Barend “TheDjShinx” Noordhoff

Fight game (title WIP)

# Contents

[1 Contents 1](#_Toc150455385)

[2 Version Control 2](#_Toc150455386)

[3 Introduction 3](#_Toc150455387)

[4 Requirements 4](#_Toc150455388)

[4.1 Functional Requirements. 4](#_Toc150455389)

[4.2 Non-functional Requirements. 4](#_Toc150455390)

[4.2.1 Player requirements. 4](#_Toc150455391)

[4.2.2 Developer requirements 4](#_Toc150455392)

[5 Use Cases. 5](#_Toc150455393)

[6 Activity Diagrams. 6](#_Toc150455394)

[7 Testing methods 7](#_Toc150455395)

# Version Control

|  |  |  |
| --- | --- | --- |
| Version | Date | Changes |
| V0.1 | 11-9-2023 | Added document structure, and added some non-functional requirements. |
| V0.2 | 11-10-2023 | Added test methods. |

# Introduction

The goal of this document is to document requirements and Use Cases. In this document you can find the functional, non-functional requirements, brief formatted and fully dressed use cases that this game “Fight Game (name WIP)” is build upon. After reading this documentation it should be clear what functionality the game is build upon and how this works.

# Requirements

In this chapter both the functional and non-functional requirements are going to be described.

## Functional Requirements.

|  |  |
| --- | --- |
|  |  |
|  |  |

## Non-functional Requirements.

Non-Functional Requirements describe what an actor should be able to do to correctly function through the application.

Actor

* Player.
* Developer.

Ps, I put developer as an actor so I can document stuff that is only for me or other developers, if you don’t like that then I am sorry you feel that way :p.

### Player requirements.

As a player I want to be able to move my character using either WASD, a joystick, a d-pad or hitbox so that I can interact with the game.

As a player I want to be able to cast an attack using either assigned or custom button layouts so that I can deal damage.

As a player I want my attacks to stun my opponent so that I can make a combo using several different attacks.

As a player I want to be able to block damage with either an assigned button or custom button so that I can evade damage and give myself time to strategize.

As a player I want the game to end when my opponent’s health drops down to zero so that I can win the game.

### Developer requirements

# Use Cases.

# Testing methods

Personally, unit testing is the only method that is known to me to effectively test. But this is only effective for functions that receive data and put data out, which are functions I will be using in this project but overall, it isn’t what the entire project will consist of. Besides this I don’t want to rely too much on Unit tests as game development doesn’t benefit from them as much as OOP might and they can be a time wasteful pain to write right. So simple functions are the only thing I want to use for tests.

Unity has a reliable build in unit test environment, which Godot seems to lack for python which can be seen as a problem since that is what I’ll be using for Godot, but it doesn’t matter to me all that much. Since simple data in, data out functions will be the only one’s tested. So, I don’t need a big heavy unit test environment anyway, so either no or a built-in like Unity’s will suffice.

In conclusion, unit testing is the only way known to me to effectively/ systematically approach testing my code that isn’t just printing out values. But I also find unit testing to be way too much of a time waste in some cases, and these cases are the most prevalent in game development as far as I know. Which means that I will be unit testing but only for smaller functions that won’t eat up my time testing them. For the rest ill just be using the console/debugger to find the problems if there are any.

# Sources