# 1.0. Introduction

## 1.1. Purpose

This document will present a detailed description of the EmojiStrike game. It will introduce the purpose and scope of this game, followed by a detailing of the features, requirements, and constraints of the product. The document will also contain cases which explain how the game system will react to stimuli from the players. The intention of this Software Requirements Specification document is to serve as a future reference point for the developers and the clients, as well as a map for the future development of the project.

## 1.2. Scope of Project

This game is a multi-player turn-based game, shown to the user as a 2-D map on the monitor screen. The game consists of avatars belonging to the players dropped in random places on a pre-defined map, followed by players then attacking each other with ranged and un-ranged weapons, with the last remaining player being the victor. Power-ups designed to enhance gameplay (for example, by making a player temporarily invincible, or increasing the potency of their weapons, or increasing a wounded player’s health) will be added to the map at certain time periods during the course of the game.

The system will be designed to maintain player interest and interactivity in the game, as it is intended as an object of leisure. It will attempt to maintain player participation until the end of the game, and will provide challenges to the player while also maintaining a record of previous victors, in order to foster competition.

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Power Up | Objects which boost the abilities of the player that captures them. |
| Avatar | The image used by the player to represent themselves. |

## 1.4. References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

## 1.5. Overview of Document

The Overall Description section of this document gives an overview of the functionality of the game. It describes the informal requirements and is used to establish a context for the technical requirements specification section that follow.

The Requirements Specification section of this document is intended primarily for the developers, and so describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same game, but since they are intended for different audiences, different language has been used.

# 2.0. Overall Description

## 2.1 Product Perspective

Our product is self-contained, and standalone in terms of functionality. There are no external data servers required since it will be a turn-based game in the same machine. Our main actor is the players who are going to play the game.

***2.1.1*** ***System Interfaces***

The user will use the keyboard to pass in inputs. While selecting an Emoji avatar for each player, and the map of the game, players will use the four arrow keys to navigate the menu, the Enter key to select, and the Backspace key to move back. During gameplay, the W, A, S, D keys will be used to move the character, and the four arrow keys will be used to aim the weapon, while the Space Bar will be used to fire.

The “P” button will be used to move the game in and out of the Pause mode. The “R” button will be used to restart the game and bring it back into the main menu, and the “Q” button will be used to quit the game.

The outputs will be passed using the computer monitor and speakers. The user interface display will be visible on the monitor, and game sounds will be audible through the speakers.

The game will be saved in a sorted order according to the date. The file which will be used will be in the root directory of the game, and will be a CSV file to keep track of different data easily. The data will consist of the game state, date, and a desired name to each save file.

***2.1.2*** ***User Interface***

The game will be displayed in the maximum windowed resolution. In the top-left corner will be a menu with options to pause, quit, and restart the game. Next to the commands will be the letters which should be pressed to bring the user into the desired state. The rest of the available space will be occupied by the game.

When beginning the game, the users will select the number of players. Following this they will select the Emoji and set the name for each player in sequence, and then finally select the map they will play on, after which they game will start.

The users will select Emojis from a list, and type in the name for each player. They will also select the number of players from a list of pre-approved numbers. A table of maps with small previews will allow them to select the map.

***2.1.3*** ***Hardware Interfaces***

Our program will use Java’s automated HID USB device recognition to connect our application to hardware. The required keyboard, sound, and display interface will be detected automatically without the manual utilization of physical ports. Java will interpret the signals automatically and convert them to Java Virtual Machine’s inputs. The supported devices matter on the port availability between computer and its physical port interfaces. It can be USB-C, USB-A, HDMI, etc.

***2.1.4*** ***Software Interface***

The game will be written in Java language, and will interact with an operating system of any platform using the Java Virtual Machine. The Java specifications are:

Java Version 8 Update 121 from www.java.com

***2.1.5*** ***Communications Interface***

N/A

***2.1.6*** ***Memory Constraints***

In the Java Virtual Machine, -Xmx is 2 gigabytes, and the -Xms is 256 megabytes. The program will use the default settings.

***2.1.7*** ***Operations***

There are four modes of operation. There is the “Loading” mode, the “Main Menu” mode, the “Pause” mode, the “Game” mode.

During the “Loading” mode, no keys can be used. There are no user operations in this mode, since the program is loading the game.

Throughout all the following modes, the users can use the “Q” key to quit the game, and the “R” key to bring them back to the main menu and delete any progress.

In the “Main Menu” mode, the user can use the arrow keys to navigate the menu, the Enter key to select, and the Backspace to move them back in the menu. Apart from these keys and “R” and “Q”, all other keys are deactivated.

In the “Pause” mode, the user can only use the “R”, “P” and “Q” keys. The “P” key moves the user back into the “Game mode”.

In the “Game” mode, the user can use all available keys. Arrow keys will change the weapon’s aim, while the “W”, “A”, “S”, “D” keys will navigate the player’s character. The “P” key will move the game into “Pause” mode, and the “R” and “Q” keys will keep their functionality. The Space Bar will shoot the weapon.

***2.1.8*** ***Site Adaptation Requirements***

1. The data initialization will be done in the same way for every deployment since there will be no difference when application will be deployed into identical Java Virtual Machines. During the menu state, the texture files and menu music will be retrieved from the aspects file of the game. During the play state, the map and player figures will be rendered from the selected pre-existing map file before the game starts. There will be additional components which will procedurally be required, like sound and visual effect files. This data will be rendered real-time since their timing and locations are unknown before-hand.
2. Since our application will not change functionality in different areas of deployment, we will not have any specific mission related features.

***2.2*** ***Product Functions***

Allow user to start game

Allow user to pick avatar

Allow user to choose weapon

Allow user to shoot other characters

Allow user to move on map

Save (?)

***2.3 User Characteristics***

Basic knowledge of how to use a keyboard required

No technical expertise, special knowledge, or extensive experience required

Game in designed to be intuitive

***2.4 Constraints***

Create in Java

Database to store records (?)