**A Level computer Science**

Component 3

Space Game

Logo

Description automatically generated

By: Louis Pattern

For: A. Issa

**Stoke on Trent Sixth Form College**

Table of Contents

[1.1 Introduction 4](#_Toc112930515)

[1.2 Problem Identification 5](#_Toc112930516)

[1.3 Why the problem is suited to a computational solution 6](#_Toc112930517)

[1.4 Stakeholders analysis 7](#_Toc112930518)

[1.5 Research 8](#_Toc112930519)

[1.6 Features of the proposed solution 12](#_Toc112930520)

[1.7 Limitations of the solution 14](#_Toc112930521)

[1.8 Stakeholder Consultation 15](#_Toc112930522)

[1.9 Hardware and software requirements 16](#_Toc112930523)

[1.10 The requirements of the solution 16](#_Toc112930524)

[1.11 Success Criteria 17](#_Toc112930525)

Chapter One: Analysis of the problem

## 1.1 Introduction

In this project, I aim of entertaining children and teenagers by providing them with a 2D videogame for recreational use. The main goal of the game should be to help teenagers relax in their free time when they are not doing homework or revision. My game will be suitable for college students and even children. Its theme is science fiction and space. The game will have both options for a single player fighting an AI and multiplayer for one player versus another locally, allowing users to play the game how they prefer. The game will involve each player controlling a spaceship on each half of the screen and shooting enemies approaching them. There will be a number of levels in single player, increasing in difficulty as the player progresses. The game will be very customisable, allowing the user to change settings. My stakeholders for this project will include a college student at Stoke Sixthform College, a child and a young adult.

## 1.2 Problem Identification

Many teenagers become stressed or worried about exams and schoolwork. Videogames could help relieve this stress in their free time by providing a means of escapism from their school life. Stress is also a major problem for young adults, so my project would not only be targeting teenagers. Videogames have also been proven to have other benefits. For example, improving focus and reaction time.

Most retro games are fully single player experiences, with no way to interact directly with another human player in the game aside from competing for a spot on a scoreboard after game completion. My project aims to go against this convention by allowing players to directly compete against each over in real time with their scores being tracked and displayed clearly on screen.  
In addition, most 2D videogames that feature a 1vs1 format are usually fighting games, a genre that most people are turned away from because it can be too competitive, forcing players to learn specific ‘combo’ moves reducing the ability for people to play casually and for fun. This could either put people of retro videogames entirely due to frustration or make younger users shy away from multiplayer gameplay entirely. There is also a limited number of retro shooter games with such a format and even fewer with a sci-fi theme.

Furthermore, a lot of 2D shooters feature either just a single large level, a limited number of levels or there is little variation between levels. If there is little change in enemies or combat between levels then the user will become bored due to lack of challenge or stimulation. However, if there is no visual variation between levels then the user can also become bored.   
My game would feature a unique background for each level. This would not only help to keep a user’s interest high (particularly for children) but also make the give the player a sense that they are travelling and progressing through the game.

## 1.3 Why the problem is suited to a computational solution

This problem is amenable by a computational approach because it is a videogame, thus has to involve the user interacting with a computer. My project being a videogame rather than a non-computational solution such as a board game has numerous advantages. Firstly, the computer can process the user’s inputs and perform tasks much faster than a human can interact with a non-computational game. In addition, animations and movement for a videogame can be easily displayed by a computer screen whereas in a normal game they cannot. Colours in videogames can also be adjusted, allowing the option to enable a colour-blind mode. Finally, most people with disabilities are still able to play because usual input methods such as mouse and keyboard do not require much movement.

Videogames also allow the user to have more customisable settings, for example: window size and colours used.

## 1.4 Stakeholders analysis

My stakeholders for this project will be, college students at a variety of colleges, a high school student and two adults. I will give each of my stakeholders a rundown on the proposed features of the game and interview them for feedback and criticism. If any of the stakeholders suggest a way to improve the game or a new feature to add to the game, I will try to implement it.

The high school and college students I have selected are casual gamers, playing a variety of videogame genres. Some prefer to play mobile games on their phone whereas others prefer to play on their home consoles. The few that owned their own PC build at home would consider themselves PC gamers.

One of the adults is a casual gamer and has their own gaming PC which they occasionally use to play games on. The other adult has little interest in videogames and does not play them at all in their free time. They also have limited experience with computers.

## 1.5 Research

One existing videogame of a similar format is Defender, an arcade game from 1981. This game is a side scrolling shooter where the player has the objective of shooting aliens on another planet. The player can move in all directions, with up and down moving the ship directly and left and right moving the terrain. However, because of technical limitations of the time, the background is extremely simplistic, with the ground consisting of a single zig-zagging line.



Defender Gameplay

The game is also single player only, something which I aim to improve upon in my game. Different enemy types feature in Defender, with each alien behaving differently and awarding a different number of points when the player destroys them. These enemies are also vastly different visually, allowing the player to easily differentiate between them. This visual difference also allows new players to easily learn the behaviour and mechanics of each enemy. Another gameplay element of Defender is that the player can rescue captured humans by shooting pods.   
While Defender was an arcade game rather than a computer game that be run on windows without the use of an emulator, I still think it has some interesting features that I could implement in my own game.

**Parts that I will apply to my solution:**

**The player’s inputs being used to control a spaceship, both movement and shooting. The scoring system is another good feature of this game, with enemies that are harder to kill rewarding the player with more points. This incentivises the player to attack new, harder variants of aliens and save more pods rather than just killing the same basic enemies repeatedly.**



Defender scoring system

**Disadvantages:**

Defender is a single-player only game, meaning it doesn’t have a 2-player mode like my game will feature. The game also obviosly cannot run on modern PCs without the user having to install additional software such as an emulator.

Another videogame of similar format is the Touhou series, specifically the windows games under the ‘bullet hell’ genre. This is a genre of 2-D shooters with a focus on a large number of projectiles or bullets that the player must avoid while shooting enemies and progressing through a number of levels. These games place emphasis on movement and the player’s dodging capabilities, with it being better to simply avoid getting hit at all whilst picking up score bonuses in a level rather than killing as many enemies as possible. The player has a limited number of lives but getting a high score can lead to bonus lives being awarded. The Touhou games also feature different playable characters, each with their own attack style (for example one character might launch projectiles in a wide arc whereas another might have a more focussed line of bullets). Characters also unlock special attacks such as homing bullets as the player progresses through the game and some characters have higher or lower movement speed.   
The difficulty level of the game can also be changed from a settings menu, making the game overall easier or harder to complete.

The UI for these games often displays the player’s score and other information in a column on the right-hand side of the screen and the game itself is displayed as a column on the left. This is appropriate considering the levels scroll vertically rather than horizontally. However, this approach would not be suitable for my game as it based on moving horizontally.



Touhou – “Double Dealing Character” UI and gameplay

Parts that I will apply to my solution:

The ability for the player to change the difficult from a menu is something I will apply to my game. Changing the difficulty effects a number of key elements such as: enemy projectile speed, the number of lives the player gets at the start of the first level, the amount of health a ‘boss’ enemy gets, the score the player receives (the harder the difficulty the higher the score). Setting the difficulty to its highest level also causes the number of enemies and enemy projectiles to increase a lot.

Disadvantages:

Because of the games focus on a large quantity of bullets, there can be a high number of projectiles on the screen at once, the game can suffer from performance issues on less powerful machines. This means (depending on hardware) it is not uncommon that the frame rate drops to below 40 frames per second. This can clearly be frustrating to the user.  
The games are also in Japanese with   
Another disadvantage of the Touhou series is the colour scheme. There are often many bright and intense colours in the UI, characters and projectiles. The colours can often be painful to look at or strain the eyes. Furthermore, can often make it harder to focus during gameplay, especially since the background almost never a single colour. This also makes the game less accessible to people with visual disabilities such as colour-blindness.

## 1.6 Features of the proposed solution

* Single player mode where the player fights against enemies controlled by AI.
* Multiplayer ‘versus mode’ where one player fights another. This will give my project a unique feature compared to other scrolling shooter games. This would also allow for friendly competition that isn’t possible in only single player games. There is a range of projectiles depending on whether they are being fired by an enemy or a player.
* Log in screen where the user enters their username and password, a database it checked to make sure they are correct. If they are, the user will be logged in and taken to the game.
* High score leader board, featuring the player’s name followed by score. This information will be stored in a database and will be displayed when a player completes the game or they manually select ‘high scores’ from the main menu. Scores should be displayed in descending order and each difficulty level will have a separate leader board. The date when the score was obtained would also be stored in the database. Only the top 5 scores would be displayed, and these scores would be arranged in descending order with each player’s name clearly next to their corresponding score.
* Difficulty levels: easy, medium and hard. For higher difficulties, enemies will have faster fire rate, being harder to dodge and more enemies will be present in each level. The player will also have more starting lives if they play on an easier difficulty.
* The limited number of lives the player has which will be correctly displayed at the top (or bottom) of the UI. The number of lives will depend on the difficulty.
* Some enemies also have multiple lives like the player, but they don’t receive invincibility frames like the player does.
* Invincibility frames: after a player takes damage and loses a life, they will be invulnerable and unable to lose more lives for a short period of time (≈ 0.5s). This feature will be accompanied by a flashing animation on the player’s ship. This is implemented because it prevents the player from losing a life for every frame they are in collision with a dangerous object. In other words, it prevents the user from losing multiple lives in very quick succession.
* The game has a ‘colourblind mode’ which can be selected from the settings menu. This will make the colours of the game more accessible and readable for people with colour-blindness.

## 1.7 Limitations of the solution

* One limitation of the solution is that the game can only use keyboard inputs alone and there is no support for mouse, controllers, or joystick input. This is acceptable because the most likely audience for the game – college students during their free periods or casual PC gamers would be less likely to prefer controller.
* Another limitation is that the max frames per second of the game is capped at 60 fps. This is because movement in the pygame loop is based frame by frame rather than real world time so increasing the frame rate beyond 60 increases the speed at which the player can move etc. This limitation is not a major issue since 60 fps has been the standard for gamers for a long time, especially for 2D games, where stability and a constant frame rate is often preferred over a higher, fluctuating frame rate. 60 frames per second appears smooth enough to the human eye. In addition, almost all modern hardware would be able to run my game at a stable 60 fps.
* The graphics for the game are all simple 2-dimensional images rather than 3-dimensional. This has the advantages of making the hardware requirements to run the game lower, making it more accessible. This is because not everyone can afford a good graphics card which is often required to run most modern 3D games. This therefore could lead to more people being able to experience and have fun playing the game.

## 1.8 Stakeholder Consultation

I have interviewed a range of my stakeholders, asking them for feedback on the game towards the end of the interview. Their responses will be summarised and placed in quotes.

**Interview with a College Student**

* **Have you ever played a videogame?**“Yes”
* **Do you play games a lot in your free time?**“Yes, when I’m not focused on homework or revision.”
* **What kind of games do you like to play?**“I mainly play first person shooters and platformers but there aren’t really any genres I don’t like”
* **Do you think playing games can be beneficial?**“Of course, they can help me relax and take a break from schoolwork. Multiplayer games also allow me to socialise and have fun with my friends.”
* **(After having showed interviewee proposed features) What features of my proposed game did you like?**“I liked the ability to directly play against another player. I also liked the idea of a leader board with the best players at the top.”
* **What features did you think could be improved?**“I think the game should have the ability to go full screen. I mostly don’t play games windowed. A pause feature would also be good.”
* **What colours do you think should be used?**“Black for the background and overall, nothing too bright.”

**Interview with an Adult**

* **Have you ever played a videogame?**“I played some arcade games when I was younger”
* **Do you play games a lot in your free time?**“No.”
* **Do you think playing games can be beneficial?**“Maybe as long as they are played in moderation”
* **(After having showed interviewee proposed features) What features of my proposed game did you like?**“I liked the retro arcade style.”
* **What features did you think could be improved?**“Add music to the game”
* **What colours do you think should be used?**“Blue for the player and red for enemies”

## 1.9 Hardware and software requirements

Hardware requirements:

* 1.5GHz or faster processor – in order to run the game smoothly with no stuttering
* Minimum 512MB RAM
* 100MB free hard drive space – sufficient to store the game
* Working keyboard and mouse – the game input is all keyboard based and the login menu needs mouse for input
* Working monitor – to display the game

Software requirements:

* Operating System: 64-bit Windows 7 or later or OS X 10.11 or later

## 1.10 The requirements of the solution

* Firstly, the user would need to log in with the log-in screen displayed when they launch the program. This would be done by entering their username and password into the boxes and hitting enter. This is done so their username can be saved to the high scores database (along with the score they get)
* The user can navigate the main menu with WASD or arrow keys and select an option by hitting the enter key or the spacebar.
* When in game, the player can move their ship normally with WASD, enable slower movement or ‘focus’ by holding shift while using WASD to move. Spacebar is used to shoot and can be hold down to shoot repeatedly. There is a maximum fire rate.

**Input requirements**

* The player is able to navigate the main menu and select the desired option: play, settings, versus, high scores or log off.
* When backspace is pressed, it takes them to the previous menu
* The player is able to move the ship using WASD on the keyboard. The ship cannot move outside the boundaries of the screen
* The player can shoot by pressing the spacebar. The key can be held down to shoot continuously.
* A second player is able to move a second ship with the arrow keys in versus mode

**Process requirements**

* The system must record the player’s score whenever they successfully hit an object or collect a score pickup.
* The system must detect whenever the player hits a dangerous object such as an alien and make the player take damage
* The player’s lives must be tracked. It will be decremented by one when the player is hit and incremented when the player picks up a health pickup.
* If the player’s lives reach 0 it takes them to the game over screen.
* The game has a timer that must be decremented every frame of gameplay. The timer is used for many important things such as when enemies appear.

**Output requirements**

* The main menu buttons must be displayed only on the main menu and have an animation for when the button that the player is currently selecting.
* The player’s lives must be displayed during gameplay. The number of hearts corresponds to the number of lives the player has left.
* The game has a timer that must be decremented every frame of gameplay. The timer is used for many important things such as when enemies appear and when the player wins.
* When the player selects the ‘highscores’ button from the main menu, the top five scores will be displayed in order on screen.
* In two-player mode, whichever player wins should display a different win screen.

**Storage requirements**

* The player’s settings are saved to a text file so that if they exit and return the settings they selected will not change.
* Usernames and passwords of users are saved to a table in a .db file
* Usernames and passwords of admins are saved to a separate table
* High scores are saved to another table, storing the player’s username, the score they got and the date the score was achieved.

## 1.11 Success Criteria

|  |  |  |
| --- | --- | --- |
| **Requirement** | **What this success means** | **Evidence** |
| Login screen before the game is played | A login window where the user can enter their username and password. If they are correct, it will login the user to the game. | Video of the login window with a correct username and password being entered. |
| Intuitive UI | The user can utilise the window without needing prior knowledge or guidance. Buttons should be clearly labelled. | Questioning and obtaining feedback from stakeholders after they have used the UI |
| Simple design | Buttons and text should not be too small, the colour scheme should also be appropriate. | Screenshots of the login window and game menus |
| Main menu for the game | A number of options that can be selected with the keyboard alone.  It should be clear which option the user is selecting. | Screenshot of the main menu of the game |
| Settings screen for the game | A settings menu where the user can change the window resolution, difficulty, audio volume and colourblind mode.  The currently active settings should be easily visible. | Screenshot of the setting menu of the game |
| Window size is changeable | When the window width and height is changed from the settings menu, the game will restart. Sprites’ size and position should scale based on window width and height. | Video of window size being changed |
| Number of lives displayed during gameplay | The HUD (heads up display) should contain a number of hearts that indicate the number of lives the player has left. In 2-player mode, each player’s number of lives must be displayed separately. | Screenshots of single player mode and 2-player mode showing different numbers of lives. |
| High-scores leader board | The top 5 scores achieved must be shown in the high-scores screen next to the player who got each score. The text must be clear and readable. | Screenshot of high-scores screen |
| Controllable ships by the players | In 2-player mode, one ship should be controlled with the WASD keys and the other controlled with the arrow keys. | Video of player movement |
| The game should have minimal bugs or glitches. | The game should function completely as intended. Any unintentional effects that completely change the game should be patched. | Evidence of bugs being patched in the logs |
| The program must not crash but instead show an error message | Tkinter messagebox is used to display a warning message at the centre of the screen with an appropriate message for the error. | Screenshot of error message |
| Game should run smoothly at a constant 60fps | The game caps frame-rate at 60 frames per second. The game is simple and 2D so it should not have performance issues even on lower end hardware. | Video of game running with external software measuring fps |