Software Requirements Specification

for

ProjectRPG

Version 1.0

Prepared by Filip Łukaszewski and Mateusz Pisera

2IP

15.05.2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Filip Łukaszewski and Mateusz Pisera 2IP	30.04.2024	-	1.0

Filip Łukaszewski and Mateusz Pisera 2IP	15.05.2024	Make documentation more legible and understandable	1.1

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the requirements and specifications for the development of an RPG game.

1.2 Document Conventions

- Classes, functions, and methods are denoted using code formatting.
- User inputs and outputs are represented in quotation marks.
- System messages and prompts are italicized.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, designers, testers, and stakeholders involved in the development and testing of the RPG game. Readers should be familiar with software development concepts and game development principles.

1.4 Product Scope

The RPG game aims to provide players with an immersive gaming experience where they can explore a dynamic game world, engage in combat with enemies, manage inventory, interact with shops, and progress through levels.

1.5 References

None

2. Overall Description

2.1 Product Perspective

The RPG game operates as a standalone application with no direct dependencies on other systems. It provides a user-friendly interface for players to interact with the game world.

2.2 Product Functions

- Player movement and interaction with the game environment.
- Combat system allowing players to engage in battles with enemies.
- Inventory management for storing, displaying stats deleting, moving and equipping items.
- Shop system for buying and selling items.
- Leveling up mechanics for enemies.
- Displaying player stats and game progress.

2.3 User Classes and Characteristics

Users of the RPG game include:

Players: Individuals who interact with the game to explore, battle enemies, manage inventory, and progress through levels.

2.4 Operating Environment

It can be compiled and run on various platforms that support C++ compilers, such as Windows, Linux, and macOS, with standard input/output capabilities.

2.5 Design and Implementation Constraints

- The game is implemented in C++, utilizing object-oriented programming principles.
- Platform-specific dependencies should be minimized for portability.
- Compatibility with common operating systems and hardware configurations should be ensured.

2.6 User Documentation

User documentation, will be provided to guide players on how to play the game.

2.7 Assumptions and Dependencies

- Assumption: Players have basic familiarity with keyboard input and game controls.
- Dependency: The game relies on standard input/output libraries for user interaction.

3. External Interface Requirements

3.1 User Interfaces

- The game provides a graphical user interface (GUI) for displaying the game world, menus, and player interactions.
- User input is primarily via keyboard controls for movement and interaction.

3.2 Hardware Interfaces

No specific hardware interfaces are required beyond standard input/output devices (e.g., keyboard, monitor).

3.3 Software Interfaces

The game interfaces with the C++ standard libraries for input/output operations and random number generation.

3.4 Communications Interfaces

- Player-Shop Interaction: The player can interact with the shop to buy, sell, or view items. The shop displays its inventory in a grid format, and the player can navigate through it using arrow or WASD keys. The player can press Enter to select an item and then choose to buy or sell it.
- Player-Inventory Interaction: The player can view their inventory, which is displayed in a grid format. They can navigate through it using arrow or WASD keys and press Enter to select an item. Once an item is selected, the player can view its details, move it to another slot in the inventory, or delete it.
- Player-Enemy Interaction: When the player encounters an enemy, they can choose to attack, defend, or escape. If the player chooses to attack, they will deal damage to the enemy based on their weapon's attack power. If the player chooses to defend, they will resist the damage they take from the enemy's next attack. If the player chooses to escape, they will leave the battle and return to the game board.

- Game Board Navigation: The player can navigate the game board using arrow or WASD keys. They can move up, down, left, or right, and the game board will update to reflect their new position. If the player moves onto a shop or enemy tile, they will enter a shop or battle screen, respectively.
- Pause Menu: The player can pause the game by pressing the Escape key. This will bring up a pause menu, where the player can choose to continue playing or exit the game. If the player chooses to continue playing, the game will resume from where it was paused. If the player chooses to exit the game, the game will end.
- Main Menu: When the game first starts, the player is presented with a main menu, where they can choose to start a new game or exit the game. If the player chooses to start a new game, the game will begin. If the player chooses to exit the game, the game will end.

4. System Use Cases

1. **Start Game**

- ***Objective: To start a new game.***
- *Priority: High*
- *Source: Player*
- *Actors: Player*
- *Flow of Events:*
- ***Basic Flow:***
- *Player selects "New Game" from the main menu.*
- ***Alternative Flow:***
- *Player selects "Exit" from the main menu.*
- *System exits the game.*
- ***Preconditions:***
- *Player must be in main menu to start a new game.*
- ***Post conditions:***
- *System initializes the game.*
- *Player starts playing the game.*

2. **Move Player**

- ***Objective: To move the player character around the game board.***
- *Priority: High*
- *Source: Player*
- *Actors: Player*
- *Flow of Events:*

- ***Basic Flow:***
- *Player presses the arrow or WASD keys to move the player character.*
- ***Alternative Flow:***
- *Player tries to move the player character into a blocked space.*
- *System prevents the player character from moving into the blocked space and spawns player at the other side.*
- ***Preconditions:***
- *Player can’t be in a menu or shop*
- *Player must press arrow or WASD keys in case to move player character*
- ***Post conditions:***
- *System updates the player character's position on the game board.*

3. Interact with Shop

- ***Objective: To buy or sell items from the shop.***
- *Priority: Medium*
- *Source: Player*
- *Actors: Player, Shop*
- *Flow of Events:*
- ***Basic Flow:***
- *Player moves the player character to the shop location on the game board.*
- *Player presses "Y" to enter to the shop options.*
- ***Alternative Flow:***
- *Player does not have enough gold to buy an item.*
- *System displays an error message.*
- *Player cancels the transaction.*
- *System returns to the shop options.*
- ***Preconditions:***
- *Player must stand on shop location on game board.*
- *Player must press exact button to enter the shop*
- *If player wants to buy items, he needs to have enough gold.*
- ***Post conditions:***
- *System displays the shop inventory, when player wants to buy an item.*
- *System displays the player´s inventory, when player wants to sell an item.*
- *Player selects an item to buy or sell.*
- *System updates the player's inventory and gold.*

4. **Interact with Enemy**

- ***Objective: To fight or escape from an enemy.***
- *Priority: High*
- *Source: Player*
- *Actors: Player, Enemy*
- *Flow of Events:*
- ***Basic Flow:***
- *Player moves the player character to the enemy location on the game board.*
- *If the enemy is defeated, the player receives gold and enemy's level increses by 1.*
- ***Alternative Flow:***
- *Player tries to escape from the enemy but fails.*
- *System calculates the damage dealt by the enemy.*
- *System updates the player's health.*
- *If the player's health reaches zero, the game is over.*
- ***Preconditions:***
- *Player must move to enemy location.*
- *Player BATTLES or ESCAPES the enemy.*
- ***Post conditions:***
- *Player selects "Fight" or "Escape" from the enemy options.*
- *System calculates the damage dealt by the player and enemy.*
- *System updates the player's and enemy's health.*

5. **Manage Inventory**

- *Objective: To manage the player's inventory.*
- *Priority: Medium*
- *Source: Player*
- *Actors: Player*
- *Flow of Events:*
- ***Basic Flow:***
- *Player presses "E" key on the gameboard.*
- ***Alternative Flow:***
- *Player tries to use an item that is not equipped.*
- *System displays an error message.*

- *Player cancels the transaction.*
- *System returns to the inventory menu.*
- ***Preconditions:***
- *Player must be in inventory page.*
- *Player needs to manage what he'll do with item (if he has one).*
- ***Post conditions:***
- *System displays the player's inventory.*
- *Player selects an item to display stats, delete, move and equip.*
- *The system shows also additional options for the equipment: expand and display player stats.*
- *System updates the player's inventory and equipment.*

6. Display

- *Objective: System shows the current state of the game board, including the player's position, shop location, and enemy position.*
- *Priority: High*
- *Source: GameBoard*
- *Actors: Player*
- *Flow of Events:*
- ***Basic Flow:***
- *Player moves around the game board using arrow keys or other input methods.*
- *Game calls the display method of the GameBoard object to update the screen with the current state of the game board.*
- *Display method clears the screen, draws the game board, and displays the player's position, shop location, and enemy position.*
- *Player continues to interact with the game, and the display method is called repeatedly to update the screen.*
- ***Alternative Flow:***
- *If the player enters the shop, the game calls the showInGameShop method instead of the display method.*
- *If the player enters the shop, the game calls the showInGameShop method instead of the display method.*
- ***Preconditions:***
- *GameBoard object has been initialized with the correct size and positions for the player, shop, and enemy.*
- *Player has input a valid movement command or other action that triggers the display method.*
- ***Post conditions:***
- *Screen has been updated to show the current state of the game board.*

- *Player's position, shop location, and enemy position are accurately displayed on the screen.*
- *Player can continue to interact with the game.*

7. Pausing game

- *Objective: System allows player to temporarily stop the game and return to the main menu.*
- *Priority: High*
- *Source: Game*
- *Actors: Player*
- *Flow of Events:*
- ***Basic Flow:***
- *The player presses the 'P' key.*
- *Game calls the pauseGame method of the Game object.*
- *pauseGame method displays the pause menu, which includes options to continue or exit the game.*
- *Player selects the 'Continue' option using the arrow keys or other input methods.*
- *pauseGame method returns control to the game loop, and the game continues.*
- ***Alternative Flow:***
- *If the player selects the 'Exit' option in the pause menu, the game exits the program with the code: 0.*
- ***Preconditions:***
- *Game object has been initialized and the game is running.*
- *Player has input a valid command to pause the game.*
- ***Post conditions:***
- *Game has been paused, and the player can view the pause menu.*
- *Player can choose to continue or exit the game.*
- *If the player chooses to continue, the game resumes from the same state as before.*
- *If the player chooses to exit, the game ends.*

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The game should run smoothly on standard hardware configurations without significant lag.
- Loading times for game elements should be minimal.

5.2 Safety Requirements

The game should not perform any actions that could cause harm to the user or their system.

5.3 Security Requirements

The game should not collect or transmit any user data without explicit consent.

5.4 Software Quality Attributes

- Player movement and interaction with the game environment.
- Combat system allowing players to engage in battles with enemies.

6. Other Requirements

No other specific requirements are identified at this time.

• System Requirements Chart

ID	Priority	Type	Source	Contained in Use Case(s)	Description
1	High	F	Filip Łukaszewski	UC1	The system shall allow the player to move the character on the game board using arrow keys or WASD keys.
2	High	F	Filip Łukaszewski	UC4	The system shall display the game board with the player's current position, shop, and enemy positions.
3	Medium	F	Mateusz Pisera	UC1, UC2	The system shall allow the player to enter the shop when the player's position matches the shop's position.
4	Medium	F	Filip Łukaszewski	UC2	The system shall allow the player to buy items from the shop using the player's gold.

ID	Priority	Type	Source	Contained in Use Case(s)	Description
5	Medium	F	Mateusz Pisera	UC2	The system shall allow the player to sell items to the shop for half of the price
6	High	F	Mateusz Pisera	UC5	The system shall allow the player to view and manage their inventory.
7	High	F	Filip Łukaszewski	UC5	The system shall allow the player to equip items from their inventory.
8	High	F	Filip Łukaszewski	UC1, UC3	The system shall allow the player to attack the enemy when the player's position matches the enemy's position.
9	Medium	F	Mateusz Pisera	UC3	The system shall allow the player to defend against enemy attacks.
10	High	F	Filip Łukaszewski	UC5	The system shall display the player's current gold, HP, equipped items, and other relevant stats.
11	Medium	F	Mateusz Pisera	UC7	The system shall allow the player to pause the game and display a pause menu.
12	Medium	F	Mateusz Pisera	UC7	The system shall allow the player to exit the game from the main menu or pause menu.
13	Low/Medium	NF	Mateusz Pisera	N/A	The system shall have a responsive and user-friendly interface.
14	High	NF	Filip Łukaszewski	UC6	The system shall display the current game status in the terminal

Use Cases:

- UC1: Player Movement
- UC2: Shop Interaction
- UC3: Enemy Interaction
- UC4: Start Game
- UC5: Manage Inventory
- UC6: Display
- UC7: Pausing Game

Appendix A: Glossary

No glossary terms are defined at this time.

Appendix B: Analysis Models

No analysis models are included in this document.

Appendix C: To Be Determined List

No items are listed as "to be determined" in this document.