# **1 Introduction**

## 1.1 Purpose of application

The main goal of the project is to implement a simple but entertaining desktop game where two players can compete against each other. We aim to give the game a high replay value through both level variety and good design, and an interesting competitive aspect.

## 1.2 General characteristics of application

The game will be a desktop, single- or multi-player application with a graphical user interface for the Windows/Mac/Linux platforms.

The game will be in real-time, which means that the application handles input from both players simultaneously. There are no time constraints to the application; a round of the game will be finished when either player has brought the other players health to zero, or when cancelled manually.

The game will be set on a flat stage where the players can move around freely in two dimensions and shoot their weapons at each other while also picking up power-ups to boost their character.

## 1.3 Scope of application

In addition to playing the game against another human player, there is also the possibility of playing against an AI-controlled computer-player. The application will not record any data or statistics of previously completed game rounds.

## 1.4 Objectives and success criteria of the project

It should be possible for one or two players to play and finish a round of the game that is both entertaining and runs smoothly. The game does not need to be highly advanced or plastered with functions, but rather characterized by smart design and a minimalist, but appealing graphical appearance.

## 1.5 Definitions, acronyms and abbreviations

# **2 Requirements**

## 2.1 Functional requirements

The player(s) should be able to:

* Start a new game.

1. Select player vs. player or player vs. computer.

* Control their character in the following ways

1. Move their character in every direction.
2. Aim and shoot their weapon independently of which direction they are moving in.
3. Perform a dashing move, which gives the player a quick speed burst.

* Pick up power-ups by running into them, boosting the player with an advantage of some kind.
* Exit the application. This will end the ongoing round.

## 2.2 Non-functional requirements

### 2.2.1 Usability

The game should be easy to learn, but difficult to master. More specifically, normal users should be able to pick up the game and complete a game-round within a short period of time, while advanced users who are more used to computer gaming should understand the game right from the get-go, but still be required to play several rounds to figure out the more advanced features and start forming their own strategies.

Tests with multiple users, both normal and advanced, should be performed to verify the usability. Test results should be part of the final documentation.

### 2.2.2 Reliability

### 2.2.3 Performance

Since the game will be very fast paced, a low input response time will be required. The aim is to achieve a response time as low as possible while keeping a threshold at maximum 50ms.

### 2.2.4 Supportability

### 2.2.5 Implementation

To achieve platform independence the application will use the Java environment. All hosts must have the JRE installed and configured. The application needs to be installed on all hosts where it will run.

### 2.2.6 Packaging and installation

The application will be delivered as a zip-archive containing:

1. A file for the application code
2. All needed resources (models, textures e.t.c.)
3. Start programs to start the game on the different platforms
4. A README-file documenting installation and start of application.

### 2.2.7 Legal

The game will be designed from scratch, including player and stage models. Therefore, no legal issues will be present.

## 2.3 Application models

### 2.3.1 Use case model

See APPENDIX for UML diagram and textual descriptions.

### 2.3.2 Use cases priority

1. 1
2. 2
3. 3
4. 4

### 2.3.3 Domain model

See APPENDIX.

### 2.3.4 User interface

The application will use a 3D graphical user interface that will be able to take into account almost all kinds of screen sizes.

## 2.4 References

# Appendix

## GUI

(Picture of GUI here)

## Domain model

(UML class diagram here)

## Use case model

(UML use case diagram here)

## Use cases

(Use cases texts here)