achieves a new high score. The currently logged in user's username and high score will be highlighted if they are present on the leader board so the user can easily find their name.

Evaluation

Evaluation

Design Modifications

During the process of designing the backgrounds of the various Windows Forms on Visual Studio and Adobe Photoshop, I made a few design modifications to further improve the aesthetic of the screens and the layout.

On the Registration page, I changed the 'First' and 'Last' labels to be just the initial text present in the Name textboxes when the Form is loaded This offered to be more of convenience as it avoided the hassle of adding more unnecessary controls and further increasing the screen size without ruining the design of the form.

On the Game Level screens, my original design of displaying the player's health as a percentage using a label in Page 12 and 13. I changed this to presenting the user's health through a progress bar which starts off full but would eventually decrease as the game progresses. I believe that this is a more pleasant design than just continually showing the user text after text, which would get mundane after a while. In addition, I altered the top left label to read 'Score:' instead of what I initially had planned 'Kills:' shown in Page 12 and 13. I consider this more preferable to keep to the user requirement of appealing to everyone, including young children who can be easily influenced. Additionally, I managed to stay true to my original map designs for the two game level forms, with my first level going through little to no reform while my second level going through a lot of changes. Firstly, I changed the design of the 'roundabout' concept to a 'road junction' granting more area to play around in for the user.

Test Plan and Testing

The initial test plans and testing helped me a lot to identify errors and crashes early, before the beta testing was carried out. I often found parts of the application that users could easily exploit for their own gain, given the right circumstances for example, playing the game without logging in. These problems were easily resolved and with it, sprung new ways of building my program, for instance, in the case of the earlier example, disabling the Play button until the user is logged in.

The initial test planning was a key ingredient in creating my game. Not only did it help me build the program step by step in a logical order, but it also aided me in understanding how well I was adhering to the original user requirements. The testing table kept me organised as I was developing and testing the program in a logical order. I have clearly highlighted the tests that had passed and the ones that had failed with their respective corrective measure. This ensured that I efficiently kept track of every test failure and easily recall and add to my original ideas to resolve it.

User Requirements

I have strived to meet all the proposed user requirements for my application to be as close to the initial idea of the game, and its contents, as possible. This is a summary of the user requirements and a description of, if and how I have met them:

General

- **Do all parts of the application have an aesthetically pleasing and eye-catching design? :** I have successfully met this user requirement as I have designed all Forms with lots of various colours, appealing to all genders, and various fonts. I have deliberately centred the controls (e.g. Buttons and Labels) on each form to create a neat layout that will catch the user's eyes immediately. I have used a myriad of darker colours to go with the ominous, 'Apocalyptic' Theme set by Monsoon Games.

- **Are all parts of the application user-friendly and use simple language? :** I believe that my game uses very simple language that can be easily understood by my target audience, consisting of people of all ages, each with a unique understanding of the English language.

- **Can the Main Manu and the Help Guide easily be accessed from any point in the application? :** Home and Help buttons are present in each Form so the user can easily return to the Main Menu or open the Help Guide at any point without actually closing the current form they are on. The Registration and Login pages each incorporate a taskbar at the bottom of the page including the Home and Help buttons. On the Game Levels, the Help button can be accessed by pausing the game or from the Game Over screen. The Home and Help button are clearly and consistently colour coded for the user to easily locate them.

Registration and Login

- **Can the user create and register an account? :** I feel that this requirement has been met as, through rigorous testing (See Tests 16 – 27 from Page 78 onwards), I have concluded that the registration process functions flawlessly. The form clearly details all the information required to create and register an account and become a member. Every input field has a validation criterion and not meeting these will result in errors with relevant error messages being displayed (See Test Evidence 17 on Page 81).

- **Can pre-existing users login to their account? :** Through the various tests I have performed, I am confident in saying that the Login Form functions correctly. Any pre-existing user will be saved in the file and so can log in without any problems or crashes by providing their correct username and password. Errors with messages are properly placed if any of the user input does not match existing member profiles. This is evident in Test Evidence 29 on Page 88.

  One of the responses from the questionnaire, comments on how there is no feature to help users who have forgotten their usernames and/or passwords and so will be stuck on the Login

page. I believe this is a valid and very constructive feedback and I plan to include this feature in future development.

Game Levels

- **Can the user move their player and fire their weapon? :** The user can easily move their character around the play area using the arrow keys or the 'WASD' keys. The player moves very smoothly across the form background without any problems or lagging, effortlessly changing the image of the player to face the direction its traveling. (See Tests 39 – 43 [Level One] and Tests 64 – 68 [Level Two]).

  I can confidently say that the user can successfully fire their weapon using the 'Space Bar' key on both levels, causing a bullet to be created and move in the direction the player is facing. To illustrate, see Tests 44 and 69. The gun shot sounds for each level is different due to the different weapons present. I added this to increase the immersion of the game making it more realistic and further creating a clear disparity between the two levels.

  A recurring graphical minor issue that my respondents discovered during the peer evaluation of my game was how the images of the player, zombies and even the ammunition refill boxes, disappear then reappear a moment later when objects are spawned or removed from the Form. I intend to patch this bug in future development to further increase the immersion of the game.

- **Do both the levels function correctly as initially intended? :** I believe that I have achieved this goal as the end product is exactly, if not more, than my original idea of the game. Both of the game levels function identically to how I had envisioned them, and I am very proud of the results. The majority of the testing for the two levels were very successful and the ones that did fail, were easily resolved, with the only exception of one issue.

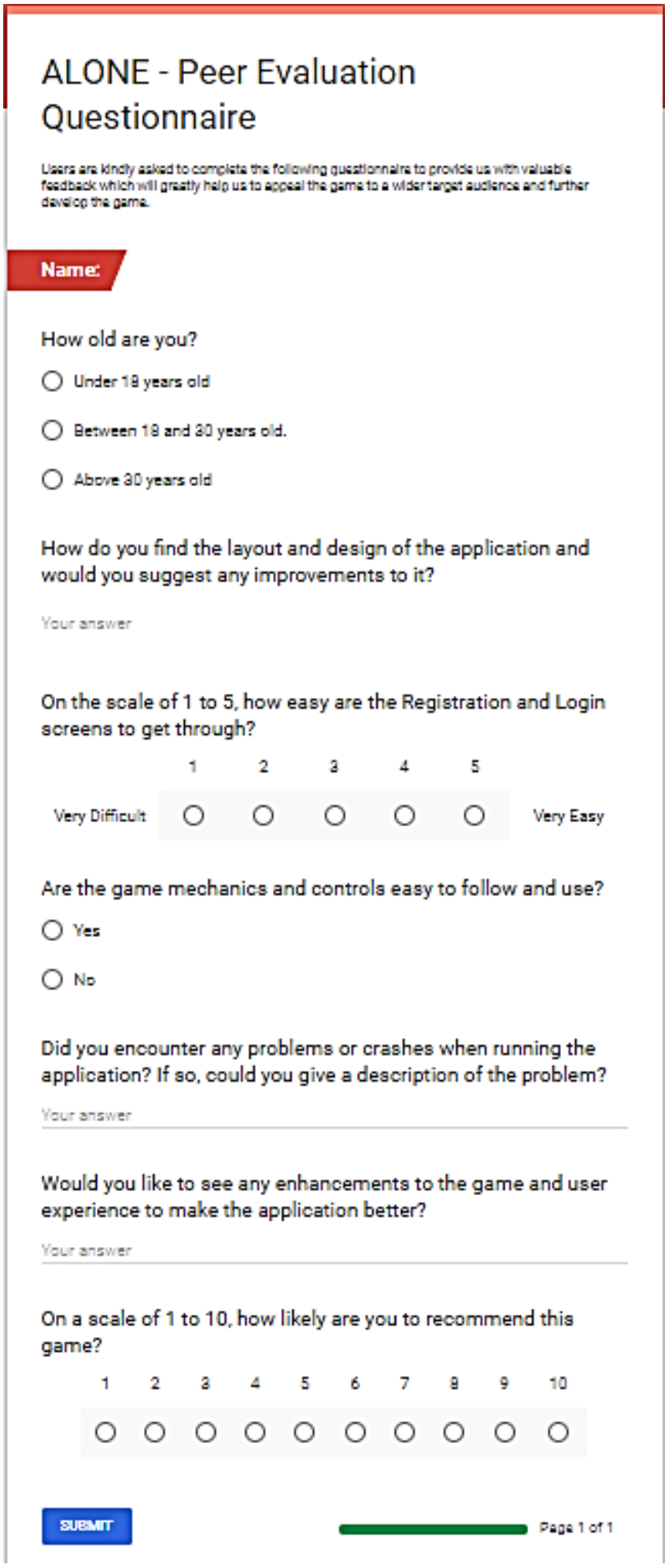
  Aforementioned, the issue that remains unresolved, on both levels, is the problem where, once the user opens and then closes the Help Guide from the pause menu from any of the two levels, they cannot resume their game from the point they paused as the focus has not been returned to the game. In summary, the only way for the user to return to the game is to start over by clicking the Main Menu button from the pause screen and proceeding from there. I planned to find a way to rectify this by looking for help from an outside and experienced source, for example a teacher or help from the internet. Unfortunately, I could not find a solution. I did highlight another solution of creating a pause/resume button in the Testing tables (See Test 57 and 82 on pages 96 and 112 respectively), however, due to my poor judgment on the time scale, I did not have enough time to fully implement this idea, but I plan to in the future. From this mishap, I have learned to manage my time more efficiently and wisely.

High Scores

- **Does the score system and leader board load correctly? :** I believe that I have succeeded in fulfilling this requirement as the score system does work effectively on both levels as the player's score correctly updates when the user kills a zombie, presented on the top left corner of each of the Game Levels.

Additionally, the leader board table in the High Scores Form loads correctly, showing the top ten high scores of players in descending order. I feel I had added the high scores component in the application to give a sort of competitive element to the game, giving an incentive for the user to come back and play the game again. However, I plan to enhance the high scores element further, in future development, to show the user's two high scores from the two levels with varying degrees of difficulties instead of how it is done currently, only showing the user's highest score out of both the levels.

Beta Testing - Questionnaire



ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- ○ Under 18 years old
- ○ Between 18 and 30 years old.
- ○ Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

Your answer

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	○	○	○	○	○	Very Easy

Are the game mechanics and controls easy to follow and use?

- ○ Yes
- ○ No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

Your answer

Would you like to see any enhancements to the game and user experience to make the application better?

Your answer

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

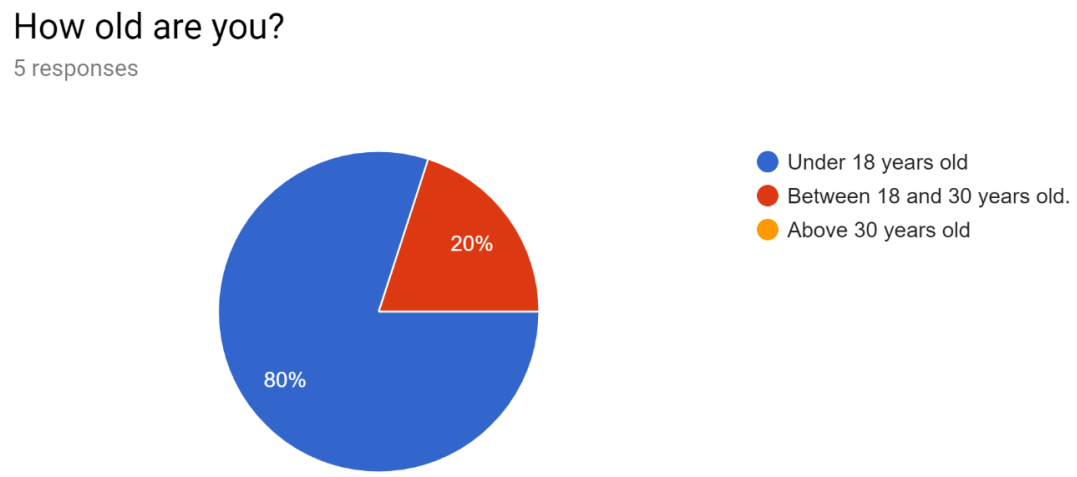
SUBMIT

Page 1 of 1

Analysis of Feedback from Beta Testing

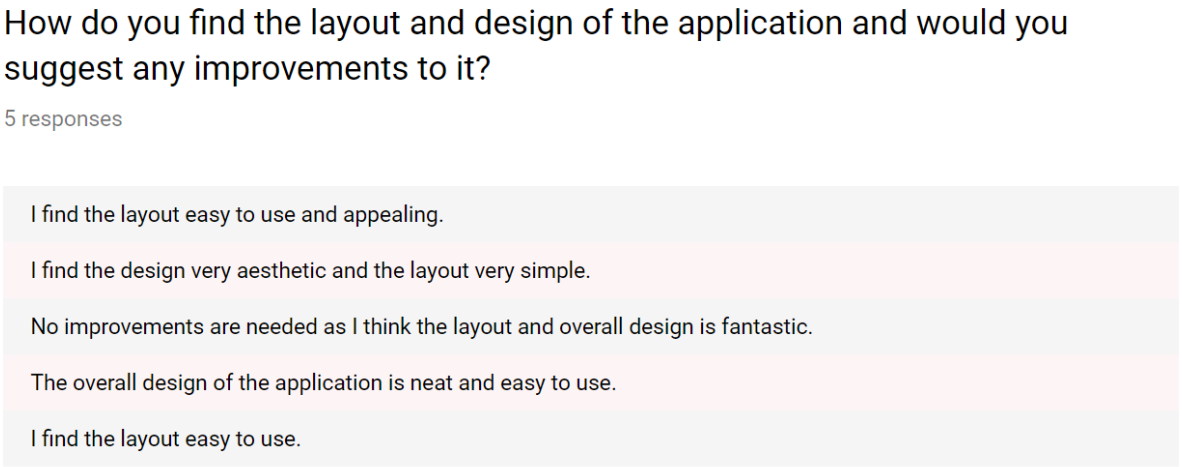
1. **How old are you?**

   This question was designed to show if I have met my target audience and how the opinions of people of different ages vary on my game. As evident in the appendix, the opinions between under 18-year-olds and the middle-aged had little or no discrepancy and so this goes to show, from my sample, that my initial target audience still holds. However, a bigger sample size should be taken as there were no persons above 30 years that had completed the questionnaire making the results statistically unreliable.

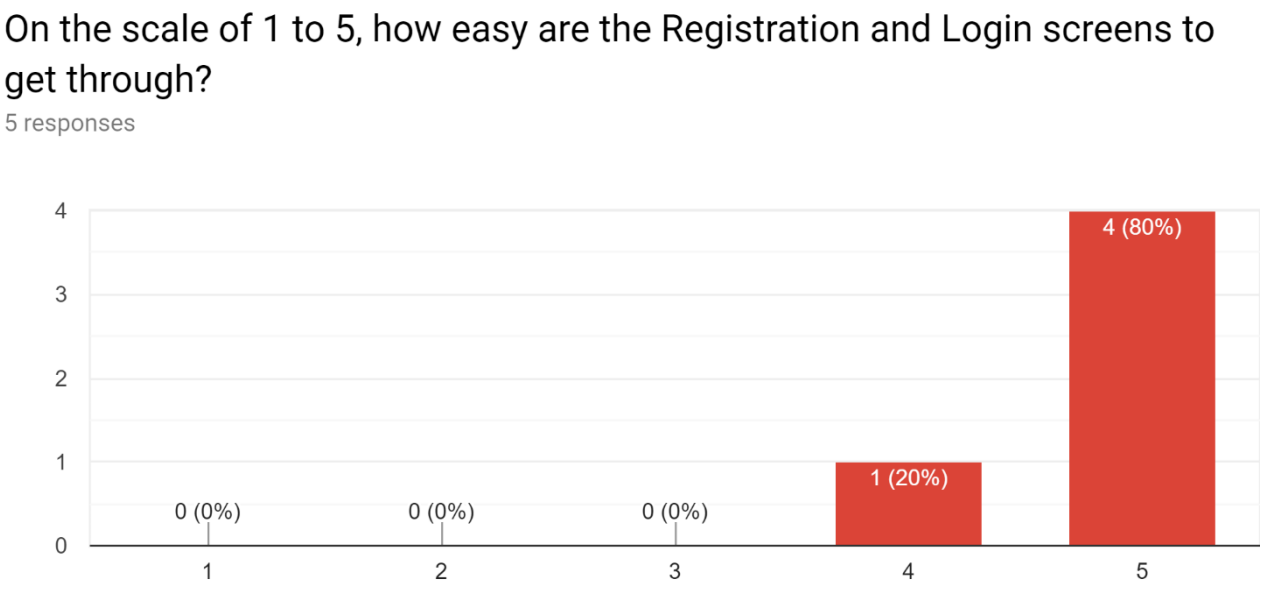


2. **How do you find the layout and design of the application and would you suggest any improvements to it?**

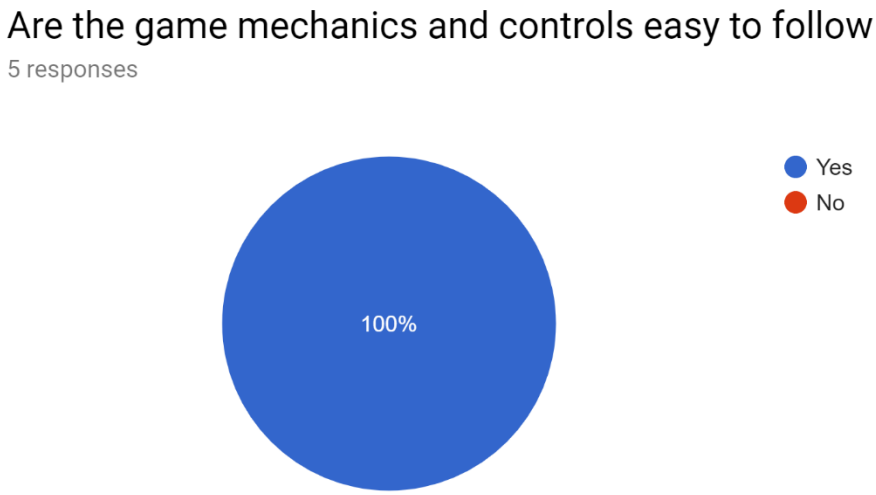
   This question's objective was to find if the application's layout and design was suitable for my target audience and as it turns out, all 5 respondents complimented the layout and design saying they found it "easy to use" and "neat". I feel this is very helpful as it shows that the layout and design require little focus when enhancing the program in future development and instead I can have more time concentrating on the other vital parts of the program. However, because of the small sample size with a restricted age range of under 30 years of age, further beta testing must be carried to give a more reliable answer.



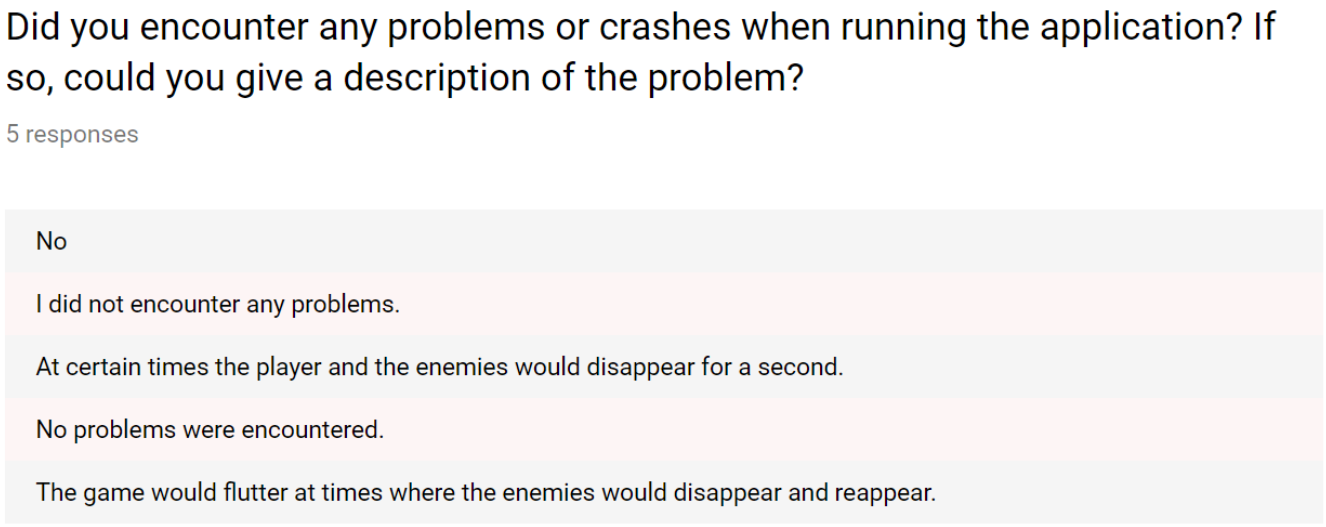
3. **On the scale of 1 to 5, how easy are the Registration and Login screens to get through?**
   From the responses to this question, I can deduce that there are little to no difficulties concerning the Registration and Login screens as the majority found them informative and straightforward to get through as evident by the graph below.



4. **Are the game mechanics and controls easy to follow and use?**
   All five respondents answered 'Yes' to this question clearly indicating that most users, who will play my game, will find the game mechanics and controls clear-cut and straightforward which is exactly what I intended when I was developing and designing the game. This again is helpful when enhancing the game in future, showing that very little has to be done about the game mechanics and controls of the game.



5. **Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?**
   Fortunately, the majority of the respondents did not come across any crashes or problems. However, two out of the five replies, highlight the same issue of the game lagging for a millisecond at certain times "where the enemies would disappear and reappear". Luckily, this is not a major issue and does not affect the actual gameplay and only serves as a minor annoyance for the user in the form of a graphical mishap. I intend to tackle this issue in future by pinpointing where the actual error is occurring using breakpoints in the code and hopefully resolve it.



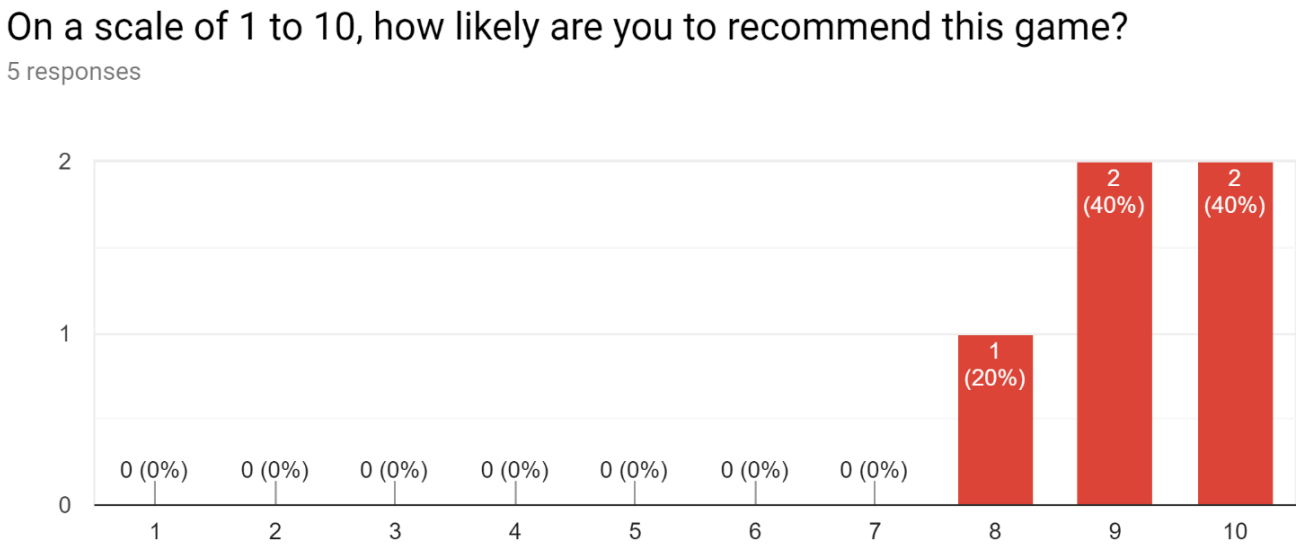
6. **Would you like to see any enhancements to the game and user experience to make the application better?**
   This question serves as the most useful in my opinion because as the game developer of 'Alone', I am continually looking for new ideas to add to the game to provide a better, more refined user experience. So, by providing the users of my application an opportunity to input their ideas to put into the game will serve as a better way of seeing what works and what does not. One of the respondents expressed that maybe a storyline should be added. This was actually my initial idea however, due to time constraints, I concluded that I will not be able to finish the project in time if I were to go down that path. Another respondent pointed out the fact that there was no help provided to members who have forgotten their username or password. This is an issue that I realised when developing the game but I decided to put the thought aside as it was unrealistic at the time and impractical due to the tight time limit. I quite liked the idea of a multiplayer system, as one respondent proposed, however, it is illogical to add this feature, when the main theme of the game is being the sole survivor, hence the name 'Alone', but I can reserve this concept for a possible follow-up to the game.

Would you like to see any enhancements to the game and user experience to make the application better?

5 responses

I would like to see a feature for 2 or more players to play together.
I would like to more variety in the enemies you encounter.
Maybe a story should be added to make the game more enjoyable.
I think the application is great, but a feature to help people who forgot their username and password would make it better.
I think a tutorial level should be included instead of going to the help guide to understand the controls of the game.

7. **On a scale of 1 to 10, how likely are you to recommend this game?**
   The intention with this question is to see if the participants actually like my game and from the response (shown in the graph below) I feel that the project was a success. The results show that my participants thoroughly enjoyed my program as their recommendation ratings range from 8 – 10 out of 10 indicating that I have a fair chance at winning the game-developing competition. However, I believe a higher sample size will increase the trustworthiness of the results so a range of differing opinions on the game can be seen.



Strengths and Weaknesses of the Package

Strengths

- Overall the package turned out to be a great success as the feedback from the people who participated in the beta testing and the people from the focus group predominately said that the game was “enjoyable” and fun to play. I think I did a great job in creating a balance on having a level that is easy and another where the difficulty and intensity of the gameplay is stepped up a lot, creating an environment where anyone will have fun.
- The fact that users can register an account to store their scores and then login a few days later without the loss of any information only enhances practicability and user experience as a whole.
- The feedback from the beta testing highlighted how the graphical material and layout of each screen was well designed and was highly commended. From my personal standpoint, without a visually attractive game, the user experience is greatly diminished, no matter how good the gameplay is. Especially for simple games like ’Alone’, the visual and auditory stimuli of the user are vital to the overall gaming experience and must be fully taken advantage of for the better.
- The code for the program is extensively commented so future improvements to the program can be easily done by anyone with coding experience as I have explained what every part of the program does thoroughly.
- The feedback the game received about the game mechanics and controls say that they are very easy to grasp due to the presence of the Help Guide explaining everything about the game. In my opinion, the fact ‘Alone’ is a survival game and therefore endless, only enhances the gaming experience as the game will only last how long the user wants it to. It is neither too short or too long.

Weaknesses

- The fact that there is no help available for users who have forgotten their usernames and/or passwords in the Login page is a huge problem and therefore has a great impact to the general user experience.
- The fact that pre-existing members have to log in every time the program is loaded is a bit of an annoyance and there should be a ‘Remember Me’ feature that the user can enable/disable so they can just open the game and start playing straight away.
- There is a graphical issue with the Game Levels where the images on the screen flicker (i.e. disappear and reappear for an instant). This might irritate some users weakening the gaming experience.
- The leader board only displays the member’s highest score out of the two levels instead an individual high score for the two vastly different levels which gives an unfair advantage to players only playing the effortless first level.

Areas of Further Development

Login

As mentioned before, a feature of helping forgetful members log in must be implemented. This could be resolved by making the member enter their e-mail address and then an e-mail being sent to them with their profile information if their entered e-mail address is present in the database (Binary File containing all member information). The member will be asked to come up with a new password from the e-mail. If the entered e-mail address is not present in the binary file, then the user will be asked to simply register an account. I consider this is a secure way of going about this problem without making it more complicated and long-winded for the user.

An additional idea to make the user's life simpler, was to incorporate a 'Remember Me' feature which will make it so that the pre-existent users do not have to keep logging every single time the application is launched, when the feature is enabled. Instead, the member can just start playing the game once the game is started. This aspect will obviously cause a lot of security problems and so will be hard to implement.

Characters

I think adding a character selection screen in the future, so the user can choose their own character will greatly enhance the game as this will make the it more engaging as the characters could be of different races, genders, ages and so on.

Story

The user experience will be significantly improved by adding a narrative with the use of text and images to act as cutscenes. The game will become more immersive and appealing and could vastly increase my chances of winning Monsoon Game's competition due to the complexity of my program.

Audio

The use of the background music and gameplay sound effects already present could have an added feature to disable them for people who do not want it giving more control to the user on the application increasing the user experience. Possibly more music and sound effects could be added.

New Levels

The prospect of exciting and bigger, new levels and map designs could be of a future enhancement to the program. One of the respondents from the beta testing had brought up the concept of a final "Boss level" which could give a final, satisfying conclusion to the game further boosting the gaming experience.

High Scores

I plan to add separate leader board tables for each of the levels, so no player will have an advantage over another due to the vast differences in difficulties between the levels. This is

important to prevent disputes between players and giving everyone a fair chance at getting better at the game.

Languages

As the game increases in magnitude, a language selector could be inserted so people all around the world could understand and play the game 'Alone'. This will ensure that no one is secluded from experiencing the game.

Self-Evaluation

Development of Game

I thoroughly enjoyed the programming aspect of the development of my game as I was constantly learning new skills and methods for my code, perfecting my C# skills and knowledge with every setback I confronted. It felt incredible to actually make my original concept and designs into a reality and truly has helped me discover my interests and a possible career opportunity even if I don't win the game making competition. However, every problem I stumbled upon during the development of the software drastically reduced my rate of work and I believe this would have been easily avoided with additional planning instead of pointlessly coding. In future projects like this or when I am enhancing 'Alone', I will be sure to design a very detailed blueprint describing every part of the application.

Additionally, I will carry out focus groups with the targeted audience in order to acquire information about features which would be appropriate and should be added. From these discussions, I can also understand what features not to include in order to save valuable time later on in the development.

The rate of progress was affected by the technological resources used as well. The computers were slow due to the low processing power resulting in the project taking more time than needed. For future projects, I will ensure that better resources and applications are equipped to maximise efficiency and reducing the time taken.

Design

I consider myself to be quite creative and fairly experienced in using Adobe Photoshop to edit and create images. Hence, I did my best in replicating my initial drawings of the two levels' backgrounds with minor alterations. The Storyboards acted as blueprints for the layout of all the controls on the forms which really aided to get an idea of the how all the screens will look like before they were implemented in Visual Studio.

Personal Skills

I can confidently say that I only got better at C# throughout the process, learning lots of new skills, for example the "Func<>" and "Action<>" delegates were entirely new areas I had not dwelled on before, but it was necessary to improve efficiency and reduce repetition of code. I learnt through my research online, that these delegates are used to pass methods or functions as parameters into

other methods. Func<> would be used if a method that returns data must be passed and called in the function while Action<> is used for methods that does not return anything (or return 'void') to be passed. Both of these delegates can hold a range of parameters that can be passed in. I used the Func<> delegate function to call the Zombie_RndSpawnPoint() method, returning an integer array, inside the GenerateZombies() method in the Zombie class. This was essential as the rules for the zombie spawn points were different for the two levels and had to be called through the Zombie class. The Action<> delegate was utilised to call the CollisionsWithSurrounding() method, with no return type (Void), from the TmrRefreshRate_B_Tick event method for the same reason as before in that the two levels had different background layouts.
Developing this game truly helped in strengthening my knowledge on the four fundamental pillars of the C# language or any other Object-Oriented Programming languages which consists of encapsulation, polymorphism, inheritance and abstraction. I may have not used all of the concepts, but I feel I have acquired a greater understanding of each. In doing so, I feel more confident in possibly learning other more flexible programming languages that utilise the concepts of Object-Oriented Programming, for example Python. In addition, during the developing stage, I learned the benefits and how to use a variety of program control structures such as If statements, Switch statements, For loops, While loops, Do-While loops and Arrays.

Time Management

I was able to complete all aspects of my game before the deadline. However, at certain times I felt I was under a lot of pressure which caused a lot of stress. This resulted in me rushing a lot of things as the deadline approached diminishing the quality of my work. As I look back now, I can see that I had not managed my time effectively and how easily I could have done a better job if I had just made a timetable marking the days when I will focus on a certain aspect of the game. This would have evenly spread out the work load and stress and I will be sure to follow this when undertaking future projects.

This process has really assisted me a lot in not just learning new programming skills but also plenty of organisational skills and time management skills that can be used in any type of project I must take on in the future. I have learned a plethora of skills using Adobe Photoshop through this project which could assist me in a variety of careers. This project has truly given me a taste of the world of work surrounding this field and I hope to do more like it.

Appendix

Responses cannot be edited

ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- (●) Under 18 years old
- ( ) Between 18 and 30 years old.
- ( ) Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

I find the layout easy to use and appealing.

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	○	○	○	○	●	Very Easy

Are the game mechanics and controls easy to follow and use?

- (●) Yes
- ( ) No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

No

Would you like to see any enhancements to the game and user experience to make the application better?

I would like to see a feature for 2 or more players to play together.

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	●	○

Responses cannot be edited

ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- (●) Under 18 years old
- ( ) Between 18 and 30 years old.
- ( ) Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

I find the design very aesthetic and the layout very simple.

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	○	○	○	●	○	Very Easy

Are the game mechanics and controls easy to follow and use?

- (●) Yes
- ( ) No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

I did not encounter any problems.

Would you like to see any enhancements to the game and user experience to make the application better?

I would like to more variety in the enemies you encounter.

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	●	○	○

Responses cannot be edited

ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- (●) Under 18 years old
- ( ) Between 18 and 30 years old.
- ( ) Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

No improvements are needed as I think the layout and overall design is fantastic.

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	( )	( )	( )	( )	(●)	Very Easy

Are the game mechanics and controls easy to follow and use?

- (●) Yes
- ( ) No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

At certain times the player and the enemies would disappear for a second.

Would you like to see any enhancements to the game and user experience to make the application better?

Maybe a story should be added to make the game more enjoyable.

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
( )	( )	( )	( )	( )	( )	( )	( )	(●)	( )

Responses cannot be edited

ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- ○ Under 18 years old
- ◉ Between 18 and 30 years old.
- ○ Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

The overall design of the application is neat and easy to use.

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	○	○	○	○	◉	Very Easy

Are the game mechanics and controls easy to follow and use?

- ◉ Yes
- ○ No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

No problems were encountered.

Would you like to see any enhancements to the game and user experience to make the application better?

I think the application is great, but a feature to help people who forgot their username and password would make it better.

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	◉

Responses cannot be edited

ALONE - Peer Evaluation Questionnaire

Users are kindly asked to complete the following questionnaire to provide us with valuable feedback which will greatly help us to appeal the game to a wider target audience and further develop the game.

Name:

How old are you?

- (●) Under 18 years old
- ( ) Between 18 and 30 years old.
- ( ) Above 30 years old

How do you find the layout and design of the application and would you suggest any improvements to it?

I find the layout easy to use.

On the scale of 1 to 5, how easy are the Registration and Login screens to get through?

	1	2	3	4	5	
Very Difficult	○	○	○	○	●	Very Easy

Are the game mechanics and controls easy to follow and use?

- (●) Yes
- ( ) No

Did you encounter any problems or crashes when running the application? If so, could you give a description of the problem?

The game would flutter at times where the enemies would disappear and reappear.

Would you like to see any enhancements to the game and user experience to make the application better?

I think a tutorial level should be included instead of going to the help guide to understand the controls of the game.

On a scale of 1 to 10, how likely are you to recommend this game?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	●