



Team 18 Nexus Evaluation Report

Table of Contents

INTRODUCTION.....	5
2. EVALUATION.....	5
2.1 SOFTWARE QUALITY	5
2.2 ACHIEVED SCOPE	6
2.3 MAINTAINABILITY	7
2.4 ACCESSIBILITY	8
2.4 USER EVALUATION	9
2.6 FUNCTIONALITY	10
2.7 USABILITY.....	11
2.6 EFFICIENCY	13
2.9 SECURITY	13
3.0 TEAMWORK	14
3.1 PROJECT MANAGEMENT	15
4.0 POSSIBLE IMPROVEMENTS	16
4.1 SOFTWARE QUALITY.....	16
4.2 TEAMWORK	17
4.3 PROJECT MANAGEMENT	19
5. CONCLUSION	21
6. REFERENCES.....	22
7. APPENDICES.....	23
7.1 TRELLO BOARD LINKS	23
7.2 TRELLO SCREENSHOTS.....	24
7.3 DISCORD	25
7.4 TRELLO MEETINGS.....	27

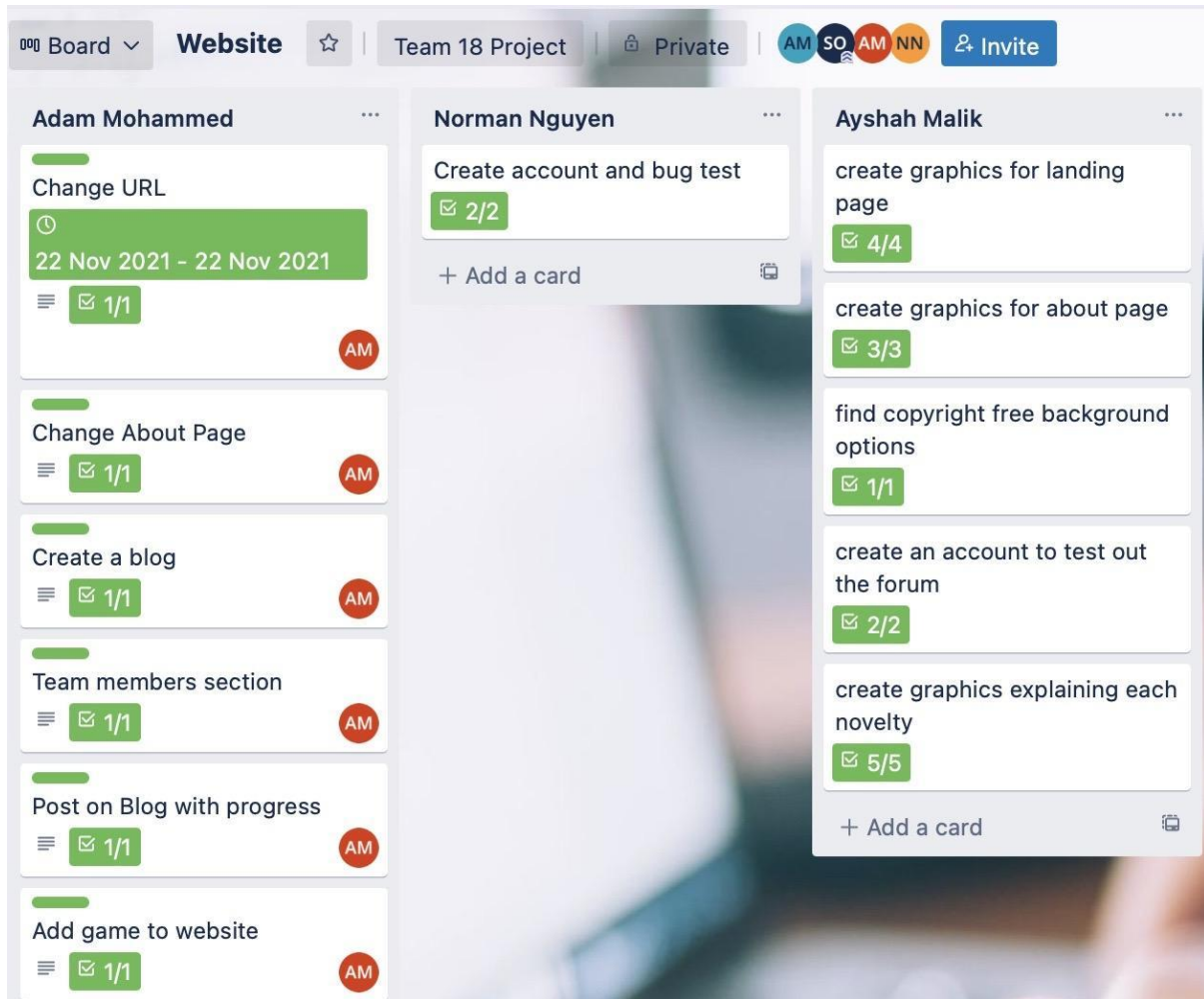
7.5 QUALITY STRATEGY DOC BOARD:

The screenshot shows a Trello board titled "Software Quality Doc" with a background of orange and white stripes. The board is organized into four columns, each representing a team member's tasks. Each task card includes a title, a progress indicator (a green box with a checkmark and a fraction), and a circular avatar of the assignee. The tasks are as follows:

- Adam Ejaz (AE):**
 - Create the template for document (1/1)
 - Start answering questions from the introduction (1/1)
 - Answered all questions from the introduction (1/1)
 - Start answering questions from the internal and external quality sections (1/1)
 - Answered majority of questions from the internal and external quality sections (1/1)
 - Answered minority of questions for security (1/1)
- Ayshah Malik (AM):**
 - Format document, include table of content and group's contribution (1/1)
- Naomi Eniola Adesiyun (NA):**
 - Improve the table of contents in software quality strategy (1/1)
- Jashan Gandham (JG):**
 - Improve layout of Software Quality Document (0/1)
 - Answer questions on software quality document (1/1)

At the bottom of the board, there is a navigation bar with the following elements: a "Board" dropdown, the title "Software Quality Doc", a star icon, "Team 18 Project", "Workspace visible", a list of avatars (AE, SO, AM, JG, +2), an "Invite" button, and a "Power" button.

7.6 WEBSITE BOARD:



.....	29
7.7 GRAPHICS BOARD:.....	29
7.8 GIT HUB COMMITS AND CONTRIBUTIONS	31
7.9 SOURCE CODE	

```

public void CreateLines() //This instantiates all the lines. This is done to later give the user the option as to how big they want the board to be.
{
    int count = 0; //a count of how many lines were made
    for (float x = (704); x <= (512 + 704); x += (512 / lines)) //This covers width
    {
        count++;
        if (count == 1 || count == lines + 1) { } //This if statement is so a the first and last line isnt made on the board as they are already outlines
        else
        {
            GameObject line = Instantiate(VertLine, new Vector3((x * Screen.width / 1920) + (2 * Screen.width / 1920), 540 * Screen.height / 1080, 0),
                Quaternion.identity, GameObject.FindGameObjectWithTag("ButtonHolder").transform);
            //Adjusted for different screen resolutions
        }
    }
    count = 0;
    for (float y = (540 - 256); y <= 256 + 540; y += 512 / lines) //This covers height
    {
        count++;
        if (count == 1 || count == lines + 1) { }
        else
        {
            GameObject line = Instantiate(Horiline, new Vector3((704 * Screen.width / 1920) + (256 * Screen.width / 1920), (y * Screen.width / 1920) + (2 * Screen.width / 1920), 0),
                Quaternion.identity, GameObject.FindGameObjectWithTag("ButtonHolder").transform);
        }
    }
}

```

Image 1:
CreateGrid.cs

```
public void OpenHelpMenu() //These show functions are attached to buttons on the UI
{
    Canvas mainCanvas = GameObject.Find("Canvas").GetComponent<Canvas>(); //This finds the Canvas in the Scene and assigns it to a variable
    GameObject mainUI = mainCanvas.transform.Find("MainUIHolder").gameObject; //These 2 lines get the Main UI and the Help Menu as Game Objects
    GameObject helpMenu = mainCanvas.transform.Find("HelpMenuHolder").gameObject;
    mainUI.SetActive(false); //These two enable/disable them accordingly;
    helpMenu.SetActive(true);
}

public void CloseHelpMenu()
{
    Canvas mainCanvas = GameObject.Find("Canvas").GetComponent<Canvas>();
    GameObject mainUI = mainCanvas.transform.Find("MainUIHolder").gameObject;
    GameObject helpMenu = mainCanvas.transform.Find("HelpMenuHolder").gameObject;
    mainUI.SetActive(true);
    helpMenu.SetActive(false);
}
```

Image 2:
MainMenuButtonHandler.cs

```
public void signup() //This is executed when a user clicks the sign in screen
{
    var isPassValidated = hasNumber.IsMatch(signupPass.text) && hasUpperChar.IsMatch(signupPass.text) && hasMinimumChars.IsMatch(signupPass.text); //This checks the password matches the regex above
    var isEmailValidated = isEmail.IsMatch(signupEmail.text); //Same as above for email
    if (!isPassValidated)
    {
        Debug.Log("Pass not validated");
        return;
    }
    if (!isEmailValidated)
    {
        Debug.Log("Email not validated");
        return;
    }
    //If both are validated it means that they can signup with their account
    JObject payload = new JObject();
    payload["method"] = "signup";
    payload["email"] = signupEmail.text;
    payload["username"] = signupUsername.text;
    payload["password"] = signupPass.text;
    ws.Send(Encoding.UTF8.GetBytes(payload.ToString()));
}
```

Image 3:
Login.cs

.....35

8.0 MEETING MINUTES.....37

8.1 GITHUB LINK.....39

Website URL: <https://nexusgame.wixsite.com/nexus>

Who Contributed To Document & How:

Adam Mohammed	<ul style="list-style-type: none">• Drafted in detail 1, 2.2, 4.2• Redrafted 2.1, 2.3, 2.8, 3.1• Created the initial plan for the document (subheadings and points to include within each heading)• Recorded all meetings (meeting minutes)• Helped with final redraft
Saeed Othman	<ul style="list-style-type: none">• Drafted 2.1, 2.3
Werner Gutierrez Biela	<ul style="list-style-type: none">• Drafted 2.4, 2.6
Jashan Gandham	<ul style="list-style-type: none">• Drafted 2.5, 3
Naomi Adesiyan	<ul style="list-style-type: none">• Drafted 2.8, 3.1

Norman Nguyen	<ul style="list-style-type: none"> • Drafted 2.9 3.2 and helped with final redraft
Adam Ejaz	<ul style="list-style-type: none"> • Drafted 2.7, 4.1
Ayshah Malik	<ul style="list-style-type: none"> • Drafted in detail 4.3, 5, 6, 7 • Redrafted 1, 2.2, 4.2, 2.4, 2.5, 2.6, 2.9, 3, 3.2 • Recorded all references in Harvard Referencing format • Managed Trello and gathered its evidence • Gathered all evidence found in the appendices (excluding meeting minutes) • Did a final, complete redraft of the whole document including formatting

Introduction

In this report, we analyse, evaluate, and critically reflect on our team's application of the software engineering lifecycle, as well as the team's culture and behaviour, to formulate our own strategies for continuous ongoing learning in the context of future team-based experiences. We also develop, justify, implement, and evaluate a strategy to ensure software quality, including quality assurance processes, an appropriate test plan, and the use of appropriate tools for the support of effective team-based software engineering activity (e.g., version control, task and bug trackers, automated testing, and deployment). We evaluate team working, which includes aspects relating to team culture, leadership, team roles, peer-learning, and team development as well as project management and process, which includes the software engineering process, quality assurance, project management, and use of task and bug tracking tools, and version control. Furthermore, we evaluate the extent of the software we produced, the way our team worked, the way the project and software development were managed and were high quality. We end with a discussion of how we could have improved the quality of our software, the way our team worked, the way the project and software development were managed.

2. Evaluation

2.1 Software quality

The game runs perfectly on both the client and server side. Due to it being a 2D game, it can run without a hitch on any platform. The server is programmed using Node.js which is known for being lightweight. The source code is well commented, and all the code is separated into different classes/files (See Appendices 7.X) This is for both the client and server codebase. This makes debugging easy for developers as when an error occurs you can go straight to

the file the error occurred on instead of searching through one file. The game does meet the intended requirements. It is possible to play online with another user, it is turn based and it is winnable

The game has two modes, a timed one where players have 60 seconds to win their game and if the time limit runs out then the game results in a draw . And a standard game mode without the timer. Both game modes include power and doom cards which are introduced to make games last longer and introduce different scenarios. This should be able to keep players interested in long enough as there are many possibilities to outplay your opponent.

Due to the nature of Web Sockets, it is quite easy to make a connection with a WebSocket server and send it requests. To test the server, a client was used to send the server, WebSocket requests that the game would normally send.

All code is currently on GitHub (See Appendices 7.X). To contribute changes, all a developer would need to do is fork the current source code from GitHub, make changes and do a merge request. Due to the way the source code has been organised; it is very easy to read. It has been formatted to look neat, all repetitive code has been separated into functions and classes (See Appendices 7.X). Also, relevant code has been commented to inform other programmers what a function/class does (See Appendices 7.X). The software can be evolved very easily. The server is built on a modular design meaning that it is very easy to import modules of code.

The target audience our game is for young to mid teenagers. On the website we have a forum, a chat box, a dedicated discord server and a developmental blog. The developmental blog is to keep our community up to date with the current affairs of the game. The forum we have is for users to chat amongst themselves about the game. The chat box is for users who are currently playing the game to chat between themselves. We also have a discord server linked on the website where users can chat with the developers and ask questions. To access the game there is a download link on the website. The server of the game is currently hosted on digital ocean meaning that the users will only have to download the game and not configure anything else (See Appendices 7.X).

2.2 Achieved scope

By the end of the project, the extent to which the scope was achieved and what we managed to deliver on time was high quality software which is reasonably bug or defects free. It meets all the requirements stated in project assessment guidelines and specification as well as our very own software quality and strategy and game design document. In the user's view, the quality of software is fit for purpose and meets the user's needs as they're able to play the game. The design of the software is aimed at the target audience of teenagers and is that of a dark blue pirate theme design which looks the part in terms of the graphics and feels immersive. The durability of the software is also good as it lasts so long as the server and bugs are maintained. The consistency of the software is rated good as the software and game features were tested to ensure the same success rates and outcomes. The game and software are reliable, but the server requires maintenance as there are occasional crashes, however, it is of an acceptable level. The functionality of the game is good as all the functions required to play the game and navigate work as intended and have been tested.

There is a good community aspect of the game as on the website there is a community tab with a forum where you can ask questions and create posts, a chat box where other players who are playing the game can chat with each other and a dedicated discord server where developers, players can play, hang out and communicate. Besides fostering a community between Nexus users', these features also allow them to voice their concerns, give feedback and suggest bug fixes. Overall, the main goal of this project was to have our game, Nexus, be a fully functional, online multiplayer turn-based game and for it to be playable by only those who've registered an account on the website, which was successful. Another goal of ours was to have cards and dead tokens given to the player randomly and create obstacles to create a competitive atmosphere which has also been implemented. The features which were mentioned in the software quality and game design document that were not implemented were a score-based system and a leader board. These were not implemented due to having a change in creative differences and not enough time, so instead of adding these features in, others replaced it, such as a login implementation within the game itself and a custom server password creation option.

2.3 Maintainability

The code, in general, is written in such a way that, should a problem arise, the current programmer in charge of the project, would be able to find the issue(s) easily. Repetitive tasks within the program are made as functions to not overwhelm the main function with incomprehensible coding. At all times, we tried to make the function's name as meaningful as possible (See Appendices 7.X) so that anyone who reads it, has an idea of what that function does. The same is true with variable/constant names, both global and local (See Appendices 7.X) In addition, comments are in place by each function to shortly describe what the function does (See Appendices 7.X) In parts of the code that are hard to understand at first glance, additional comments are given to clarify the significance of that piece of code and the purpose it was given by the author. These good coding practices, and others, were employed to make it easy to fix and/or make changes to the code should the need to do so happen, without having to rely on the original programmer who designed it.

Throughout the development of the code, each function/piece of code was tested to see if it produced the expected outcome. This was done to enforce the reliability of the code and ensure it delivered what was expected. If the code did not work as anticipated, time was taken out to evaluate the results and resolve the issue(s). Issues with the speed, sound, memory and even graphics were corrected and improved. For example, a bug in the performance of one of the power cards ('Enabled') was found where it was permanently active when activated instead of being one-time-use as it was intended. However, just because parts of the code work well individually, it does not guarantee that it'll work well as a whole. So additional testing was done to check if all parts of the code are working as intended when used in cohesion. Members of our team, who were not directly involved with the coding aspect of the project, were asked to try out each iteration of the project, as a form of black-box testing, to see if any bugs arise during use, in the perspective of the player. When bugs were found, the project was sent back to correct those coding errors and a new version of the project was formed to be black-box-tested once more.

As is sometimes the case, it is hard to predict if the patch you introduced to fix a previous bug could possibly introduce a new one. The very nature of coding will almost always have

bugs. Add to that the pressure of deadlines, which exhausted developers are rushing to meet, and the complexity of the project, the likelihood of introducing another bug is so much higher. Nonetheless, we aimed to patch as many bugs as we could as several tests were conducted during the development of the game, to eliminate all found bugs and develop a usable and playable copy. Since coding at this level is highly complicated and prone to bugs no matter how careful the developers are, a software bug forum is available to acquire the user/player base assistance in locating them for the creation of future patches.

The game was developed using the Unity Game Engine, and so, the game was designed to be used on a windows platform.

The code is clearly marked with adequate comments, in the respective files, and so making corrections was relatively easy. Since new developers were introduced (under the lead programmer's guidance) to assist with the coding of other aspects and features of the game, having these comments in place made it easier to locate where the problems are within the code when making changes. The maintainability of the project's code is highly viable to anyone with intermediate knowledge, or higher, in C# programming and the use of the Unity game engine, as well as a basic understanding of server-client distributed systems. Should any of the developers become indisposed or new developers are introduced, between the comments laid out across the code and the documentation that comes along with it, anyone with the requirements stipulated above, should be able to take over and maintain the code for the program.

2.4 Accessibility

To accommodate for a larger audience, when creating Nexus, we considered different factors which would enhance accessibility. These factors were initially considered in the Software Quality And Strategy Document and can be found within the External Quality: Building The Right Product section. How we would implement these features were then explored in the Design Document. The software aims to be perceivable, operable, understandable, and robust.

When creating the website, we explored existing games and analysed the users' critiques. Often, they tended to complain about how complex it was to: navigate through the website, learn how to play the game, and understand the language used. We aimed to create a website which was clean and simple. Anything we felt wasn't necessary in understanding the game and how to play it wasn't added to avoid clutter. Furthermore, when designing Nexus, we also avoided using jargon and instead used simple to understand language, both on the website when explaining what Nexus entails and in the game when explaining how to play and what each wild card can be used for as shown in (See Appendices 7.X). The purpose for this was to avoid making users feel like novices when playing the game and to accommodate to a wider audience whose first language may not be English as well as our target audience which is focussed to teens. Another example being offering sound options to aid those with visual impairments as well as general users who may need additional help in understanding where they are in the game. The landing page of the game plays a pirate themed tune to introduce Nexus and let players know the game has loaded. This sound feature can be muted if the user prefers. Furthermore, when playing the game, users will hear a clicking sound to confirm they have placed their token on the grid. In addition to this,

since Nexus has adopted a dark nautical, pirate theme, we ensured this didn't affect the usability of the game. The colours selected both in the game and on the website starkly contrast each other which makes it easier to read. The novelties presented in the game appear by taking over the screen. By doing so, users are clearly able to read what is presented to them before having to decide. We also avoided flashing images and patterned effects for potential users with photosensitive epilepsy.

The way in which the software was designed makes it accessible and easy to navigate for people of differing ages and varying cognitive abilities. How to play the game can be found both in the game and on the website. The navigation bar of the website clearly indicates where users can download the game. Once the game has been launched, users can begin playing the game. Along with the details above, the software is perceivable as it makes use of familiar user interface components such as buttons and checkboxes. Since we were aware of the potential perceptual barriers users may face when interacting with the game, we purposely chose to use large font sizes, clear to read fonts and provide a description when images are used as shown through the wild cards (See Appendices 7.X)

2.4 User evaluation

We believe that the target audience will be satisfied with the game as it was intended. The game hosts a pirate theme that will appeal to teenagers with fun graphics, immersive music, and a competitive environment. Since the novelties presented within each game appear at random, it ensures that no two games are alike, thus increasing its replay value.

The game is challenging as the player is not only fighting against their opponent(s), but the game itself. All throughout the game, dead tokens will randomly pop on the game board, rendering the tile it sits on unusable, which could ruin the player's strategy. In addition to this, Nexus also introduces the concept of wild cards, of which there are two types, 'Power Cards' and 'Doom Cards'. These cards are designed to give the users a competitive edge against their opponents when they become available. Players are encouraged to use them to bring a level of uncertainty to the game, having their opponent second guessing their next move. Power Cards help the player by altering the way the game plays, for example giving the player an extra turn. They can be held for later use or used immediately. The player will be given the option to select one of two power cards. Once the player has selected the power card which best suits their strategy, they will be able to view their selected card on their respective card holder. On the other hand, we have the Doom Cards. They work against the player by forcing them to choose a "punishment" of sorts, like losing their turn in the game. The Doom cards come into effect as soon as the player selects them.

The game is highly intuitive and therefore easy to play and enjoy. The objective is simple, to connect five tiles together either horizontally, vertically, or diagonally. Each player has a set coloured token during gameplay. When placing tokens on the board game, the player's colour is used on their tokens, making it easy to distinguish them between the players. Dead tokens are all grey in colour. The wild cards are easy to understand, as they all come with a short description of what it does when used. The wild cards are presented in a concise

manner to ensure players understand their use upon first glance and can focus on how they want to use the card (See Appendices 7.X)

Nexus has two game modes, Normal and Timed. The 'Normal' game mode poses as an easier alternative to the 'Timed' mode, players can focus on their strategy and enjoy competing against their friends. In the Timed mode, however, the players need to connect five tiles before the selected time runs out, adding an extra element of challenge to the game as they are competing with their friends whilst racing against the clock. We introduced a second game mode to account for the fact that although the game is targeted towards teenagers, the game can be played by anyone. And since everyone has varying cognitive abilities, we wanted to give Nexus players the option to select the game mode which they deem appropriate and allow them to personalise their experience by selecting how long the game will last (See Appendices 7.X)

2.6 Functionality

High quality is achieved due to this game being an efficiently run and entertaining game, with various other additions to ensure that it contains a competitive aspect to it when in comparison to other traditional games. This is demonstrated through the form of power cards, doom cards, which provide numerous types of advantages and disadvantages towards each player at any given time whilst playing the game.

The game does run as it's intended to, when considering the most important facets of the game. These include the background sound being able to run whilst the game is in use, the game being able to generate a randomised code in order for players to join a specific game room. A key function that's crucial to the game is simply being able to link together the tokens within the grid as it's intended to. Another instrumental function of the game is the power and doom cards so if they don't work as they're supposed to it removes the excitement factor as well as an element of competitiveness between both players.

In terms of the requirements within the specification a fully functional turn based multiplayer game is needed to be created, with an element of competitiveness, along with community building features. So, this game meets those very specific requirements as nexus itself requires a constant switch in turns, with users making their move and then needing to wait for the other player to have their turn as well.

For the competitive aspect nexus provides a timed run feature within the game which is essentially an additional game mode where players are having to fight against the clock as well as each other to win the game. Finally, with the last requirement, Nexus provides an interactive community through its online forum where players can engage with one another in conversation to decide who they think is the best player overall.

The game nexus is suitable for visually impaired users as it provides relatively large fonts so that it's easily viewable for players. This is suitable for certain audiences that are at a disadvantage, as they would normally struggle to read the text but now, they'll be able to read it clearly with no difficulties. Another aspect of the game that makes it suitable for the target audience would be the use of colourful images which help to engage the younger audience whilst playing the game.

Regarding accuracy within the game, it offers features such as power and doom cards that are known as wildcards that occur at any point whilst playing the game. This is something that demonstrates accuracy as it occurs to both players rather than one player gaining an advantage or disadvantage throughout the game. One of the key forms of accuracy within nexus is through the timed run game mode, which is where users can select the time frame in which they'd like to play, ranging from 20-40-60 seconds per match. Another form of accuracy is presented through the dead tokens which is what hinders both players whilst playing as it's a token that neither of them can claim unless they acquire a specific power card.

In terms of the functionality being secure, we have used a variety of different security measures such as providing a profanity checker to ensure that users aren't able to enter in any forms of profanity within the game, otherwise an error message will occur. We have also used Server sided checks to stop users from attempting to override the other players turn by selecting a token that they player wouldn't normally choose. Another form of security is displayed within the game rooms, so when a player wants to create a game, they can use a specific password to create a private match from other users.

When complying with functionality, the game provides a high resolution for users that contain either the 720p or the 1080p compatibility, with 1080p being the highest form. Other areas of compliance involve the creation and use of player accounts through the sign up, login or guest menu page which is what's needed to create a game. The game is consistent as it demonstrates a clear goal to be victorious when playing. It offers various pop-up messages that occur for example, when players are creating an account, they'll need to choose a suitable name otherwise an error message will pop-up, informing players that they've used profanity. Another message that would occur is when taking too long on a turn, a prompt message will appear signalling to players to speed up their turn otherwise they'll forfeit the game. Another form of compliance is through the multiplayer section, with Nexus offering up to 4 players compatibility as one of its key features. Another area of the functionality being compliant is narrowed down to the game being able to perform successfully without any crashes or bugs, but instead a smoothly run game (10 Ways to Ensure Your Video Game Passes the Compliance Test, 2022).

Nexus incorporates an element of interoperability within the game, since it allows gamers to compete against each other on different platforms, be it either laptop/ computer, or mobile and tablet. Now whilst this isn't one of the key factors of interoperability, for future updates we can incorporate more forms of Interoperability through allowing gamers to use their own designs and concepts created from other games and embed them into ours. This isn't something that is available now due to the hard coding of colours within the programming. However, for future versions of the game, it is something that can be implemented over time, as it can help to provide an immersive experience for gamers that would like to mix different games with one another (Interoperability Key for Next Generation Gaming, 2022).

2.7 Usability

Usability measures how easy it is for a user to navigate a software system. To ensure that our software system is user-friendly, we made sure that the user interface was as simplified as possible. It uses standard interfaces elements including but not limited to input controls

like buttons, and navigational components like the '?' icon (User Interface Design Basics | Usability.gov, 2020).

Once the user presses play, they are welcomed with a standard menu. There are six input controls in the form of buttons and when each of them is pressed, users can enter their name, enter a new room, join a room, create a room, and use the server browser instead. There is also a '?' button where users can receive help if needed. We have also made sure that the dark text is contrasted with a light background to cater to users that are visually impaired.

The simplicity of our game has many benefits. One is that users will not need to 'relearn' how to play it if they do not use it for a long period of time. This is important as it will retain our users and allow them to keep coming back. Also, users will be able to accomplish the game's objective on their first try of using the software. The user will either win or lose, and they can use wildcards to their advantage whilst playing the game.

Although our game is usable, there are improvements that can be made.

One improvement that can be made is the input lag (latency). This is the delay between the action of the user (input) and the reaction of the server supporting the task. Having a usable game is not enough as lag will still discourage the user from wanting to use the software. Although the objective of the game would have been achieved, it should be achieved in an appropriate amount of time. The player would then be more likely to return to the game.

Another improvement that can be made is to introduce error messages into the game.

Error messages are indicators, they inform users that a problem has occurred and provide solutions for resolving it. A good error message has three parts: recognition of the problem, cause of the problem, and a solution. (7 steps to design error messages, 2021). In our game, if a user clicks when it is not their turn for example, a message could pop on the screen reading 'You cannot place a token when it is not your turn'. This message recognises the error, what caused it and implies a solution. The error message is not harsh, but significant so that we can interact with our users and establish relationships with them. Humans make mistakes so it is understandable (7 steps to design error messages, 2021).

The design and look of a user interface are important, as well as the usability of it. To improve it, user's opinions concerning the design of the interface could be collected through a questionnaire for example. A beta version of the interface could be sent to users along with the questionnaire. This would ensure that the design of the interface is designed in accord with the user's requirements and recommendations (Maatuk, 2016). The questionnaire could include questions like 'Did you like the design of the user interface? If not, please tell us how we can improve.', 'Did you like the colour scheme and the design theme? If not, please tell us what you would prefer instead', and 'Did it take too long to load the game?' (30+ Website usability survey questions to understand user behavior | QuestionPro, 2022). After those improvements are made, the updated version of the software will be sent to the user with a new questionnaire. This process can be repeated until most users are satisfied. Figure 1? shows the proposed framework to improve the usability of the software. Overall, this will increase the usability of the software system as it would be exactly how the users want it to be.

2.6 Efficiency

A handful of resources were used to record our tasks and communications such as Discord, WhatsApp, and Trello as well as regular meetings in person, (See Appendices 7.X). Such resources facilitated communications online rather than in person, being able to research concepts and ideas to the game such as graphics and audio features and improving time management. Meetings enabled the team to provide feedback to the game as well as discover potential features and bugs that could occur. These were noted in our WhatsApp group chat for reflection and solutions were posted onto our Discord for easy access and recall, (See Appendices 7.X)

Utilising Unity ensures the best and cost-efficient game engine is being employed to create the software, thus achieving specifications for instance, multiplayer capabilities and hosting on an online server. Unity was used to create the game software, however, understanding the programme's language (C#) was a must to bring its full productivity, which meant uncovering individuals with potential skills in the team to establish roles within the project. This ensured members were providing efficient results in their performance based on their talent, supporting the development and the ease of diagnosis of issues. Our choice of Unity was based on the team's skill such as coding knowledge and integrating graphics, and tools (Asset Store) in addition to its ease of use.

Furthermore, the team was split into subgroups based on their skill set for instance report writing or graphics. We found splitting the group into subgroups enabled efficiency when it came to delegating roles for submission as we knew what was expected from each subgroup, yet we also made the effort of supporting colleagues where possible and not just focusing on our primary role, this is illustrated in (See Appendices 7.X) The Trello boards highlight how despite working on our individual tasks, we made sure we worked together on other aspects to ensure each member was involved in many aspects of the project.

2.9 Security

To enforce security aspects within the website and game, we decided against saving passwords in plaintext and instead opted for SSL, displaying, and storing passwords for users as a hash. Furthermore, to protect the integrity of the game, (See Appendices 7.X) (login page) shows how only those users with an account can access the download page as well as the forum, chat box and development blog. Nexus generates a unique game id code when creating a game ((See Appendices 7.X) - game id) and provides further security and privacy for users by allowing them to allocate a password when creating a game to prevent unwanted users from joining and ruining the experience ((See Appendices 7.X) - create a game password). Since users are required to create an account before being able to play the game, security is ensured as the player's details are stored, and their account can be reviewed at any time.

An area for improvement includes ensuring there is no data at risk of unauthorised access. As of now, if users do not enter a password for their games, unauthorised users may be able to intercept the game if the game id is leaked. Furthermore, since the chat box is available for all users, if someone shares sensitive information it can pose as a threat as we've not filtered what can be posted. However, since the server is secure, it cannot be intercepted or

tampered with. The IP addresses of our users are protected as is their location. Since we require a minimum of eight characters, one capital letter and a digit for an accounts password, it ensures they aren't too simple and thus will be harder to hack. As a team we enforced many security testing measures. An example being ensuring only one email account can be used per account. This is because it warrants that a user cannot have a spoof account to, for example, verbally abuse others. Another feature added is a text profanity filter to prevent inappropriate text from being entered, this is to ensure that the social environment is safe and appropriate for our target audience: teens. Another security testing measure we established was penetration testing to essentially draw out any faults within Nexus to be fixed. A few faults were discovered such as clicking continuously would cause the server to crash, however this is no longer the case. Nexus runs on a secure server as it uses a Secure Socket Layer. This enables all moves the player makes to be protected from unauthorised users tampering with the game. This can be found on the server browser as well as by entering a leaked game code. Nexus runs on a digital ocean server because it has low bandwidth, and it allows the game to be scalable which makes it trustworthy. For example, if we were to add a feature in the game such as 8 players instead of 4 the server would have no problem dealing with this change. Due to the SSL protocol, messages sent between one another will also be secure as it provides encrypted and safe communication.

3.0 Teamwork

Our team used various platforms to communicate with each other. Group chats were made on these platforms, and each one had a different use.

Trello was used to track the progress of the tasks we assigned to each other. We used the Trello boards by adding what was discussed in group chats onto it, and then ticking it off when the task was complete. (See Appendices 7.X)

WhatsApp was our most used communication tool. It was used to plan meetings, talk about our progression, delegate roles and even for general discussion. At the beginning of this project none of us knew each other beforehand but, as time went on, we became comfortable with our frequent communication on WhatsApp. Saeed and Adam M developed the game, Werner was the graphic designer, Naomi and Adam M developed the website and others including those already mentioned contributed to all documents. (See Appendices 7.X)

Discord and blackboard were used for video calls. As well as Discord being used as a platform for us to hold team meetings, it was also used for us to share our progress. Werner, our graphic designer, frequently uploaded pictures of the design of the tokens for the game (See Appendices 7.X) The team would give him feedback and he would take it on board. It was also used to share our code depository (See Appendices 7.X)

Blackboard was used strictly for video meetings (See Appendices 7.X). Firstly, we would use WhatsApp to determine a suitable time to meet for all team members. When this time was decided, we would decide whether we wanted to meet on Discord or Blackboard. There were also several in-person meetings. Most of these meetings were rehearsals for the hack

and the final demonstration. We would spend hours going over our lines and testing the game, making sure that the real demonstration was as close to perfect as possible.

Although most of our sessions were productive, there was never a day where no team member was late. In the short run, lateness did not affect the productivity of our meetings. However, in the long run, lateness is not something that should be 'the norm' or acceptable. In teams, all members should respect people and their time, and make it their responsibility to show up at the time they were meant to. Another thing is that some team members took a while to reply. Sometimes team members came unprepared to meetings. They haven't made any progress from the last time we met, or they didn't memorise their words for the demo for example.

We found that the easiest way for us to decide on things was for one or two people to come up with an idea, present it in the group chat, and then others would give their feedback on it. If the whole team agreed, then we would proceed with the idea(s). Our team does not agree with the notion of going with what the 'majority' wants as every team member's opinion matters. We will compromise for a team member if we must.

Overall, our communication was average. Even if team members took too long to reply or just did not reply at all, this was not a very frequent thing. Most of the time we were able to communicate effectively.

3.1 Project management

Scrum was used to structure our project, delegating roles based on the team's skillset. By utilizing our strongest talents, the team was able to cross-relate in different aspects of the projects, from creating the website, design and quality documents, presentations, and the software with the utmost quality and efficiency. As the team was made up of 8 members, it was decided that a minimum of two should be allocated on each assignment. This considers factors such as supporting, helping each other where need be in the case of any potential obstacles, and learning from each other based on competencies. However, this proved to be flawed for the team. (See Project Management 4.3)

Team planning and scheduling meetings were required frequently in order to achieve project goals (See Appendices 7.X) (see Trello board and meetings) and allow members extra time for deadlines, in consideration with other modules and out of university life. Meetings were scheduled either online or in-person to discuss progress (See Appendices 7.X) (See GitHub, Discord and Trello). A way to improve communications was to set regular meetings every fortnight to track deadlines on a regular interval, in addition to providing updates and attendance for the team as a whole, whereas meetings were mainly based on members availability during the upcoming weeks. Having a set schedule would confirm upcoming dates ahead and ensure the majority of the team can attend. (See Project Management 4.3)

The team used different processes to track tasks that enabled reflection based on what had been implemented and what was to be achieved. Trello enabled us to categorise our tasks, and members could create or update them when need be. The team can then also be updated on assignments even if they were not required and see each other's roles or tasks.

GitHub was used for software tracking and development which enables the lead programmer to record what has occurred, such as issues, and actions as well as the source code. (See Appendices 7.X)

The use of the forum was intended to incorporate a source of feedback for the game, an additional platform for community members to communicate with the team and not just using the chat box or Discord.

Discord was used to discuss feedback from mentors to improve the project. Meetings were scheduled online based on convenience and to enhance time management. The Discord has channels for the software design and build for which the team can communicate through chat or voice channels to work on the project. In addition, members were able to work on the project remotely, but still have the support on hand when needed. For example, prior to submitting documents, proof-reading was a must from a handful of members to amend any mistakes. Using Discord enables users to share their screen and work together on documents as well as the software.

4.0 Possible improvements

4.1 Software quality

In terms of the quality of the software, a few other areas that could have also been improved involve increasing the amount of debugging of the software. This would've been helpful as it would indicate if other areas needed addressing, for example testing the different game codes presented within each game. This would've been helpful as we'd understand how random the codes are, or if they contain any similarities with one another. Another form of debugging that we could have tested against was the occurrence in which a dead token appears within each game. This would've been a good area to check as we'd be able to gain a clearer perspective as to which game mode they appear more frequently on.

Regarding improvements within the quality of the software, we could have implemented more quality control at the start of the project, so that when tests were being made, we would've been made more alert of any issues. This could've taken place through the form of regularly checking each game mode to see if there were any discrepancies, such as an incompleteness of tokens for each player, the amount of power/ doom cards occurring for each player, to see if they were the same or different. Another way to provide more quality control would've been to check if the game was even able to be completed successfully once a player was able to link their tokens together and then wait for the game to determine that they were the winner or not.

Another area of improvement regarding the game is that we could've been better prepared when creating it for example, had we addressed the key specifications of the game earlier, then maybe we could have spent less time working on it. So had we addressed the requirements sooner rather than later, we could've reduced the amount of workload we had to do, for example, improving the multiplayer features from 2,3 and 4 players compatibility. If we had this as one of our main priorities then we would have achieved this much sooner, which in turn would allow us to focus on implementing other game modes more freely without a designated time frame.

A potential aspect that maybe could've been addressed is the use of another methodology when developing our game, Nexus. For example, we could've used the agile methodology, which is where the process of creating the game is split up into iterations that are relatively short. So, when in development the game focuses on the key features straight away to ensure that its key features are addressed properly before all of the additional ones that can be implemented later on (Eden, 2022 Agile in Game Development | Melior Games).

4.2 Teamwork

A way in which we could've improved teamwork was to have better and fluid communication. What this means is that all team members openly communicate on a routine basis. This would help improve our project since if all team members were in sync, it would mean that everyone would be on the same page in terms of scheduling meetings, producing cohesive work, and meeting deadlines. Furthermore, having transparency in communication ensures that members are on par with the project goal and know what is expected of them. It also helps build trust within the team which allows everyone to work better together and builds morale for the entirety of the project. We could've also respected and adjusted to everyone regarding their communication styles as everyone is different which means that some members may prefer communicating for the sole purpose of achieving a goal whereas others may want to build a bond and get to know the team. We should've set out a communication plan (Laura LaPrad, 2018) which clearly outlined who is responsible for each task, how information will be shared and that the team is looped in on each aspect of the project. This would've helped improve the project significantly as team members could've referred to documentation which was written for the project as well as deeming necessary expectations for when and how updates will be shared, emphasising the project status, allowing for team members to chip in their feedback to allow their voice to be heard, pushing for productivity and engagement in meetings. The way in which this could've been explored further was to have more frequent meetings (in university, blackboard, or discord), have a google document to allow discussion and live edits, collaboration apps like Trello and to-do lists.

We could've also broken the work in more divisions because it allows for more work to be done efficiently and everyone knows what part they're working on. For example, in our project, for the programming aspect, we could've had a lead and senior developer with other team members being developers under them. This would've meant that everyone would be contributing, with the senior and lead developers helping and guiding them. Another way in which teamwork would have improved was to have two people coding at once this means that in some way, one person would be able to see the other's screen either through discord or a video call etc. The benefit of this is that you can catch bugs easier as the person who is looking at the work rather than physically typing it can catch any errors the coder makes as well as overlooking to help make any changes. Another benefit is that with pair programming, both people are on the same page as they're working hand in hand. This would encourage and improve communication as you're engaging and working together. (Wiki.c2.com, Pair Programming Benefits, 2006)

We could've improved team working further by having strenuous testing on the software and more testers to carry them out. We lacked in this field as we distributed the workload and only completed what we were assigned with. This meant that the programmers were left to test by themselves with a team member helping them out, but it would be more effective if more thorough testing was achieved with the whole team rather than task assigned members. Having more cooperation in terms of willingness to work together and not alone during tasks would've led to less problems occurring as there was no sync and whilst trying to complete tasks, could've gained more help and insight from teammates. Also, it would encourage more contribution and shared responsibility, this means that more tasks would be completed to a better degree and there wouldn't be a lack of someone falling short in trying to complete a task as they have a team member there to help them. Furthermore, limiting and lowering the workload imposed on each member would have dramatically improved and relieved the stress and pressure some team members were feeling whilst trying to complete their tasks before deadlines. It also would have ensured that the team wasn't overwhelmed by the workload and that they could work efficiently. By doing this it would've increased performance rather than lower it. If we had mixed and matched tasks, this would've improved the fact that everyone would know what is going in in the project and across the board. Not only this, but we could have allocated each team member a main task and sub task(s). That's because the sub task would've been more detailed in what to do and this is helpful when working on a large project.

In addition to this, had we reviewed our peer's work more often and thoroughly this would've helped improve and alleviate our standards as well as the quality of the software. By reviewing each other's work, it allows there to be more than one opinion and a different viewpoint which means that any errors or ideas can be discussed and agreed on for the betterment of the software. It would also increase consistency as everything can be observed by different team members rather than one, as that one person who is managing and overlooking work can't possibly hold someone accountable as they're not always looking.

Another major improvement which we must adhere to and focus on is that during meetings, more team members need to participate, engage, listen, and contribute. This is because everyone's voice needs to be heard, not just the same two people. Having one or two people talking limits the feedback and opinions which could prove useful to the team, project and software and isn't productive as well as a waste of time. Gaining everyone's opinions and thoughts can lead to a greater understanding of the tasks and potential problems.

Having more task specific meetings would've improved team working as those working on the same task can engage and collaborate to listen to each other's vision, perception, ideas, and feedback. This would have ensured that those on the same task know what to do between each other and aren't confused. Also, having a specific meeting allows there to be focus on that task rather than a meeting where many things are discussed so it helps to concentrate and gauge information.

Improvement in team working could've been achieved by avoiding personal criticism of team members. Constructive criticism differs from this, it is effective as it helps the team members know what can be improved or a different view on how to do things whilst being polite, however personal criticism isn't healthy and can cause problems within the team. This

directly has an impact on the work and could affect the quality of work and the software. If a team member has personal criticism, they should either talk to that team member in private or avoid mentioning it.

What could've been improved as a team was the team morale and mindset of wanting to succeed on the project and module. Some team members weren't bothered enough and so, they did the bare minimum. This conflicted with those who did want to succeed as they had to work around and try to do what they could to keep the team spirits high as well as taking on more roles to ensure the quality of work was to a high standard. We could've improved by focusing on the team's strengths and to understand that everyone is unique and distinct. We should've extenuated each member's skill set and respected each other to maintain a healthy and calm environment rather than a stress or force filled one.

4.3 Project management

A way in which we could have better managed this project was by ensuring the documents, required for our earlier submissions, were created in a manner which would allow us to draw ideas on how to best approach future tasks. The purpose of a design document is to cover all aspects of the product's design. This documentation should include information about users, product features, and project deadlines. Although the design document did outline the information listed above, it failed to delve into the 'how to' aspects of design, rather, its focus surrounded what features the game would have and why. We focussed on creating a design document which would act as a means of 'selling the game' to our stakeholders, however, we should've aimed to demonstrate our approach: how we would achieve our end goals. This in turn led to discrepancies when creating the game as the team had to research what they individually believed was the best way of implementing the design as well as executing it. Considering that product design is a collaborative process and documenting it correctly would aid the implementation of it (Babich, 2020), it would've been better to brainstorm as a team during the creation of the design document to decide which method, when implemented, would lead to the result we were looking to achieve. An instance in which the documents assisted in the implementation of the game design is through 'graphics', section 5.5 in the design document. This section walks the reader through the design, colour, theme, and key motifs the game elements will adopt. The software quality and strategy document complete this as it delves into why these choices were selected and the effects and ambience, they intend on creating. On the other hand, had the rest of documentation been written clearly, by answering 'how should we build this' and 'why should we build this', the team would have been able to monitor which requirements we've satisfied, and which are yet to be completed. This would have cleared a lot of confusion and allowed members to complete their tasks to a better standard. An example of where the software quality and strategy document lacked detail was regarding the Development System. The software we intended to use for the game and website are mentioned simply without justifying why they would be the perfect fit for our game (Nexus). Rather than designating two people to write the documents, the team should've collaborated to break down all aspects of how to implement this design, not just how it will look and where it will appear. The forum and chat box feature are mentioned without explaining where it can be accessed, who can access it and what tools will be used to implement it. Had these details been defined in the design document, those working on the website and coding the game would have a firm understanding on how to approach implementing such features in an appropriate manner.

The team agreed on using Trello as our main tool for tracking progress. This collaboration management application offers further tools which could've aided our team in understanding what the project's end goal was. Trello allows for the creation of boards, lists, and cards. It allows users to customise and expand with more features as the teamwork grows (Trello, n.d.). By optimising the features Trello offers, we could manage projects, organise tasks, and build team spirit in one place; it would allow us to plan ahead and not leave tasks so last minute. This would make it easier to access and refer to when needed. Furthermore, it would be especially useful for when the team cannot meet up in real life, as Trello can be updated remotely, allowing the entire team to collaborate virtually in real time. An area of concern for our team was our poor communication. Since everyone had different timetables and commitments, we often found it hard to: meet up in real life; chase up teammates to find out what's been completed, what is incomplete, and when the deadline is. Only after making use of the deadlines and boards feature on Trello, were we able to define subgroups within the team and track our progress, (See Appendices 7.X) in the appendices illustrate this. This allowed us to see what was left to be completed regarding the remaining time till the deadline. Besides this, an issue which could've resolved other problems was that we would host meetings without having a plan of what we wished to achieve. By not setting an outcome for the meeting, a lot of time was wasted. Sometimes we'd leave submission tasks at the very last minute and try to resolve this by holding a meeting without any consideration for how we'd manage the time in the meeting and what we'd discuss. However, had we created a list on Trello of topics to cover during the meeting, we would have a structure ensuring all points were covered. This would also allow for members to prepare notes and questions to bring up during the meetings, opening further discussions.

Although splitting the team into subgroups to work on different aspects of the game sounds convenient, with hindsight, it proved to be unsustainable. This was because it disrupted our attempts of having a clear workflow procedure. By dividing the group, we were limiting everyone's understanding on the parts they weren't assigned. A study found that subgroups are based on fault lines; team members who share specific characteristics bond with each other to create a subgroup (Subgroups in Agile and Traditional IT Project Teams, 2018). An example of a characteristic which could define a subgroup can include demographic and knowledge. Both are reasonable characteristics which we as a team could've used to divide the group into subgroups. Demographics would include those members of our team who have experience playing games and can distinguish not only the key features users would be looking for in Nexus but also cater to the niche aspects too. By splitting the team according to their knowledge, those with experience in graphic design, coding, report writing etc could be placed into their respective subgroup. The reason we originally chose to implement subgroups was because it allowed everyone to 'self-manage' their respective group. This was befitting for us as we were all on different schedules however led to discrepancies in our work as there wasn't anybody overlooking everyone's output to ensure everything aligned. We didn't deem it necessary since we were all drawing on points agreed in the design document and software quality and strategy document. However, we failed to account for the fact that everyone's interpretation of the documentation could differ which led to clashes. Something the team has learnt from this experience is that having a clear workflow procedure was more of a priority than finding an easy way to get the work done. Since product quality was the ultimate priority, we should've first established the best method for ensuring consistency and clarity. If we still wanted to use the subgroup method, we should have organised a means for exchanging our work and checking for disparities.

However, as mentioned above, the team found it difficult to meet up regularly, and so, we would probably opt for online group working sessions. If we extended a similar method as used for group coding collaboration (we used GitHub), perhaps a collaborative google doc (for the design document and software quality and strategy document), it would enable us to complete tasks together without having to meet up as frequently.

Another way in which optimal project management could've been achieved is by resolving issues and overcoming hurdles quicker. Forbes Technology Council has ranked 'communication breakdowns' as the second most common software development obstacle (12 Common Software Development Obstacles And How To Tackle Them, 2019). As they've stated, all members should be aware of the project's coding strategy, goals, and objectives. This is fundamental in managing the fallout which would impact our output. The problem we faced as a team was the confusion surrounding our coding styles. It was personal to each member which made it difficult when attempting to clean up the code and edit it to fit in new parts of code. This could be resolved by adding in comments to explain what each line of code does, and to make it clear when one section of code ends by neatening the indentations. Another way to achieve a cohesive, clear coding style would be by giving variables obvious, related names. This would allow for easy recall and make it simpler for those editing the code. Furthermore, since multiple people were editing the same source code, good version control should've been implemented to protect individual code contributions from being overwritten and consequently lost. Overwriting code is a permanent decision and since this project has been new for us, it is important to have a copy of the original code to refer to and maybe even bring back into the source code. An instance where this can be illustrated is, at the very beginning when working on the hack, a large chunk of source code had to be rewritten. However, after a discussion with the team, they realised the errors were easily avoidable had they used a more modular approach. Since Unity allows for building interdependent coding architecture, it led to performance issues when trying to find an object in the hierarchy. To optimise our time better, since then, the team has used GitHub as a means of managing and maintaining good version control and to accommodate pair programming. GitHub offers features where the coder can easily explore the changes made. Since it offers sections to insert data and notes as well as coding scripts, it has prompted our coders to make use of these features to ensure those pair programming are aware of why certain lines of code exist, what they do and if they link, how they link.

5. Conclusion

To conclude, the way in which we approached this team project was by focusing on the criteria which needed to be met for each submission rather than thinking about the game objectively; we used the submissions as a to-do list. Had we planned each aspect of the game from the beginning, we would be able to consider a wider range of issues: for example, accessibility for a larger audience including those with disabilities. This could be established in the font selected for the game, ensuring it is clear and legible to most people. Or, perhaps in the colours selected for the game tokens by using colour-blind friendly colour combinations- so we would steer away from using blue and purple together or red and green (Collinge, 2017). Instead of using the submissions as a to-do list, we should have used it as a means of ensuring all requirements were met: a list of our priorities, our baseline not our end goal. By doing so, it would highlight the need to focus on, for example, making the game compatible for multiple players but also push us to think about who those players may be,

and what additional support they may need. Another example of how our approach to this module affected our outcome, which has been explored above, is in the way we managed the project. By considering future for the game, we could've better prepared the way in which distributed tasks, handled disputes, and organised meetings. By creating a sustainable approach, like using Trello to manage the project's tasks or a collaborative calendar to note everyone's availability for meetings, we could've optimised our time better.

6. References

- Babich, N., 2020. *Design Documentation: Why You Need It*. [online] Adobe XD Ideas. Available at: <<https://xd.adobe.com/ideas/principles/web-design/design-documentation/>> [Accessed 14 March 2022].
- Blog | GlobalStep. 2022. *10 Ways to Ensure Your Video Game Passes the Compliance Test*. [online] Available at: <<https://globalstep.com/blog/cheatsheet-to-ensure-your-video-game-passes-the-compliance-test/>> [Accessed 1 April 2022].
- Collinge, R., 2017. *How to Design for Color Blindness*. [online] Getfeedback.com. Available at: <<https://www.getfeedback.com/resources/ux/how-to-design-for-color-blindness/>> [Accessed 27 March 2022].
- Eden, 2022. *Agile in Game Development | Melior Games*. [online] Melior Games. Available at: <<https://meliorgames.com/game-development/agile-in-game-development/>> [Accessed 1 April 2022].
- Medium. 2022. Interoperability Key for Next Generation Gaming. [online] Available at: <<https://medium.com/play-to-earn/interoperability-key-for-next-generation-gaming-d38f399a42f3>> [Accessed 1 April 2022].
- Trello.com. n.d. *Trello*. [online] Available at: <<https://trello.com/en-GB>> [Accessed 14 March 2022].
- Research Gate. 2018. *Subgroups in Agile and Traditional IT Project Teams*. [online] Available at: <https://www.researchgate.net/publication/320567168_Subgroups_in_Agile_and_Traditional_IT_Project_Teams> [Accessed 27 March 2022].
- Forbes. 2019. *12 Common Software Development Obstacles And How To Tackle Them*. [online] Available at: <<https://www.forbes.com/sites/forbestechcouncil/2019/06/05/12-common-software-development-obstacles-and-how-to-tackle-them/?sh=7d8576445e01>> [Accessed 8 March 2022].
- Usability.gov. 2020. *User Interface Design Basics | Usability.gov*. [online] Available at: <<https://www.usability.gov/what-and-why/user-interface-design.html>> [Accessed 25 March 2022].]
- Medium. 2021. *7 steps to design error messages*. [online] Available at: <<https://bootcamp.uxdesign.cc/7-steps-to-design-error-messages-49e509f03d18>> [Accessed 25 March 2022].
- Maatuk, A., 2016. *An Approach to Improvement the Usability in Software Products*. [online] Academia.edu. Available at: <https://www.academia.edu/73382659/An_Approach_to_Improvement_the_Usability_in_Software_Products> [Accessed 25 March 2022].

- QuestionPro. 2022. *30+ Website usability survey questions to understand user behavior* | QuestionPro. [online] Available at: <<https://www.questionpro.com/blog/website-usability-survey-questions/>> [Accessed 26 March 2022].
- Small Business - Chron.com. 2022. *Advantages & Disadvantages of Team-Based Organizations*. [online] Available at: <<https://smallbusiness.chron.com/advantages-disadvantages-teambased-organizations-25370.html>> [Accessed 14 March 2022].
- Game Music: <https://www.youtube.com/watch?v=hVBgKCYrI-c>
- Game tile click sound: <https://www.youtube.com/watch?v=h8y0JMVwdmM>
- <https://www.teamgantt.com/blog/project-management-communication-plan>

7. Appendices

7.1 Trello board links

<https://trello.com/b/Y0z8BqmT/hack>

<https://trello.com/b/tqqNLgVy/demo-16-12-2021>

<https://trello.com/b/sQ2vXtdS/design-doc>

<https://trello.com/b/CYVMMjim/graphics>

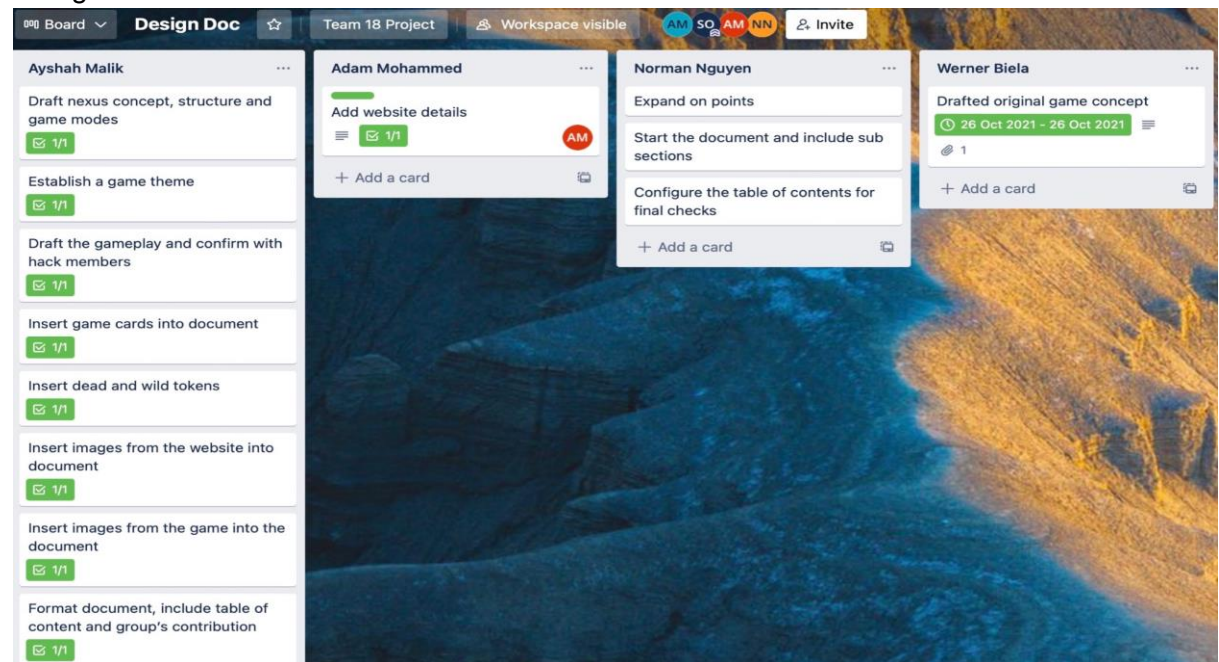
<https://trello.com/b/7uCHTejY/meetings>

<https://trello.com/b/PGqdf5H9/software-quality-doc>

<https://trello.com/b/WliUH1y3/website>

7.2 Trello screenshots

Design Doc Board:



7.3 Discord

Team Project 18

TEXT CHANNELS

general

game-design

game-builds


game-codes

VOICE CHANNELS

general

game-design

Space to talk about the design elements/aspects of the game.



23 March 2022


11:46 saeed

Nexus_Data.zip

37.70 MB

Download icon

saeed



GIF


25 March 2022

10:50 saeed

https://cdn.discordapp.com/attachments/847594187196334091/956850066910507038/Build_Public.zip

29 March 2022

14:55 AAA




31 March 2022

13:51 naomi

i'm gonna have to go guys

16:07 saeed



Team Project 18

TEXT CHANNELS

general

game-design

game-builds

game-codes

VOICE CHANNELS

general

game-builds

Welcome to #game-builds!


This is the start of the #game-builds channel.

14:33 saeed

<https://1drv.ms/u/s!AnIMBwb6oeB5IAsBtiPQamAVy0c8?e=PNvi4I>

Email attachments

Folder



13:39 saeed

Build.zip

37.71 MB

Download icon

14:14 saeed

Build.zip

37.71 MB

Download icon

08:27 saeed

Build_2.zip

37.72 MB

Download icon

09:32 saeed

new build

Build_Online.zip

37.72 MB

Download icon

10:20 saeed

builds_2323.zip

37.72 MB

Download icon

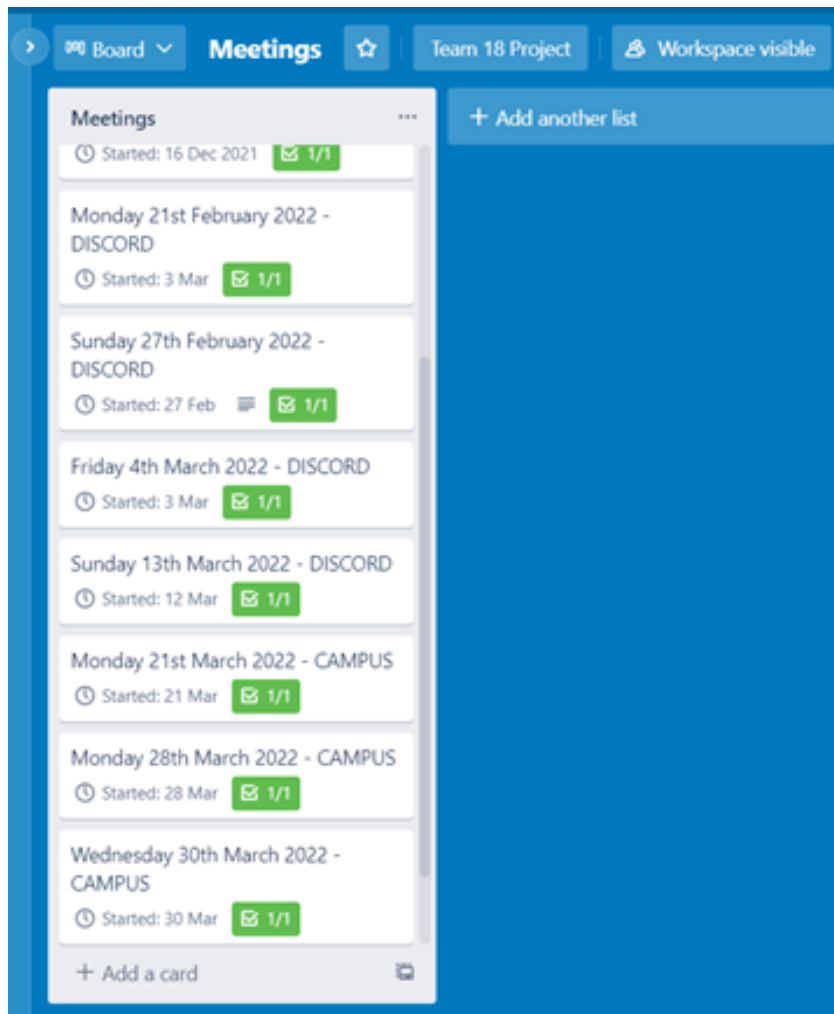
saeed

Nexus.zip

37.72 MB

Download icon

7.4 Trello Meetings



7.5 Quality Strategy Doc board:

The screenshot shows a Trello board titled "Software Quality Doc" with a header bar containing navigation and user information. The board is organized into four columns, each representing a team member's tasks. Each task card includes a description, a progress indicator (a green box with a checkmark and a fraction), and the assignee's initials in a circle. The tasks are as follows:

Team Member	Task	Progress
Adam Ejaz (AE)	Create the template for document	1/1
Adam Ejaz (AE)	Start answering questions from the introduction	1/1
Adam Ejaz (AE)	Answered all questions from the introduction	1/1
Adam Ejaz (AE)	Start answering questions from the internal and external quality sections	1/1
Adam Ejaz (AE)	Answered majority of questions from the internal and external quality sections	1/1
Adam Ejaz (AE)	Answered minority of questions for security	1/1
Ayshah Malik (AM)	Format document, include table of content and group's contribution	1/1
Naomi Eniola Adesiyun (NA)	Improve the table of contents in software quality strategy	1/1
Jashan Gandham (JG)	Improve layout of Software Quality Document	0/1
Jashan Gandham (JG)	Answer questions on software quality document	1/1

7.6 Website board:

The screenshot shows a Trello board titled "Website" for "Team 18 Project". The board is organized into three columns, each assigned to a team member: Adam Mohammed, Norman Nguyen, and Ayshah Malik. Each column contains a list of tasks, each represented by a card. The cards are color-coded with a green header and a green progress bar. The progress bar shows the number of tasks completed out of the total number of tasks for that column. The background of the board is a blurred image of a person's face.

Board: Website | Team 18 Project | Private | AM SO AM NN | Invite

Adam Mohammed

- Change URL
22 Nov 2021 - 22 Nov 2021
1/1
- Change About Page
1/1
- Create a blog
1/1
- Team members section
1/1
- Post on Blog with progress
1/1
- Add game to website
1/1

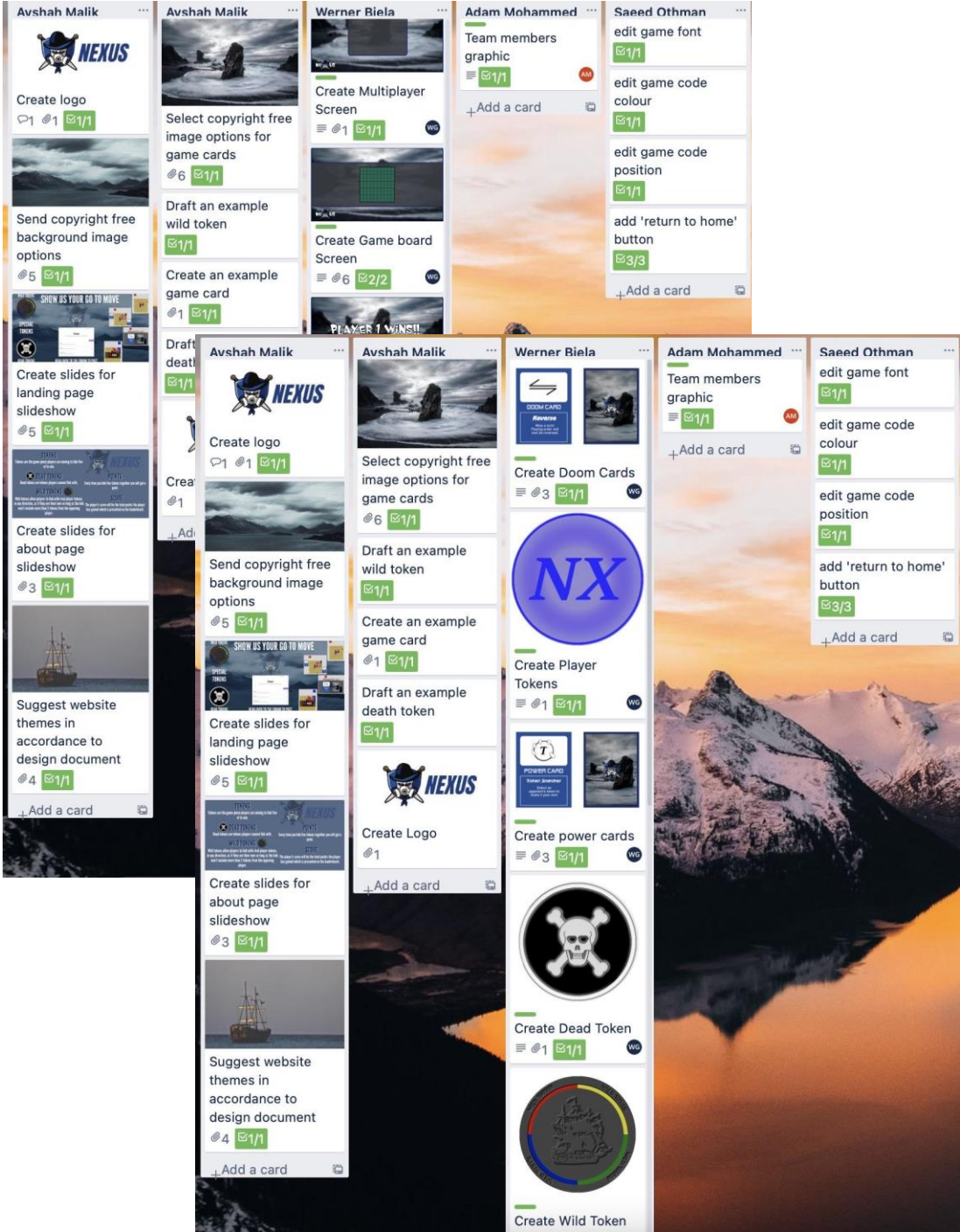
Norman Nguyen

- Create account and bug test
2/2
- + Add a card

Ayshah Malik

- create graphics for landing page
4/4
- create graphics for about page
3/3
- find copyright free background options
1/1
- create an account to test out the forum
2/2
- create graphics explaining each novelty
5/5
- + Add a card

7.7 Graphics board:



7.8 Git hub commits and contributions

The screenshot shows a GitHub repository page for 'Hectahx/NexusGame'. The repository is on the 'main' branch, has 1 branch, and 0 tags. It has 12 commits and 0 forks. The repository description is 'This is the Github repo for Team 18s game for the team project'. The repository contains several files and folders, including .vscode, Assets, MacBuild.app/Contents, NativeWebSocket/Assets, Packages, ProjectSettings, UIElementsSchema, WebGLBuilds, .gitignore, Build.zip, and README.md. The commit history shows various updates, including porting the game to online, fixing server crashes, and adding a server browser. The repository is owned by Hectahx and adam5.

github.com/Hectahx/NexusGame

main 1 branch 0 tags

Go to file Code

About

This is the Github repo for Team 18s game for the team project

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Packages

No packages published

Contributors 2

Hectahx

adam5

Commit history:

File	Commit Message	Time Ago
.vscode	Ported game over to online	4 months ago
Assets	TLS is not enabled on NativeWebSocket so I need to convert it to WebS...	2 days ago
MacBuild.app/Contents	Added a server browser and full compatibility with the websocket serv...	24 days ago
NativeWebSocket/Assets	Fixed server crash due to incorrect game code on client side	9 days ago
Packages	Fixed server crash due to incorrect game code on client side	9 days ago
ProjectSettings	Fixed a lot of things but honestly idk	4 days ago
UIElementsSchema	Fixed server crash due to incorrect game code on client side	9 days ago
WebGLBuilds	Added a server browser and full compatibility with the websocket serv...	24 days ago
.gitignore	Converted to WebGL. Added sound, help menu and displayed code on s...	4 months ago
Build.zip	Added different gamemodes, a timer selection and a password to the g...	2 days ago
README.md	70% of win logic completed just have to do some diagonals	5 months ago

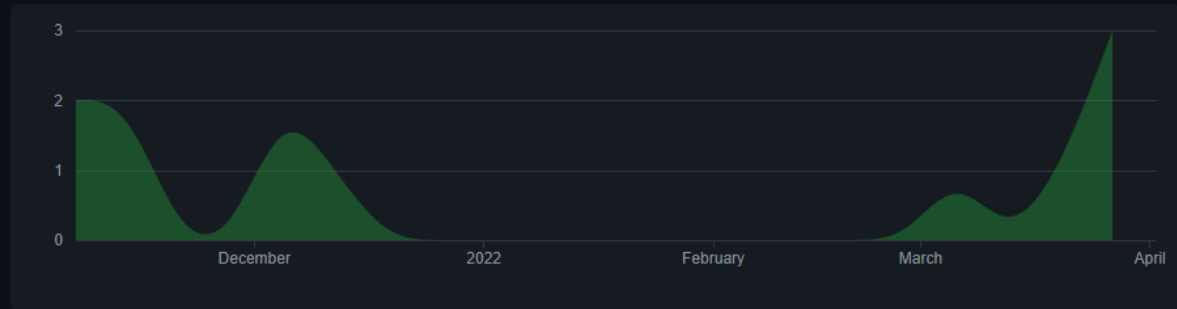
README.md

Team 18 Group Project

Nov 7, 2021 – Apr 2, 2022

Contributions: Commits ▾

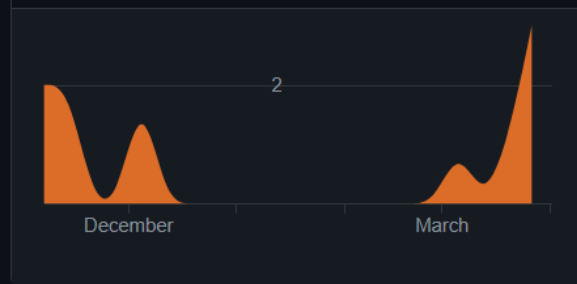
Contributions to main, excluding merge commits and bot accounts



Hectahx

11 commits 1,133,445 ++ 1,049,522 --

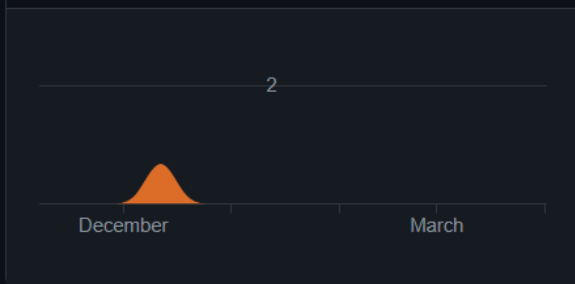
#1



adamm5

1 commit 9,236 ++ 919 --

#2



Commits on Mar 31, 2022

TLS is not enabled on NativeWebSocket so I need to convert it to WebS...

Hectahz committed 2 days ago



87d471f



Added different gamemodes, a timer selection and a password to the game

Hectahz committed 2 days ago



5d52191



Commits on Mar 29, 2022

Fixed a lot of things but honestly idk

Hectahz committed 4 days ago



ed896cb



Commits on Mar 25, 2022

Fixed server crash due to incorrect game code on client side

Hectahz committed 9 days ago



675f82a



Commits on Mar 9, 2022

Added a server browser and full compatibility with the websocket serv...

Hectahz committed 24 days ago



a89d9d2



Commits on Dec 13, 2021

Converted to WebGL. Added sound, help menu and displayed code on scre...

adamm5 committed on Dec 13, 2021



e01a0e5



Commits on Dec 11, 2021

Added Doom Cards / AFK Timeout

Hectahz committed on Dec 11, 2021



dcf9d13



Commits on Dec 8, 2021

Ported game over to online

Hectahz committed on Dec 8, 2021



894af1d



Commits on Nov 14, 2021

end game screen and active player added

Hectahz committed on Nov 14, 2021



7c561e8



Added in Wilcards / Win Logic and fixed the .gitignore

Hectahz committed on Nov 14, 2021



bcc6d8a



Commits on Nov 10, 2021

70% of win logic completed just have to do some diagonals

Hectahz committed on Nov 10, 2021



d6f895c



Commits on Nov 9, 2021

Initial Commit

Hectahz committed on Nov 9, 2021



276136d



7.9 Source code

```
public void CreateLines() //This instantiates all the lines. This is done to later give the user the option as to how big they want the board to be.
{
    int count = 0; //a count of how many lines were made
    for (float x = (704); x <= (512 + 704); x += (512 / lines)) //This covers width
    {
        count++;
        if (count == 1 || count == lines + 1) { } //This if statement is so a the first and last line isnt made on the board as they are already outlines
        else
        {
            GameObject line = Instantiate(VertLine, new Vector3((x * Screen.width / 1920) + (2 * Screen.width / 1920), 540 * Screen.height / 1080, 0),
                Quaternion.identity, GameObject.FindWithTag("ButtonHolder").transform);
            //Adjusted for different screen resolutions
        }
    }
    count = 0;
    for (float y = (540 - 256); y <= 256 + 540; y += 512 / lines) //This covers height
    {
        count++;
        if (count == 1 || count == lines + 1) { }
        else
        {
            GameObject line = Instantiate(HorLine, new Vector3((704 * Screen.width / 1920) + (256 * Screen.width / 1920), (y * Screen.width / 1920) + (2 * Screen.width / 1920), 0),
                Quaternion.identity, GameObject.FindWithTag("ButtonHolder").transform);
        }
    }
}
```

Image 1:
CreateGrid.cs

```
public void OpenHelpMenu() //These show functions are attached to buttons on the UI
{
    Canvas mainCanvas = GameObject.Find("Canvas").GetComponent<Canvas>(); //This finds the Canvas in the Scene and assigns it to a variable
    GameObject mainUI = mainCanvas.transform.Find("MainUIHolder").gameObject; //These 2 lines get the Main UI and the Help Menu as Game Objects
    GameObject helpMenu = mainCanvas.transform.Find("HelpMenuHolder").gameObject;
    mainUI.SetActive(false); //These two enable/disable them accordingly
    helpMenu.SetActive(true);
}

public void CloseHelpMenu()
{
    Canvas mainCanvas = GameObject.Find("Canvas").GetComponent<Canvas>();
    GameObject mainUI = mainCanvas.transform.Find("MainUIHolder").gameObject;
    GameObject helpMenu = mainCanvas.transform.Find("HelpMenuHolder").gameObject;
    mainUI.SetActive(true);
    helpMenu.SetActive(false);
}
```

Image 2:
MainMenuButtonHandler.cs

```
public void signup() //This is executed when a user clicks the sign in screen
{
    var isPassValidated = hasNumber.IsMatch(signupPass.text) && hasUpperChar.IsMatch(signupPass.text) && hasMinimumChars.IsMatch(signupPass.text); //This checks the password matches the regex above
    var isEmailValidated = isEmail.IsMatch(signupEmail.text); //Same as above for email
    if (!isPassValidated)
    {
        Debug.Log("Pass not validated");
        return;
    }
    if (!isEmailValidated)
    {
        Debug.Log("Email not validated");
        return;
    }
    //If both are validated it means that they can signup with their account
    JObject payload = new JObject();
    payload["method"] = "signup";
    payload["email"] = signupEmail.text;
    payload["username"] = signupUsername.text;
    payload["password"] = signupPass.text;
    ws.Send(Encoding.UTF8.GetBytes(payload.ToString()));
}
```

Image 3:
Login.cs

```

if (result.method === "join") { //These if statements are all executed when the server gets sent data and the method is one of the ones listed
  joinGame(result);
}

if (result.method === "play") {
  playMove(result);
}

if (result.method === "cards") {
  cards(result);
}

if (result.method === "setCard") {
  setCards(result);
}

if (result.method === "doomCards") {
  doomCards(result);
}

if (result.method === "timeout") {
  timeoutHandler(result);
}

if (result.method === "leaveGame") {
  //TODO Need to dispose of game properly when a player chooses to leave
  //leaveGame(result)
}

if (result.method === "login") {
  login(result, connection);
}

if (result.method === "signup") {
  signup(result, connection);
}

if (result.method === "guest") {
  const clientId = guid();
  clients[clientId] = {
    connection: connection,
  };

  const payload = {
    method: "guest",
    clientId: clientId,
  };

  try {
    connection.send(JSON.stringify(payload));
  } catch (error) {
    console.error(error);
  }
}

```

*Image 4: Snapshot of
Server's Code*

8.0 Meeting minutes



CS2TP Team Project Minutes of meeting

Project website

Logistics

Time:	4pm
Date:	27/10/2022
Attendees:	Adam M and Naomi
Please Bring/Read:	
Meeting purpose	Discussing what software to use to create the website, what theme it should be etc.

Agenda

Item	Time	Agenda Item	Presenter
1	4 - 4:10	Waiting for people to arrive	
2	4:10 - 7pm	Discuss website plans and make a start.	Adam M and Naomi

Closed Actions

No	Action	Who	When	Status or comment
1	Website plan meeting	Adam M and Naomi	27 th October	Completed

Open Actions

No	Action/Discussion	Who	When	Status or comment
1.	Continuously work on the website and make sure all requirements are fulfilled by submission 2.	Adam M and Naomi	27 th October	Completed

Next Meeting Details: Discuss progress made since first website plan meeting.

Demonstration

Logistics

Time:	8am
Date:	01/04/2022
Attendees:	Nasiri, Adam M, Adam E, Norman, Joshua, Saeed, Ayshah, Werner
Please bring/Read:	Memorise your scripts, Norman, Werner and Saeed need to bring their laptops
Meeting purpose:	Demonstration rehearsal

Agenda

Item	Time	Agenda Item	Presenter
1	9:00 – 9:20	Wait for people to arrive	
2	9:20 – 9:40	Test game, make sure it is running smoothly	Saeed and Adam M
3	9:40 – 9:50	Nasiri will practice presenting the website for the demo	Nasiri and Adam E
4	9:50 – 10:30am	Everyone speaking at the demo will practice	All

Closed Actions

No	Action	Who	When	Status or comment
1	Demo Preparation - fix any bugs in the game and begin assigning tasks for the demo	Everyone	13 th March	
4	Report - ensure roles are assigned to start completing report	Everyone	21 st March	
	Demo Rehearsal - Begin presentation practice	Everyone	28 th March	
	Demo rehearsal - improve game further and ensure presentation runs smoothly	Everyone	30 th March	
	Demo rehearsal - final practice run of presentation, everything is ready, and game is working	Everyone	1 st April	

Open Actions

No	Action/Discussion	Who	When	Status or comment
1.	Fix bugs on the game and make sure the website is finished	Adam M	Whole of March	Completed

2.	Plan the demo and assign roles	Ayshah	Whole of March	Completed
3.	Fix bugs and make sure the game is completed by April 1st	Saeed	Whole of March	Completed
4.	Practice presenting website for the demo and contribute to the evaluation report	Nao mi	Whole of March	Completed
	Practice for the demo and contribute to report	Every one else	Whole of March	Completed

Next Meeting Details: This was our last meeting.

MVP

Logistics

Time:	11am
Date:	10/12/21
Attendees:	Whole team
Please Bring/Read:	Memorise your scripts
Meeting purpose:	MVP rehearsal

Agenda

Item	Time	Agenda Item	Presenter
1	11:00 - 11:33	Waiting for people to arrive	
2	11:33 - until we are finished	Rehearse for the demo	Everyone

Closed Actions

No	Action	Who	When	Status or comment
1	MVP Preparation meeting, start drafting up scripts and roles ready for presentation.	Entire team	3 rd Decemb er	Completed
2	MVP rehearsal - start rehearsing the presentation to eliminate chances of error.	Entire Team	10 th Decemb er	Completed

Open Actions

No	Action/Discussion	Who	When	Status or comment
1.				
2.				
3.				
4.				

Next Meeting Details: On 17th December we will have another rehearsal.

The Hack

Logistics

Time:	11am
Date:	4/11/21
Attendees:	Saeed and Adam M
Please Bring/Read:	Saeed and Adam M to bring their laptops
Meeting purpose:	Make sure the source code is neat and create the screenshot

Agenda

Item	Time	Agenda Item	Presenter
1	11 - 11:05	Waiting for Saeed to arrive	
2	11:05 - 3pm	Work on the source code	Saeed and Adam M.

Closed Actions

No	Action	Who	When	Status or comment
1	Fix bugs by:	Saeed	12 th Novemb er	Completed
2	Create screenshot by:	Adam M	12 th Novemb er	Completed

Open Actions

No	Action/Discussion	Who	When	Status or comment
1.	Continuous improvement of the source code	Adam M and Saeed	Novemb er	Completed
2.	Inform the rest of the team about how the game works, and show the source code so far so that they understand how it works.	Adam M and Saeed	Novemb er	Completed

8.1 GitHub link

<https://github.com/Hectahx/NexusGame/releases/tag/v1.0.0>

<https://github.com/Hectahx/NexusServer/releases/tag/v1.0.0>