Requirements Document Rev 0
A World Apart

Team SantaHatesPoorKids
Jim Wu, 001409055
Ian Yang, 001217664
Gabriel Castagner, 001412885
Junhao Wang, 001215428

March 3, 2018

Supervised by:
Dr. Jacques Carette
Software Engineering 4GP6

Table 1: **Revision History**

Date	**Version**	**Notes**
Oct 15	1.0	Create Document
Oct 16	1.1	Finalized first Draft.
November 1	1.2	Fixing TA Feedback part1
November 5	1.3	Added Requirements for V&V
December 3	1.4	Moved some save/load requirements to working room for Design Doc
February 10	2.0	Version 2 of Requirement Document, Feedback highlighted

Contents

1 Purpose of the Project

1.1 Background

This game is to being developed as the main project for the SFWR ENG 4GP6 capstone course. This game had been an original concept that never made it out of it's idea developing stage and the development team decided that the game's idea and aesthetic would be a great project to make.

1.2 Goals

The development team wants to create a enjoyable strategic rogue based game as a personal achievement. This game has a few aspects that have open ended solutions such as map generation and random chance variables that will give us the knowledge of how to make them and improve upon them in the future. We as a development team want to develop this project so that we can improve our abilities to become excellent game developers and understand the process of creating a video game using the Unity Engine. The Development team wants our audience to have an enjoyable, relaxed experience while playing this game. Rogue-likes can usually be very stressful and frustrating due to its random and trial and error nature, but we wish to change this experience and base it more on long term thinking and reward thoughtful resource management.

2 Stakeholders

There are a few stakeholders in for this project.

Player - The Player stake in the game is that they are the ones who are playing and enjoying the game. They are important to take into account as we will need to ask questions like, "Is this portion of the game fun" or "Does this feature benefit the player's understanding of a mechanic". The main concern for the player will be to ensure that the game is fun and understandable.

Development Team - The Development Team's stake in the game is that we are the ones who are creating and organizing the game. This project will reflect our knowledge that we have acquired over the course of our university career so we must show that we can develop this game properly and so that it can be improved upon in the future.

Instructor - The instructor's stake is that they will have constructive control over the duration of the project. They will provide feedback and give guidance so that the development team can create the best version of their game.

Course TA - The TA for the course, Dan, has a stake in the project as he is the one is assessing and providing feedback to both the project and the corresponding documents so that the project is finished professionally and efficiently. Judges - The judges stake is that they will give a review based on merit and specific guidelines. They decide how good the game is and how well it fits the development criteria.

3 Mandated Constraints

There are a few constraints that the instructor wants us to follow.

MC1. The project must be developed in the Unity Game Engine

MC2. The project's documents and source files must be managed in the respective GitLab files

MC3. The project must be a fully fledged, standalone video game that must be created from scratch with the exception of game assets (models, sprites, audio, etc).

3.1 Solution Constraints

There are a few constraints that the development team can use as solutions wants us to follow.

SC1. The project must be developed in the Unity Game Engine

SC2. Any game assets in the Unity assets store must be referenced in both the project documents and in the game folders.

SC3. Any game assets outside of the Unity assets store and not original to the development team must have documented consent to use such game assets if not under free use terms and conditions.

3.2 Implementation Environment of the Current System

There are a few environment implementation constraints.

IE1. The game will be published to be played on Windows, IOS and Linux Machines.

IE2. The game will be able to run on a player's laptop or desktop machine.

IE3. The game will not need the internet to functions.

3.3 Partner or Collaborative Applications

N/A

3.4 Schedule Constraints

There are a few Schedule constraints.

SC1. The project's Sales pitch and first demo will need to be presented October 17th 2017

SC2. The Game requirement's document will need to be ready by October 19th 2017

SC3. The first Implementation document will need to be ready by December 7th 2017

SC4. The first V and V document will need to be ready by January 4th 2018

SC5. The last Game requirement's document will need to be ready by February 27th 2018

SC6. The last Implementation document will need to be ready by March 29th 2018

SC7. The last V and V document will need to be ready by April 6th 2018

SC8. The final demo will need to be presented sometime in April 2018

3.5 Budget Constraints

There are a few Budget constraints.

BC1. The development team will allocate $10.00 for any necessary game assets.

BC2. The development team members will be responsible for purchasing any 3rd party software needed to develop game assets.

4 Naming Conventions and Terminology

The following are name conventions that the development team will use:

Project - Interchangeable with Game and Application.

Game Assets - Used to reference the aesthetics of the game. These include character models, sprites, textures, character specific scripts, and audio.

Game Developers - References the four group members responsible for creating the game.

Submenu - References any menu that can be accessed from the Main or Game menu, usually referring to save, load, new game menus and options menus.

5 Relevant Facts and Assumptions

5.1 Relevant Facts

There are a few facts about game environment the developers made.

RF1. Maps are generated using a pseudo randomly generated seed. It is theoretically possible to generate the same map given the same seed.

5.2 Assumptions

There are a few assumptions the developers made about the players.

A1. We assume that all users of this product are capable of using a computer and a keyboard.

A2. We assume that all users of this product will be able to differentiate between their character and the map.

A3. We assume that all users will have an Apple or Windows machine.

A4. We assume that all users have used a WASD + spacebar control scheme before.

A5. We assume that some users have played a some sort of game that uses turn based rules.

A6. We assume that when the user achieves a game over that that specific game is no longer re playable.

6 Scope of the Work

6.1 Existing Inspirations

There are a few existing game that was are using as inspiration to model our game's mechanics and theme off of.

EI1. FTL- We are using FTL's map generation as a general concept on how the level will be connected together.

EI2. Dustforce - We are using Dustforce's art style as a general concept to keep things simple and clean yet looking good.

EI3. Banner Sauga - We are using Banner Sauga's Health and Armour system to help improve our own combat system.

6.2 Context of the Work

There are a motivational factors that the development team will keep in mind when we are developing the project.

CW1. The development team will challenge each feature with the idea of fairness, is the feature implemented fair to the player. The team will as to see if the feature can be accounted for and deflected. If the feature can't be accounted for, will it terminally upset the player causing game breaking moments.

CW2. The development team will ensure that the art style, gui and map design is considered clean and clear. The developers will question if features like tool tips, indications and menus are easy to read and that selections are simple for the user to understand. There should be limit of the clutter on screen, both entity and texture wise.

7 The Scope of the Product

7.1 Product Boundary

This section will cover what the project will include and what it will not include.

PB1. The development team will challenge each feature with the idea of fairness, is the feature implemented fair to the player. The team will as to see if the feature can be accounted for and deflected. If the feature can't be accounted for, will it terminally upset the player causing game breaking moments.

PB2. The development team will ensure that the art style, gui and map design is considered clean and clear. The developers will question if features like tool tips, indications and menus are easy to read and that selections are simple for the user to understand. There should be limit of the clutter on screen, both entity and texture wise.

7.2 Product Use Case Table

PUC	PUC Name	Actors	Input/Output
1	Load Game	System	Load Game button initialization, Save game file, Re-initialization of Previous Game
2	New Game	System	New Game button initialization, Initialization of New Game
3	Display Tool Tip Info	System	Mouse x and y coordinates, GUI display of Tool Tip Info
4	Perform Character Action	System	Player Input, Appropriate System output, GUI output
5	Perform Character Modification	System	Player Input, Appropriate System modification, GUI output
6	Audio Modification	System	Player Input, Appropriate Audio modification
7	GUI Modification	System	Player Input, Appropriate GUI modification
8	Merchant Interaction	System	Player Input, Appropriate GUI modification, Resource update modification
9	Overhead world movement	System	Player Input, Appropriate GUI modification, Map update modification

7.3 Individual Product Use Cases

This Section will cover the Product Use Cases in detail. They will include their trigger conditions, preconditions, procedures, outcomes and any use cases that are related.

PUC1. **Load Game -** The user wants to load a desired game session.

Trigger Conditions: Player Selects Load Button

Preconditions Conditions: Game Session is paused, Player is in Game Menus, Player is in Main Menu

Procedures: System asks if they wish to leave their current game session. If selected yes then system reads desired game file and re initializes desired game data.

Outcomes: If player selected no to leaving current game session, then the system returns the player to the game menus. Otherwise, the desired game is displayed on screen in the same state that it had been saved in.

Related Use Cases: N/A

PUC2. **New Game -** The user wants start a new game session.

Trigger Conditions: Player Selects New Game Button

Preconditions Conditions: Player is in Main Menu

Procedures: System asks to select difficulty, name the game type and waits for player input to start game. The player selects start game, the base save file is created with the corresponding difficulty and seed. Game file is generated using the corresponding seed.

Outcomes: New game is initialized, new game save file is created.

Related Use Cases: N/A

PUC3. **Display Tool Tip Info -** The user wants to view specific character values and how they are derived.

Trigger Conditions: Player has mouse icon hover over the GUI of a value for 0.5 seconds.

Preconditions Conditions: The player is viewing a menu where that value is displayed.

Procedures: The System retrieves the desired value's equation and formats the dependencies in a listed format similar to:
value - dependency_name.

Outcomes: Returns a GUI format of the values dependencies and names.

Related Use Cases: N/A

PUC4. **Perform Character Action -** The user wants to perform a specific action related to a character.

Trigger Conditions: Player right clicks on any object in the scope of possible options.

Preconditions Conditions: The player has selected a character that is able to perform an action.

Procedures: The System determines if the selected option is possible, if not, then the system will provide and audio feedback that indicates that is not a possible action to the player. If it is a possible action, then the system will perform that function by updating the values of the character(s) and entities involved.

Outcomes: Returns the Graphical representation of outcome of the actions and updates the current game session.

Related Use Cases: N/A

PUC5. **Perform Character Modification -** The user wants to apply a specific modification to a character.

Trigger Conditions: Player matches a modification with a desired character.

Preconditions Conditions: The player is in a character editing menu.

Procedures: The System determines if the selected option is possible, if not, then the system will provide and audio feedback that indicates that is not a possible action to the player. If it is a possible action, then the system will perform that function by updating the values of the character.

Outcomes: Returns the Graphical representation of outcome of the modification and updates the current game session.

Related Use Cases: N/A

PUC6. **Audio Modification -** The user wants to apply a specific modification to the audio of the game.

Trigger Conditions: Player makes adjustment to an audio feature.

Preconditions Conditions: The player is in the audio menus.

Procedures: The System applies to the change to the game session base on the value corresponding to the menu.

Outcomes: Returns the Graphical representation of outcome of the modification and returns a audible que to the player representing the selection that they have changed.

Related Use Cases: N/A

PUC7. **GUI Modification -** The user wants to apply a specific modification to the graphics of the game.

Trigger Conditions: Player makes adjustment to a graphical feature.

Preconditions Conditions: The player is in the graphics menus.

Procedures: The System applies to the change to the game session base on the value corresponding to the menu. The system may need to readjust it's output values depending on the input selection.

Outcomes: Returns the Graphical representation of outcome of the modification and returns a graphical indication to the player representing that the change has been made.

Related Use Cases: N/A

PUC8. **Default Setting Reset -** The user wants to revert the settings to their default settings.

Trigger Conditions: Player presses the reset to default button.

Preconditions Conditions: The player is in the graphics or audio menus.

Procedures: The system poses to the player if they want to reset to default options. If they say no, then they are returned to respected menu. If the player says yes, then the system resets the values to the default settings.

Outcomes: Returns the Graphical representation of values being reset to default.

Related Use Cases: N/A

PUC9. **Merchant Interaction -** The user wants to buy or sell things from the merchant.

Trigger Conditions: Player presses the buy or sell button

Preconditions Conditions: The player is in the merchant menu. The user has the appropriate resources to trade for the desired item.

Procedures: The system removes the required resources from the player's resource pile and adds the desired item to the player's equipment list or resources.

Outcomes: Returns the Graphical representation of values being changed. The change is applied to the character's resources.

Related Use Cases: N/A

PUC10. **Overhead World Movement -** The user wants to move the character party to an adjacent node.

Trigger Conditions: Player presses selection on a adjacent node.

Preconditions Conditions: The player is in the overhead world view. The player has enough resources to proceed.

Procedures: The system graphically moves the characters to the desired node.

Outcomes: The system shows initializes the new node phase initiation. The party position is updated in the corresponding file.

Related Use Cases: N/A

8 Functional Requirements

This section will cover functional requirements that do not include main game functionality. It will primarily consist of back end workings.

FR1. If the player selects the exe file, the game is initialized from its previous build.

FR2. If this is the player's first time running the game, the save folder is built to store save files.

FR3. If the player its the close window button, the game terminates.

8.1 Core Mechanics

CMR1. The system generates a map that has unique regions, each with their respective node maps using the generated file seed.

CMR2. Each Region has more than 10 nodes, each node being connected to at least 4 other node and no more than 8 nodes.

CMR3. Each node has a cost of supplies in that the player needs to get to the specific node.

CMR4. Each node has the possibility of have positive, negative or neutral outcomes of acquire resources phase.

CMR5. Each node should not deadlock the character with a negative outcome that makes them lose the game.

CMR6. Each node has a cost of supplies in that the player needs to get to the specific node.

CMR7. Each character is given a class type and base stats values.

CMR8. During the combat phase, the individual character and enemy turns are listed at the bottom of the screen.

CMR9. The character and enemy turns orders are randomly mixed up and are kept in the same order for the duration of the combat phase.

CMR10. If an enemy or character dies, they're turns are removed from the combat phase.

CMR11. Characters and Enemies have default health and Armour.

CMR12. The player and the AI can decide either to attack the health values or the amour values of their respected enemies.

CMR13. When the Armour value reaches 0, that selection cannot be made.

CMR14. The Armour value must provide a reduced damage value that is of some sort of inverse exponential or logarithmic value to promote the intensive to remove amour.

CMR15. Each statistical value has a tool tip area that will display a tool tip menu when the player has hovered over the value for a given period of time.

CMR16. The tool tip displays how the value is derived.

CMR17. The tool tip will show different ways on how the value is changed.

CMR18. The tool tip will always fit on screen, never trailing off screen.

CMR19. The tool tip disappears if the mouse is hovered away.

8.2 Primary Game play Mode

This section will cover requirements that are related to the primary game play of the project, which is the fighting portion of the game. The scope of the fighting game play will include it's setup, duration and outcome of combat. The requirements of the fighting game play is:

PGR1. The player is always has the first turn.

PGR2. The combat phase's turns happen on individual character basis.

PGR3. The player's or AI's turn shall not be interrupted

PGR4. The player wins if all objectives are completed.

PGR5. The player loses if all character die or an objective is failed.

PGR6. If the player wins, then the system updates the file with the success while having a GUI output to player the outcome of the fight. The player is then returned to the Overhead map view.

PGR7. If the player loses, then the player is shown a closing dialogue sequence and is returned to the main menu. The file is deemed unplayable.

PGR8. While a character is performing an action or the AI's turn is being conducted, the player shall be able to enter the menus, pausing the game.

PGR9. If the player is in the menus while a character is performing an action or the AI's turn is being conducted, the player is not able to save or load a game.

PGR10. The Player may not be able to interrupt the AI's turn while the AI is still performing it's turn.

PGR11. When it is the player's turn, the player shall be able to move the chosen character.

PGR12. When it is the player's movement phase, the player will not be able to move the chosen character on an obstacle on the map.

PGR13. When the player has moved the chosen character, they maybe be able to select one of the four actions.

PGR14. When the player is in the action phase of a chosen character, they will not be able to choose an actions that does not meet the subsequent criteria.

PGR15. When it is the AI's turn, the AI shall be able to select a multitude of actions to perform to best situate itself given the subsequent criteria.

PGR16. When it is the AI's movement phase, the AI will not be able to move the chosen character on an obstacle on the map.

PGR17. The player shall be able to view character stats and enemy stats at anytime.

PGR18. If a character dies, then they are removed from play.

PGR19. If an enemy dies, then they are removed from play.

PGR20. If a character dies and there remains only one final character, the immediate turn after the current phase is the remaining character.

PGR21. When a new game is initiated, the player is given base character, resources, and equipment.

PGR22. A player may travel to any node that is connected to the current node if they have the correct amount of resources.

PGR23. A node consists of Dialogue phase, an acquire resources phase, a possible action phase and a possible merchant phase.

PGR24. Each region has a terminating node that has a boss battle.

PGR25. Once the boss has been defeated, the player will proceed to the next region.

PGR26. The game is won if the player defeats the final boss in the final region.

PGR27. The game is lost of the player runs out of the supply resource, unable to move.

PGR28. If the player losses the game, they are given a graphic indication and are returned to the main menu. The file is deemed as unplayable.

PGR29. If the player wins the game, they are given a graphic indication and are returned to the main menu. The file is deemed as unplayable.

PGR30. If the merchant phase is triggered, the merchant menu is opened.

PGR31. While the merchant menu is open, the player is able to view their resources, the rest of the map, and is able to exchange resources and equipment with the merchant for fixed prices.

PGR32. Once the merchant menu is close, it can not be opened again on the same node.

PGR33. If the combat phase is triggered, the player is moved to the Combat phase with current equipment layouts and resources.

PGR34. If the player returns from combat phase, the rest of the node shalt progress as expected.

PGR35. The Combat map will be created with the correct N by N size.

PGR36. The player and the AI can decide either to attack the health values or the armour values of their respective enemies.

8.3 Alternate Game Modes

The following section will cover the management game mode of the game, they are still important to the game but no active strategy is being conducted.

AGR1. If the edit character layout button is pressed, then the respected menu is opened, listing all available characters.

AGR2. While in the edit character layout menu, if a character is selected, then the character menu is opened displaying character stats and current equipment layout. The equipment menu is also opened which has all unequipped items.

AGR3. While in the character menu, the player can equip and unequip items to the character if they meet the subsequent requirements.

AGR4. If the player hovers over the stat values, a tool tip will indicate where those values are derived from.

AGR5. If the player hovers over the equipment, a tool tip will indicate a description of the weapon and it's stat bonuses. The character's stats will also be temporally updated to show which stats are better or worse compared to the current equipment.

AGR6. If the player selects the back button in the character menus, they will be returned to the list of characters.

AGR7. If the player selects the back button in the edit character layout menus, they will be returned to the overhead world view.

AGR8. Items will be listed in a grid system with respected image and border colour to indicate item type.

AGR9. When items are selected, their information will be displayed to the screen.

AGR10. If items are of the consumable type, the use of this item will remove the item from play and the player's item list.

AGR11. If items are of the expendable type, the use of this item will decremented until they reach 0, where they will be refilled when the combat phase ends.

AGR12. If items are of the rechargeable type, the use of the item will not allow the player to use the item again until the number of cooldown turns been completed.

AGR13. If items are of the passive type, the item applies it's ability every turn for the course of the combat phase. These items are not "used" like other items.

AGR14. If items are of the instant type, the item can be used for every turn of the assigned character.

AGR15. If an item is equipped to character, no other characters are able to equip that same instance of the item.

AGR16. If the mouse is hovering over a desired item and the item changes the stats of a character, those stats will be updated on the selected character with incremented values being coloured in blue, decremented stats being coloured in red and unchanged values being coloured in white.

8.4 Menus and other Systems

This section will cover requirements that are related to the different menus of the game. There are a list of menus in the game, but the two main menus are the Main menu that is shown when the game is first initialized and the In Game menus which is the menu that will be displayed while the game is being played. The requirements of both these menus are:

MR1. The Main Menu is opened when the game is first initialized or when the player exits the current game session.

MR2. The Main Menu is closed when the player exits the game or enters a game session.

MR3. The Game Menu is opened when the esc key is pressed or the menu button is pressed.

MR4. The Game Menu is closed when the esc key is pressed or the close menu button is pressed.

MR5. For both menus, if the audio button is selected then Audio options is opened.

MR6. For both menus, if the graphics button is selected then Graphics options is opened.

MR7. For both Audio and Graphics menus, if the back menu is selected then the changes are saved and the respected menu is closed.

MR8. A new game menu is created if the player presses the New Game button from the Main Menu.

MR9. A Load game menu is created if the player presses the Load Game button from the Main Menu.

MR10. A Save game menu is created if the player presses the Save Game button from the Main Menu.

MR11. All sub menus can be closed if the player presses the esc key or the corresponding close menu key.

MR12. The player should be able to access the the Item's menu from the Overworld menu by clicking the respected button or the respected hotkey.

9 Look and Feel Requirements

9.1 Appearance Requirements

APPR1. The Menus should have simple colours and patterns.

APPR2. The project should have the same format that a modern game would conform to.

APPR3. Buttons, Menus and item backgrounds should be a truncated square.

9.2 Style Requirements

SR1. The game world should make up of only 2d sprites, no 3d models.

SR2. The character, enemy, NPC and terrain should consist of only 3-4 colours, keeping the style simple.

SR3. The map pallets should only have around 16 colours for the pallet of the world.

SR4. The weapons, icons, costumes and text should be of the sci-fi theme.

9.3 Requisite Assets

We used the turn based strategy framework (TBS Framework) from the Unity Asset store because it suited the nature of the combat phase of our game very well. It handled turn taking, calculating and applying damage, the stacking of certain combos and victory conditions.

9.4 Audio

Asset Type	Number Required	Rationale
Winter Biome Background	2	We felt that this music was appropriate for the winter levels of our games and represent the biomes correctly
Main Menu	1	We felt that this is the appropriate music that can be used for the overall theme of the game, it matches the main menu title well.

9.5 Visual

All items listed below will be referenced in the their respected directory to reduce clutter on this document, it is encourage that the reader view the photos in the Assets folder while reviewing the sections below.

Sprite1. Kroner - One of the main characters. Matching file name is Kronder-Proto.png

Sprite2. Lee - One of the main characters. Matching file name is LeeProto.png

Sprite3. Robot - One of the main enemies. Matching file name is Robot1.png

Texture1. Snow - The main title texture for the winter biome.

UI1. Blue Menu Button - The main image that will be used for the buttons in the menu. Matching file is BluePanel.png

UI2. Title Page - The main menu image that the player first sees when the game is loaded. Matching file is TempTitle.png

Environment1. Rock - The defualt rock used in the winter biome.

10 Usability and Humanity Requirements

10.1 Ease of Use Requirements

EUR1. The product shall be usable by a student in 6th grade or equivalent education.

EUR2. The player should be able to use an xbox 360 controller if desired.

10.2 Personalization Requirements

PR1. Application shall not have any region specific language or icons

10.3 Learning Requirements

LR1. The project shall be easy to use by someone who is familiar with operating a graphical computer interface.

LR2. A first time user shall not have difficulty understanding what the project does or how to use it.

10.4 Understand ability and Politeness Requirements

UPR1. The project should be language ambiguous and limited, emphasizing more on showing the player rather than telling.

10.5 Accessibility Requirements

AR1. The product shall be usable by colour blind users.

11 Performance Requirements

11.1 Speed and Latency Requirements

SLR1. The program should not hang or freeze for a long period of time.

SLR2. Loading and saving periods should not result in disrupted periods of play

SLR3. The application shall load on most standard computers running Windows 10 or IOS and should not take extensive periods of time to initiate.

SLR4. After the application is installed on the device, it shall be available for use 24 hours per day, 365 days per year.

11.2 Precision or Accuracy Requirements

PAR1. The mouse clicks should be accurate with ray casts up to error of 2 pixels.

PAR2. Seed generations should give near perfect maps and outcomes if the same seed was to be used again.

11.3 Reliability and Availability Requirements

RAR1. The tutorial and tool tips should be easy to understand and the provide unique incite to the player.

11.4 Robustness or Fault Tolerance Requirements

RFR1. The program should not crash during save or loading periods.

RFR2. The program should make frequent auto saves during run time to help reduce the loss of data.

11.5 Capacity Requirements

N/A

11.6 Scalability and Extensibility Requirements

SER1. The project should be able to be able to handle scalable map sizes and have the flexibility to add new cards and attachment to characters.

11.7 Longevity Requirements

LR1. The project shall be updated and supported to function on the latest version of the Unity Engine for a minimum of four years after the project is launched.

12 Operational and Environmental Requirements

12.1 Release Requirements

RR1. Every project "update" release shall not cause previous features to fail

RR2. The project should be ready for public-release by April 14, 2018

RR3. The project should be able to run on any desktop or laptop machine running Windows, IOS or Linux. The designated machine should have a graphics output, audio output and a keyboard and mouse input.

12.2 Expected Physical Environment

EPE1. The application shall function in all physical environments the devices the application is running on is able to operate.

13 Maintainability and Support Requirements

13.1 Maintenance Requirements

MR1. The application's documentation shall be relevant and up to date, being updated within one week of any major changes made

13.2 Supportability Requirements

SUPR1. There shall be adequate supporting documents of the project after every major release.

SUPR2. There will be a supporting manual to be added to the final release of the project.

13.3 Adaptability Requirements

AR1. The program is expected to run on Windows, IOS and Linux machines.

14 Security Requirements

SR1. Developers of the application and administrators of the SE 4GP4 class, course TA and professor respectfully, should be able to access the source files of the project.

SR2. Application should not include and work or files that were not included in the original GitLab file.

SR3. Application shall not store user's personal information or data.

SR4. Application shall not transmit user's personal data.

15 Cultural Requirements

CR1. Application shall not be offensive to religious or ethnic groups.

CR2. Application shall not display offensive imagery to the user, or contain offensive language.

16 Legal Requirements

There are a few Compliance requirements for our project.

LR1. The project will abide by all Canadian laws

LR2. The project will not infringe on any existing intellectual property or copyright

LR3. The project will abide by the Documentation Redistribution Policy standards

16.1 Compliance Requirements

There are a few Compliance requirements for our project.

COMPR1. The development team must make reference to all 3rd party game assets that are used in the project and declare that such assets are not property of the development team.

COMPR2. The development team will agree to follow Unity's Terms of Service, Private Policy and Copyright Policies.

16.2 Standards Requirements

N/A

17 Project Schedule

The following section has a table of the what will be covered in the coming months. Currently with the early stages of development, the list of things needed to be completed is subject to change and should be taken with an approximation of what is to be completed.

Month	List of tasks to be completed
October	Finalize first Project demo, Develop character art assets, Develop Random Map Generator, Develop Node Generator
November	Test Map Generator, Test Node Generator, Develop Combat Phase Setup, Develop terrain art assets, Develop map textures
December	Finalize Implementation Design Document first draft, Develop Story dialogue, Develop event dialogue, Develop Tool tip format.
January	Finalize V&V Document first draft, Fully Develop Combat Phase, Develop remaining character assets, Develop special effects assets.
February	Finalize Concept and Requirements Document, Develop event list, Develop event assets, Develop Tool tip mechanics, Redevelop Map Generation for current needs.
March	Finalize Implementation Document, Develop audio recordings, Test Tool Tip mechanics, Perform system testing, Perform game testing.
April	Finalize Project Demo, Create standalone version, Finalize V&V Document, Finalize testing, Finalize assets.

18 Risks

The following are risks that may be associated with the project

R1. **The system does not de-alocate pointers causing a memory leak on the user's computer.** This risk will be minimized by using the C# Script programming language which has a different garbage collection system. Unless the developers use pointers, there should be next to no de-alocation problems.

R2. **The program will cause physical injury to the user such as causing eye soreness or seizures.** This risk will be minimized by limiting the number of flashing imagery.

19 Costs

There may be no costs on the development team unless the team proceeds to purchase $10 worth of game assets or any other 3rd party software.

20 User Documentation and Training

20.1 User Documentation Requirements

The Game Developers will try to meet the same coding styles throughout the development process. These styles will include having the same spacing, parenthesis format and tabulation. The development team will also attempt

to document all trivial functions and describe what each created script does and when it should be used. The Developers will also keep a log book handy between group members as an physical reference to team meetings and work being completed.

20.2 Training Requirements

The Game developers will need to learn a few concepts, languages and how to use a few programs during the course of this project.

1. **Graphic layering and 2d projections** This will be useful for when we try to layer graphics on top of each other in a orthographic view. It will help us polish the game better.

2. **C# Scripting Language** This is the primary scripting language that Unity uses for it's games and will need to understood so that each member can contribute developing the game.

3. **Photoshop** This will be useful for developing any game assets that cannot be found on the Unity asset store. It will be used to develop character sprites, textures and GUI.

4. **Unity Interface** Unity is the game engine that we will be using and understanding how the engine works, how to put simple levels and scripts together will be necessary to complete the project.

21 Waiting Room

The following are a list of features that may be added to the project if time persists once all requirements have been fully implemented and polished.

1. In game wiki that lists every game mechanic, character strengths and weaknesses, equipable item description and effects, consumable item description and effects, lore and strategies.

2. Multiple endings for completion of the game, the more endings, the more diverse the options will need to be.

3. A Score mechanic that will track how well the use performs given difficulty, resources, and choices made.

4. A save button that will allow the user to manually save their progress rather than autosave.

22 Off the Shelf Software

There are a off the shelf software that will be needed to develop the game's assets.

OSS1. The game's textures and character sprites will need to be created in either Photoshop or Gimp 2 depending on the developer.

OSS2. The game's models will need to be created in either Blender or Maya.

23 Ideas for Solutions

The following ideas for solutions to problems that we may face during the course of the project:

1. To solve our graphic layering problem, we will look to many bird eye view games such as Don't Starve or Roller Coaster Tycoon to see how the developers seamlessly layered their graphics in an orthographic manner.

2. To solve our problem of finding at least one continuous pathway, we will reference Graph Theory to ensure that all maps can be determined as continuous.

3. To ensure our game is fair to the player we will look into games that have similar mechanics and try to see which game made their mechanic more fair and less dependent on chance. A few contrasts that we will make will be looking at XCOM vs Mario + Rabbits Kindom battle, FTL and The Binding of Isaac.

4. To ensure that our art style is simple and understandable, we will reference Dustforce's art style and take note to what makes it simple and how to maximize the limited number of colours on our palette.