**Software Requirements Specification**

**For**

**Project R**

|  |  |
| --- | --- |
| **Instructor:** | Professor Char |
| **Team Members:** | Min Dye, Scott Hargrove, Peter Wainwright, Ryan Do |
| **Cycle:** | 2017-2018 |
| **Date Submitted:** | 30 January 2018 ***(Updated 7 April 2018)*** |

Document template copyright 2005-2015, CCI Faculty. Version 2.3. Use permitted under Creative Commons license CC-BY-NC-SA. See http://creativecommons.org/licenses/by-nc-sa/3.0/.

***(changes are written in bold and italic)***

1 **Introduction**  
This section provides an overview of Project R. Scope provides a short description of the product and how it is useful; Definitions explains terms with which a reader may not be familiar; and User Profile identifies the ways different groups of people would make use of Project R.

1.1 **Scope (“Steam Description”)**

Project R is a 2D side-scroller game with a permanent death. ***The game resets all of the character’s stats upon death and restarts from the first stage.*** The player controls a character to explore the ominous world to face enemies, obstacles, and puzzles. Enemies progressively get stronger, and the only way for you to survive is to gather items. At the end of the adventure, a powerful foe awaits for you.

1.2 **Definitions, Acronyms, and Abbreviations**

There are no definitions, acronyms and abbreviations for our project during this time.

1.3 **User Profile**

***The targeted users are teenagers to mature audiences between age of 18-35 since there is some violence present We expect more males than females will play the game.*** Users will be referred as players.

2 **External Interfaces**  
This section identifies ways in which Project R interacts with people and other systems.

2.1 **User Interface**

Our game will have a main menu, pause menu and in-game user interface for the player to interact with. The main menu includes three buttons: Play, Options***, Restart***, and Exit Game. Pause menu also contains Options***, Restart,*** and Exit buttons. In-game UI includes player character’s HP bar, currently equipped items, and currency. More elements may be introduced during development phase.

2.2 **Data Interface**

***The game will store some variables on a local file to avoid prompting tutorials more than once. Other variables may be added during development.***

3 **Specific Requirements**

3.1 **Functional Requirements**  
The statements below define the functional requirements for the system.

001 - Main Menu Buttons  
 Main menu has ***four*** functional buttons: “play” starts the game, “options” allows users to adjust game settings, ***“restart game” resets the current session,***  “exit game” quits the program.

002 - Tutorial

If players are playing for the first time, a tutorial will explain how to play the game.

003 - Enemy Patterns

Enemies must be able to maneuver in different ways to challenge players. ***Enemies will follow the player if the player is within their range, but will return to their original behavior if the player leaves its range.***

004 - Combat

Players can attack enemies to damage them, and vice versa. When a character’s HP falls under 0, the character dies. ***Once the character dies, they will restart at the start of the very first level, with no items. (basically a “Game Over”)***

005 - Weapon Slots

Player can press key 1-4 to change between weapons. Currently equipped weapon on the selected slot drops when you pick up a new one.

006 - Item Drops

Killing enemies have chance to drop equipments.

007 - Equipments

Equipment can be picked up from the ground. Each equipment has different stats and attack types. ***Rather than picking up equipable items off the ground, we may implement minor storage assets (like chests)***

008 - Currency

Killing enemies drops currency depending on their strengths.

009 - Save

The program will save game data when prompted.

010 - Augmentation

Currency can be used to augmentate player’s equipments, which assigns random stats to the weapon.

011 - Pause

Players can pause the game by pressing a button, which brings up a pause menu that allows users to change settings or quit the game.

3.2 **Performance Requirements**  
The statements below define the performance requirements for the system.

001 - Operating System

The game will be run on Windows systems, primarily the latest versions. There is no plan to port the game to any other platform.

002 - Graphics Card

The graphics card will allow the program to display content on users’ monitors.

003 - Central Processing Unit (CPU)

The game will require at least a SSE2 Instruction Set Support in terms of CPU. Most computing systems made past the year 2001 will have this or better. ***The game will not require high-performance CPU.***

004 - Keyboard Support

The hardware that the game will need to control the character, adjust settings, etc.

005 - Response Time  
 There should not be a significant delay between the player pressing a button and the action happening in the game.

006 - Saving

Save files must be relatively small, and easily importable by the program.

007 - Error Message

When an error occurs, a message should appear displaying the error message.

008 - Downtime

If there is a downtime on the program, it will not harm any other system.  
  
3.3 **Design Constraints**

3.3.1 **Constraint: Project R will be Windows only**   
 Reason: Developing for other platforms like Macintosh and mobile devices will increase difficulty of the project and near impossible to finish within the deadline.

3.3.2 **Constraint: Music will not be developed in the house**

Reason: Our team does not include a sound engineer, and all hands are on artwork and programming. Trying to learn how to create music will increase time spent on the project. Instead, we will utilize copyright-free music.

***3.3.3 Constraint: Storytelling***

***Reason: A story and the atmosphere must be determined in order to have a clear transition from the start to the end.***