20th November 2023

# Requirement Specification

## GROUP 17: REIMAGINE SKILLS BUILD AS AN RPG GAME

JOSHUA PULHAM, NAM LE, JAMES WATSON, CHARLES DUBOIS-VELTMAN, AKANKSHA SIROHIA

# Contents

# 1 - Introduction

## 1.1 - Overview and Justification

Our client is IBM, and our contact at IBM is Mr. John McNamara. We have been tasked to reimagine Skills Build as an RPG game. This RPG game must facilitate accessing IBM Skills Build courses by linking them and their subsequent knowledge check questions, as well as reward users with in-game items (powerups and progression) if they are able to answer.

The game should also be fun and engaging to the user; it should be an amazing study tool integrated smoothly into an engaging RPG experience. However, the game does not need to teach the course content but can refer to the courses from which the IBM Skills Build badges are obtained.

This document is the Requirement Specification for our RPG IBM Skills Build project. There are three parts to this document. The first part is the introduction, which gives a high-level description of the project, its scope, and domain area. The second part, the Solution Requirements, is the main part of the requirement specification. This part specifies the projects' functional and non-functional requirements and assess the risks/issues which may occur when developing the project. The third part describes our approach to the development of the game, as well as our schedule to complete the project within the agreed deadline.

## 1.2 - Project Scope

The IBM Skills Build website is designed to be an educational website with courses on various subjects related to Computer Science. However, the current website is sometimes confusing, complex, and can be a barrier to entry for students or people who are interested in computer science. This is the problem we are aiming to solve with our project.

The purpose of the software is to provide a fun way to navigate the website and discover IBM Skills Build courses. This will be done by making a top down 2D RPG game where the courses are linked throughout the game, and where you need to learn about the topics covered to progress through the game. This would help solve the problem by making the courses more accessible and encourage the users to do the courses and get the badges.

This aligns with the interest of our client and main stakeholder as they would like to remove the unnecessary barrier to entry. Whilst we cannot remove the barrier to entry completely, by having a game which links to the resources, this barrier can be lowered and make the courses more accessible.

The stakeholders for our project are our users and IBM (John McNamara). Our users will mainly consist of university students and secondary school students who are interested in Computer Science. This may include people who have never played video games before, hence our game should be as accessible as possible.

In the future, we might want to expand upon the game. These are some features that could be added in future versions:

- A leaderboard ranked by completion time

- Cooperative multiplayer

- Player vs Player mode

## 1.3 - System Description

### Our proposed System

Our proposed solution to the IBM's Skills Build website issue is the creation of an educational 2D RPG game, featuring questions from the website. These questions will be at the forefront of the game and will allow the player to progress when answered correctly.

Each Skills Build course will be represented by an area in the game and each area will feature puzzles and a boss fight, which will pose questions to the player.

Combat will introduce a further element of risk, in the form of a health system which deducts a heart from a player upon answering a question incorrectly. Loss of all hearts results in the player restarting the entire fight.

A skill tree will enable the player to alter combat to suit their play style, by choosing powerups after defeating each boss.

All areas will be linked by a hub world, allowing the player to select an area to enter in any order they please.

All of our design choice for the game were based on market research found here.

### Existing Solutions

**Mario is Missing (by Nintendo)** [1]

**Description:** Mario is Missing challenges the player to save global landmarks stolen by Bowser. By learning about geography from non-playable characters (NPC) the player can identify and return the landmarks to their correct geographical location.

**Advantage:** The game acts as an educational tool by improving the players geographical knowledge through exploration of locations.

**Disadvantage:** As the game's target audience is children, it is extremely easy. Furthermore, the answering of questions during play is not mandatory and so the educational aspect of the game can be avoided.

**Use for our System:** Negative reviews suggest the gameplay cycle of finding landmarks to be tedious and the questions unnecessary. Our game should contain essential questions with other gameplay elements to engage players.

**Minecraft Education (by Microsoft)** [2]

**Description:** Minecraft Education enables students to engage with lessons created by teachers in Minecraft which focuses on academic subjects.

**Advantage:** The game facilitates learning through fun interactive classes. Learning is at the forefront and knowledge gained can be directly applied to real life.

**Disadvantage:** Players familiar with Minecraft will feel frustrated, as many core gameplay features are limited to facilitate learning.

**Use for our System:** Players of our game who know the answers to all questions should be able to complete the game more quickly, but the game should still have interactive puzzles to engage players.

**Mario and Sonic at the Olympic Winter Games 2010 DS (by Nintendo and Sega)** [3]

**Description:** Bowser and Dr. Eggman have imprisoned the Snow Spirits to sabotage the Olympic Winter Games and Mario and Sonic must save them. The educational aspect of the game is delivered through Winter Olympic themed trivia scattered around the game world.

**Advantage:** The trivia notes, which are location specific, are hidden around the world and so well integrated into the game. The player is rewarded for finding the information, with completionist badges.

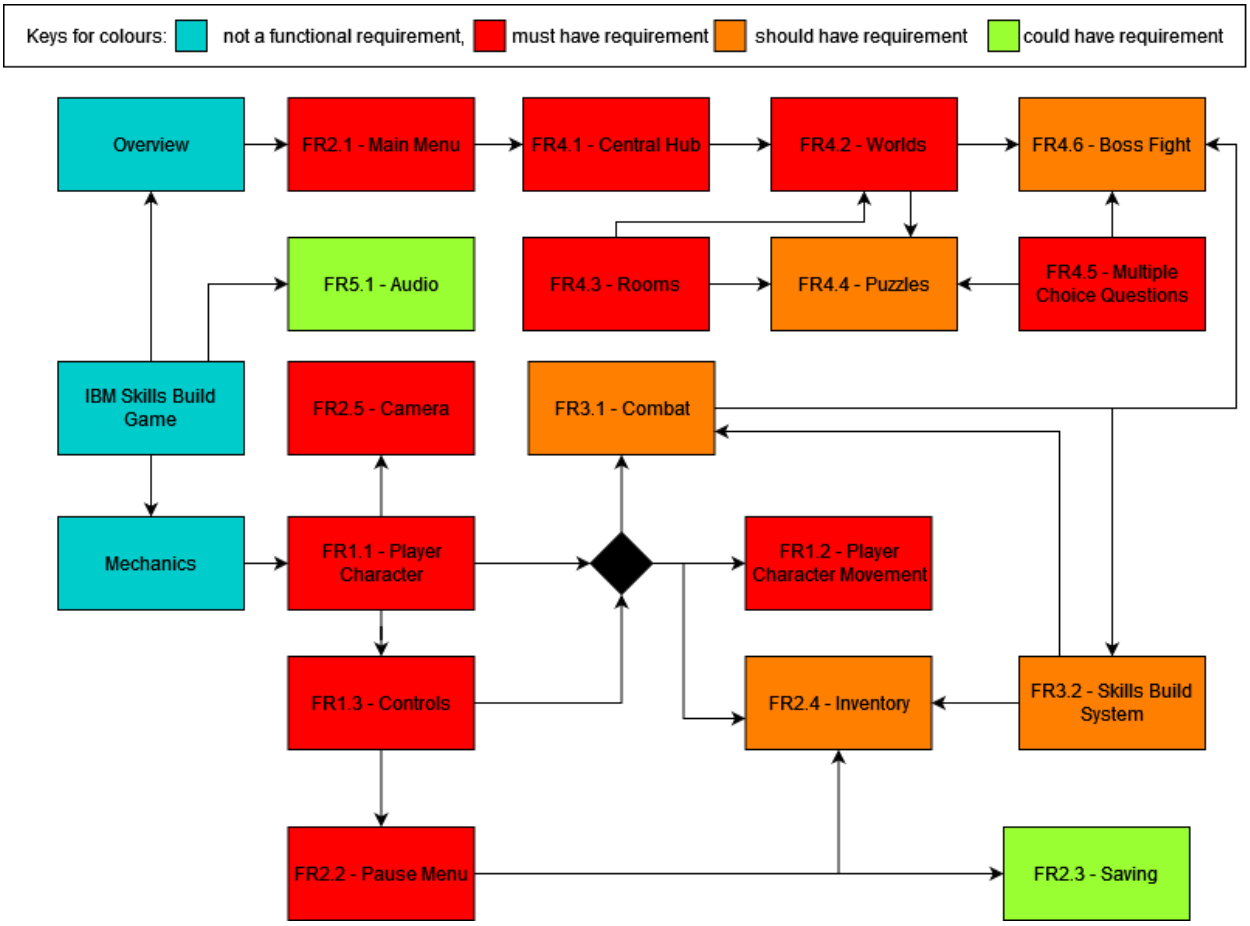
**Disadvantage:** The game is not focused on these trivia notes, rather they are an additional feature, which despite it giving the player a badge, does not facilitate progression: They are a side quest.

**Use for our System:** Our game should reward the player for answering questions correctly and enable progression. Without correctly answering the questions, the player should not be able to progress. The questions asked should fit the environment of the game.

# 2 - Solution Requirements

## 2.1 - Function Requirements

Below is a dependency graph for our functional requirements: Functional Requirement Dependency Graph




### Player-Based system

| ID, Type, Title | FR1.1 - Player-Based system - Player Character |
|---|---|
| Description | Main playable character which the game centers around, should be a student (user) stand-in |
| Priority / MuShCo | High / Must have |
| Dependencies | N/A |
| Expected results | Character should have distinct/unique sprite both in and out of battle (small zoomed out model outside of battle, portrait inside) |
| Exception handling | If the character sprite cannot be loaded, it'll be a placeholder inside |

| ID, Type, Title | FR1.2 - Player-Based system - Player Character Movement |
|---|---|
| Description | Buttons which allow the player character to move in any direction (even though the stages are built within a grid) but does not allow the player do go through walls, closed doors, or occupied tiles |
| Priority / MuShCo | High / Must have |
| Dependencies | FR1.1 |
| Expected results | Player character should be able to move in the four cardinal directions and diagonally with either WASD (holding a combination of two keys, say W and A, would make the player move diagonally up and left) or via controller |
| Exception handling | If the player gets stuck on a tile, they should be able to wiggle out and free themselves either by mashing all directions or holding one direction |

| ID, Type, Title | FR1.3 - Player-Based system - Controls |
|---|---|
| Description | Buttons which allow the player to interact with both the game entities (items/characters) and the menu |
| Priority / MuShCo | High / Must Have |
| Dependencies | FR1.1 |
| Expected results | There should be at least two buttons like that of the old Gameboy. [A] generally for accept, and [B] generally for cancel. If a player character is between two game objects, the one they are closer too should be the one being interacted with |
| Exception handling | N/A |

## Game System

| ID, Type, Title | FR2.1 - Game System - Main Menu |
|---|---|
| Description | A menu which is opened at after launching the game |
| Priority / MuShCo | High / Must have |
| Dependencies | N/A |
| Expected results | The menu should have a title, a background, and buttons. Contains buttons to: Start a new game, continue an ongoing session, open the options, credits, and exit the game |
| Exception handling | N/A |

| ID, Type, Title | FR2.2 - Game System - Pause Menu |
|---|---|
| Description | A menu which pauses the game when opened, with options available |

| | |
|---|---|
| | such as quitting |
| Priority / MuShCo | High / Must have |
| Dependencies | FR1.3 |
| Expected results | After pressing the [Esc] button, a menu should open. When this menu is open, all entities and events should be paused. This menu contains buttons to: Access the inventory, save the game, open the options menu, or quit (to the main menu) |
| Exception handling | If opened during a timed event, the menu should not open |

| **ID, Type, Title** | **FR2.3 - Game System - Saving** |
|---|---|
| Description | A button to save the player's progress at any point in the game and saves should automatically be done throughout the game |
| Priority / MuShCo | Low / Could have |
| Dependencies | FR2.2 |
| Expected results | In the case that the user wants to take a break, or wants to go on Skills Build to learn about the topic, they should be able to save their progress, or in case the game crashes and be able to load back their save file |
| Exception handling | If the save failed, the user should be notified and asked if they want to save again |

| **ID, Type, Title** | **FR2.4 - Game System - Inventory** |
|---|---|
| Description | A menu where the player can see all their currently collected items |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR1.1, FR2.2, FR3.2 |
| Expected results | A menu that shows the user their current items and their descriptions. This menu can also display other items the user can get but shade them out if the user has not gotten them yet. The user cannot read the descriptions of shaded items |
| Exception handling | N/A |

| ID, Type, Title | FR2.5 - Game System - Camera |
|---|---|
| Description | A camera that can track the player character and can be moved around the room |
| Priority / MuShCo | High / Must have |
| Dependencies | FR1.1 |
| Expected results | This is the main tool to display the game. Most of the time, the camera will be centered on the player character, but it can be used to pan around a room |
| Exception handling | In the rare case that the camera is not centered around the player character, there should be a button (in the pause menu) to center the camera. Only implement if the camera gets stuck frequently |

## Gameplay Systems

| ID, Type, Title | FR3.1 - Gameplay Systems - Combat |
|---|---|
| Description | A combat system, where the player has a health bar and takes damage when attacked by enemies |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR1.1, FR1.3, FR3.2 (Co-dependency) |
| Expected results | A (decorated) bar which depicts the health of an entity. Once this reaches zero, the entity should die. It should show the current health, and the maximum health, and the player should take damage when attacked |
| Exception handling | If the player character somehow stays alive with negative health, they should lose at the start of their next action |

| ID, Type, Title | FR3.2 - Gameplay Systems - Skills Build System |
|---|---|
| Description | A system which permanently upgrades the player character as the game progresses. Can be accessed through a menu |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR3.1 (Co-dependency) |
| Expected results | A skill tree which has nodes that provides various benefits, like giving the player character more health or attack. After finishing a world, the skill associated with that world unlocks. Nodes can be activated by spending skill points/ or acquiring unique items |
| Exception handling | If the player somehow has negative skill points, they should not be able to quit the Skills Build menu |

## Gameplay

| ID, Type, Title | FR4.1 - Gameplay - Central Hub |
|---|---|
| Description | A central hub world where the user can access other worlds |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR2.1 |
| Expected results | The user can, from the hub world, access all the main worlds in any order. This place will be more fleshed out compared to other places since the user will return here a lot |
| Exception handling | N/A |

| ID, Type, Title | FR4.2 - Gameplay - Worlds |
|---|---|
| Description | The different realms where the game takes place. Each world contains many rooms and ends with a boss fight |
| Priority / MuShCo | High / Must have |
| Dependencies | FR4.1, FR4.3 |
| Expected results | At least five main worlds, each with their own puzzle style accessible through the central hub. Each world represents a Skills Build course |
| Exception handling | N/A |

| ID, Type, Title | FR4.3 - Gameplay - Rooms |
|---|---|
| Description | A room that contains a puzzle or a boss, with a door to the next room |
| Priority / MuShCo | High / Must have |
| Dependencies | N/A |
| Expected results | A room has at least one entrance and can have none or multiple exits. The room size can be from one screen to multiple screens. After entering through a door, the player character is placed right outside the door of another room, and the door to the next room should only be opened once the puzzle is solved or the boss is defeated |
| Exception handling | N/A |

| ID, Type, Title | FR4.4 - Gameplay - Puzzles |
|---|---|
| Description | Themed minigames |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR4.2, FR4.3, FR4.5 |
| Expected results | Puzzles have a theme in each world. Most puzzles will be a multiple-choice question disguised as a minigame |

| | |
|---|---|
| Exception handling | Some puzzles will require a reset button, which restores the state of the room it's in to how it originally was |

| **ID, Type, Title** | **FR4.5 - Gameplay - Multiple Choice Question** |
|---|---|
| Description | Questions from a Skills Build course which the user must answer |
| Priority / MuShCo | High / Must have |
| Dependencies | N/A |
| Expected results | Questions are taken from the Skills Build quizzes. They are all multiple choices |
| Exception handling | N/A |

| **ID, Type, Title** | **FR4.6 - Gameplay - Boss Fight** |
|---|---|
| Description | Fights where the user must correctly answer questions from the current world where the boss is in |
| Priority / MuShCo | Medium / Should have |
| Dependencies | FR3.1, FR4.2, FR4.5 |
| Expected results | Both the player character and the boss have a health bar. If the user answers a question correctly, they get to attack the boss. If they don't answer correctly, they get hit by the boss. Bosses can ask any questions from the previous puzzles in their respective world. The final boss can ask questions from all worlds |
| Exception handling | If the fight lasts long enough, there will not be enough unique questions to ask the user. To prevent this happening, questions can be reused after all other questions gets exhausted |

### Audio

| **ID, Type, Title** | **FR5.1 - Audio – SFX and BGM** |
|---|---|
| Description | Sound effects for some actions, such as room transitions and background music for the game and combat music for fights |
| Priority / MuShCo | Low / Could have |
| Dependencies | N/A |
| Expected results | Sounds should be made for most of the common actions, like opening menu, attack an enemy, or walking through a room. Music should loop indefinitely |
| Exception handling | N/A |

## 2.2 - Non-Functional Requirements

### Playability

| **ID and Title** | **NFR1.1 - Executable file** |
|---|---|
| Type | Usability |
| Metric | The game can be launched by running a single executable file |
| Constraint | The user must be on windows OS |

| **ID and Title** | **NFR1.2 - Platforms** |
|---|---|
| Type | Usability |
| Metric | Game should be playable on mouse & keyboard as well as controller |
| Constraints | Not all controllers need to be supported, just common ones e.g., Xbox and PlayStation controllers |

### Customization

| **ID and Title** | **NFR2.1 – Graphics** |
|---|---|
| Type | Usability |
| Metric | Graphics abide by the Game Accessibility Guidelines, so the game is enjoyable for users suffering from colour blindness |

### Gameplay

In this section, any NFR metrics which is subjective in nature (NFR3.2 – 3.7) will be judged a through user feedback questionnaire in the testing phase.

| **ID and Title** | **NFR3.1 - Response Time** |
|---|---|
| Type | Performance |
| Metrics | There should be less than 0.1 seconds of delay between pressing the movement/interacting keys and the player moving/interacting |
| Constraints | User has a good machine with more than 4GB of RAM and running windows 10 and above |

| **ID and Title** | **NFR 3.2 - Smooth Movement** |
|---|---|
| Type | User Experience |
| Metrics | Movement speed and animation should feel natural |

| ID and Title | NFR 3.3 - Beginner Friendly |
|---|---|
| Type | Usability |
| Metric | Game and UI should be intuitive to use even for people who have not played games |

| ID and Title | NFR 3.4 - Room Transitions |
|---|---|
| Type | Performance |
| Metric | Transition between scenes should be smooth - there should be no more than a 3 second delay when moving to another room |
| Constraint | There should still be some fading effect |

| ID and Title | NFR 3.5 - Boss Battles |
|---|---|
| Type | User Experience |
| Metric | Battle should feel engaging |
| Constraint | User must learn the relevant topics through IBM Skills Build beforehand |

| ID and Title | NFR 3.6 - Music |
|---|---|
| Type | User Experience |
| Metric | Music should fit each area |

| ID and Title | NFR 3.7 - Progression |
|---|---|
| Type | User Experience |
| Metric | The game must have an aspect of progression where items are gained which benefits them |

## 2.3 - Risks and Issues

This section outlines our potential risks, how they could harm out project, and how we plan to mitigate these risks.

| Probability of Happening / Potential Consequences | Almost Impossible (1) | Not Likely (2) | Could Happen (3) | Known to Happen (4) |
|---|---|---|---|---|
| Insignificant (1) | 1 | 2 | 3 | 4 |
| Minor (2) | 2 | 4 | 6 (R4) | 8 |
| Moderate (3) | 3 | 6 (R7) | 9 (R2, R6) | 12 (R5) |
| Major (4) | 4 | 8 (R8) | 12 (R1) | 16 (R3) |

| **Hazard** | What is at Risk? | How could they be harmful? | Uncontrolled Risk Level | How to minimize the risks | Controlled Risk Level |
|---|---|---|---|---|---|
| **R1 - Not Being Completed on Time** | Project timeline & client satisfaction | Delays result in missing milestones, which will disappoint the client | 12 | Implement a well-defined schedule and monitor progress to deal with issues | 6 |
| **R2 - Group Dynamic** | Team cohesion & project quality | Arguments lead to lower quality product, and delays | 9 | Establish clear roles and responsibilities, address issues promptly | 4 |
| **R3 - Lack of Group's Technical Knowledge** | Project quality and timeline | Inability to meet project requirements | 16 | Identify skill gaps and train to fill gaps | 9 |
| **R4 - Scope Creep (uncontrolled expansion of project)** | Project timeline | Project not completed on time due to base features not being completed | 6 | Define project scope clearly, use agile methodology to add new features after base completed | 4 |
| **R5 - Team Member Drops Out or is Unavailable** | Group members & project timeline | Other group members have more work, delays if key member | 12 | Cross train team members, or have backup plans | 6 |
| **R6 - Poor User Feedback** | User acceptance & client happiness | Fail to create a positive user experience | 9 | Conduct occasional user testing and gather feedback | 4 |

| | | | | | |
|---|---|---|---|---|---|
| **R7 - Requirements Change** | Project scope & timeline | Will have to go back and change content, potentially from a fundamental level | 16 | Document and manage project requirements, make the game flexibly to adapt to change | 8 |
| **R8 - Hardware Compatibility** | Game performance or accessibility | Poor performance or being unable to play on systems | 6 | Test on several platforms and hardware configurations | 2 |

# 3 - Project Development

## 3.1 - Development Approach

### Team Roles

Before tackling the project, we decided to do a Belbin-like test, called the Plum Test [4], and assign roles for our team based on this.

We have two members whose top talent is innovation, hence we assigned them more creative roles.

- Innovation, Execution: **Lead Game Designer**
- Innovation, Adaptation: **Creativity Director**

We also have two other members who have varied top talents.

- Communication, Teamwork: **Communication and Management Specialist**
- Execution, Decision-Making: **Technical Lead**

The last member of the team could not take the Plum Test, and hence was assigned to the role of **Developer**.

In the end, this approach is best for our team due to our existing skill set and allows individual member's talents to shine.

### Agile Versus Plan Driven

We will be taking an agile Scrum approach to development, as it helps us in several ways:

1. **Adaptability to Change**

   Features and priorities will greatly change as development progresses, so with Scrum we can rapidly change aspects of the project without losing progress.

   A waterfall approach to development would not allow this kind of adaptability.

2. **Client Involvement**

   Clients require regular updates from us, so we need a method that involves them in the development process. The agile methodology allows for this as the client can serve as the product owner role.

   The lean innovation model is not ideal as it solely focuses on completion of the project, whereas we want to have a good relationship with our client and focus on their needs. [5]

3. **Cross-Functional Team**

   Everybody in our team has a different level of knowledge when it comes to game development. By using Scrum, team members can share, receive knowledge, and collaborate easier through the Scrum master and daily meetings.

   A plan driven approach will cause major issues if a member ends up falling behind on their tasks unbeknownst to the team.

4. **Improved Risk Management**

   If something doesn't go as planned, a Scrum style approach will let us identify, and mitigate any risks that arise within our sprint reviews.

   In a plan driven approach, if any issues were discovered late, it could cause problems to stack up and thus cause progress stagnation.

## Workflow

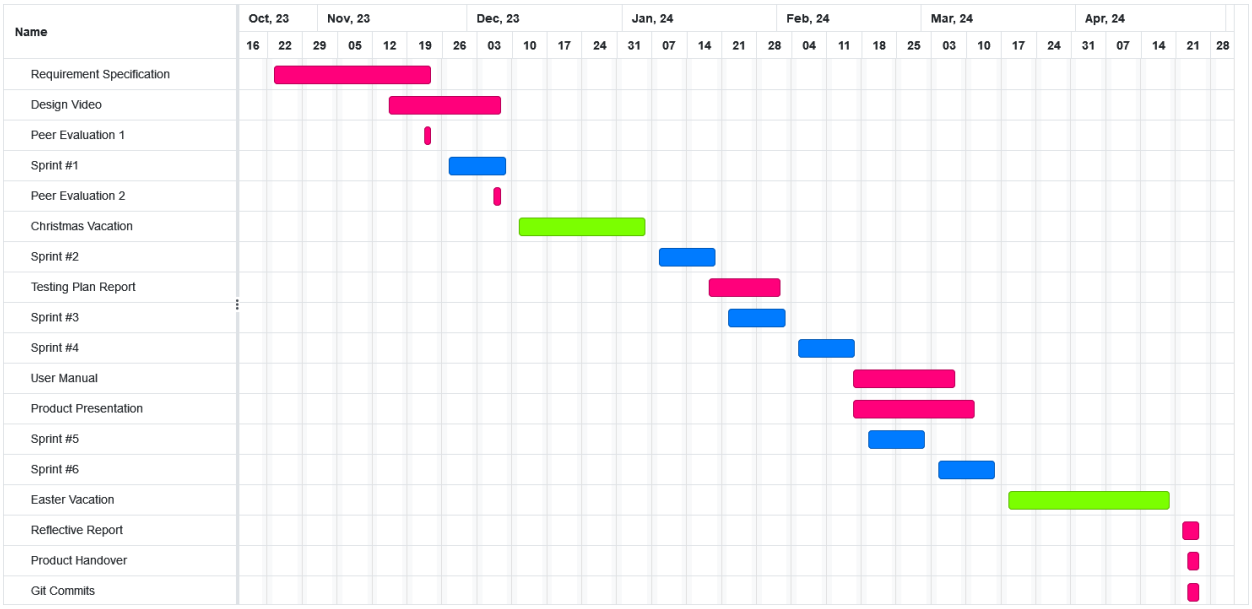
Before starting sprint cycles, we will need to write up our product backlog, as well as assign points for each task. The way points are decided will be based around a modified version of the point-to-effort estimation, where we also take into consideration the priority of each task (tasks which take less effort but are more important will weigh more points). The tasks can then be further broken down in the sprint cycles.

As we do not have easy and frequent contact with our product owner (John McNamara), our Communication and Management Specialist will act as a stand-in representative for this role.

For each sprint cycle, we will have a new Scrum master, so that everyone can have a grasp of how the role operates. Each cycle lasts two weeks, and we aim to have daily meetings for at least weekdays to discuss progress and problems, ending in an end-of-cycle review. We start our sprints on Mondays, and end on Fridays to allow for a weekend of rest.

## 3.2 - Project Schedule

The Gantt Chart shown below outlines the development time for our project and highlights the academic deadlines too.




The chart considers the number of members and the weekly workload we have; we have 5 members, and on average work on Software Engineering for 4h per week plus any extra time voluntarily invested. Also, we have made sure to keep holidays free of any work on this project.

Since our chosen software development cycle methodology is Scrum, we do not have specific plans for when each part of the project should be done. However, we still have outlined some self-imposed deadlines along with the academic deadlines which will guide our Scrum sprints. We have chosen to have each sprint be 2 weeks long.

### Self-imposed deadlines

- Finish initial game setup - 8th December 2023 (Post sprint #1)
- Game ready for testing - 28th February 2024 (Post sprint #4)
- Finish Art - 10th March 2024 (Post sprint #6)

## References

[1] DeadPark. (2016, April 3). Mario is Missing! (SNES). https://www.deadpark.com/30-minute-reviews/mario-is-missing-snes/

[2] Microsoft. (2023). Minecraft Official Site | Minecraft Education Edition. Education.minecraft.net. https://education.minecraft.net/en-us

[3] Super Mario Wiki. (2023, October 17). Mario & Sonic at the Olympic Winter Games (Nintendo DS) - Super Mario Wiki, the Mario encyclopedia. https://www.mariowiki.com/Mario_%26_Sonic_at_the_Olympic_Winter_Games_(Nintendo_DS)

[4] Plum. (Retrieved 2023, November 20). What are the 10 Plum Talents? [Review of What are the 10 Plum Talents?]. https://help.plum.io/hc/en-us/articles/360002388234-What-are-the-10-Plum-Talents-

[5] Lean Business Model Pros and Cons: Can You Afford Frugality? (n.d.). National Center for the Middle Market. https://www.middlemarketcenter.org/expert-perspectives/lean-business-model-pros-and-cons-can-you-afford-frugality