A Software requirements specification

On

2D Platformer Game

Submitted in Partial Fulfillment for the Award of Degree of Bachelor of Technology in Computer Science and Engineering from Rajasthan Technical University, Kota

**

**MENTOR: SUBMITTED BY:**

**Mr. Kailash Soni Gaurav Biyani (17ESKCS723)**

(Dept. of Computer Science & Engineering)

**COORDINATOR:**

**Mr. Ankit Kumar**

(Dept. of Computer Science & Engineering)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



**SWAMI KESHWANAND INSTITUTE OF TECHNOLOGY, MANAGEMENT & GRAMOTHAN**

**Ramnagaria (Jagatpura), Jaipur – 302017**

SESSION 2021-22

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

4. System Features 3

4.1 Jumping 3

4.2 Death Zone 4

4.3 Title Screen ………………………………………………………………………………………..4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

# Introduction

## Purpose

This Software Requirements Specification document is intended to give a complete overview of 2D Platformer Game including the game mechanics, user interface, etc. The SRS document details all features upon which 2D platformer game have currently decided with reference to the manner and importance of their implementation.

## Document Conventions

There is a clear distinction between the use of words “player” and “character”. The “player” is the human being interacting with the game in the real world, while “character is the in game player avatar being manipulated. This document is printed on A4 paper in Times New Roman font. Normal text size is 12 black, while section heading are size 18 bolded black and subheadings are bolded size 14 black.

## Intended Audience and Reading Suggestions

The SRS document also gives testers and project managers a way to ensure the game's adherence to original vision Although the document may be read from front to back for a complete understanding of the project, it was written in section and hence can be read as such For an overview of the document and the project itself, see Overall Description (Section 2). For a detailed description of the game play elements and their interaction with the character, see System Features (Section 3). Readers interested in the game-play interface and navigation between different front-end menus should see External Interface Requirements (Section 4). Technical standards to which the team will hold the project are laid out in Other Nonfunctional Requirements (Section 5). The development schedule, meanwhile, will be maintained in the Key Milestones (Section 6).

## Product Scope

It is a simple 2D platformer game developed using Godot Game Engine which includes simple movement of player (forward, backward, jump), enemies (which can kill player and also can be killed by player if player jumps on them), gems (player can collect gems), to complete the level you need to collect all the gems in that level.

## References

* <https://docs.godotengine.org/>
* <https://godottutorials.pro/>
* <https://conceptartempire.com/godot-tutorials/>

# Overall Description

## Product Perspective

While the game itself is not a continuation of any predecessor, its implementation is made possible through the game development tool Godot Game Engine and its ability to compile and build projects is very efficient. Main goal is while designing, the game is to be able to let the user understand the gameplay at the first look no matter how often and how least he opens the game, this can be achieved by showing the control guide screen every time at the beginning of the game, hence engaging the user every time with the game. The game would not allow the users to pause the game once it started.

## Product Functions

The following is a summary of the major features implemented in the game. They are separated into categories based on those that are necessary for the game to function.

* **JUMPING:**

A necessary movement mechanic, allowing sudden movements upward from solid ground.

* **DEATH ZONES:**

Areas that trigger the restart of the level, be they encompassing stage boundaries or spike-like traps.

* **TITLE SCREEN:**

The first viewable screen upon starting up the application, containing buttons for play game, options.

* **END SCREEN:**

When the game ends the end screen will appear with a option to return to title screen.

## User Classes and Characteristics

The control scheme is designed to be intuitive and the game-play innovative and unfamiliar. Therefore, experience with games will not be a major factor in determining who is able to play: players of any age or skill level should be able to pick it up. However, as with any game with a large enough fan base, there will be a natural divide between casual and hardcore players. These two classes will naturally differentiate themselves through the use of the gameplay mechanics and the structure of the levels. The abilities and characteristics distinguishing hardcore players will include:

* **Collecting:** The ability to gems in order to win the game.
* **Killing:** The ability to kill enemies by jumping on them.

## Operating Environment

* Operating System : Windows
* Platform : Godot Game Engine

## Design and Implementation Constraints

* GUI is only in English.
* Game is only playable on Godot Engine.
* The title screen has the instructions to play the game.
* Gem counter will only increase if player enters in the collision shape of gem.

# External Interface Requirements

## User Interfaces

* GDScript (Godot Engine has its own built-in scripting language)
* Godot Game Engine
* Assets (player and enemy sprites, tile set for level design)

## Hardware Interfaces

* Windows supported device with version Xp or higher.
* A computer which supports Godot Engine.

## Software Interfaces

* Operating System - We have chosen Windows operating system for its best support and user-friendliness.
* Database – To save the details of job seeker and employer.
* Development End – HTML, CSS, JavaScript, Bootstrap, SQLite, Django.

# System Features

## Jumping

Being the secondary method of player input, jumping maintains the second highest game-play priority. This mechanic allows the player to navigate gaps without falling in death zone.

### Functional Requirements

**Req-1:** Levels should be adapted to this standard jump.

**Req-2:** Jumping should take into account the current vertical and horizontal velocities as well as the surface inclination.

**Req-3:** Jump force should be large enough that the mechanic is not made useless but not so large that gravity can easily replace the effect.

## Death Zones

Death zones are the opposite of level completion zones in that when touched, they send the character back to the starting point of the level. Some death zones take on the form of environmental obstacles. Death zones are a necessary replacement for bottomless pits, which prove impossible to implement given the changeling nature of gravity. These zones increase the game's difficulty and incentivize good performance via negative reinforcement. They should be prioritized in level design as they bring the game-play from "playable" to "enjoyable."

### Stimulus/Response Sequences:

**Step 1:** The character comes into contact with a death zone.

**Step 2:** The character is immediately returned to the beginning of the level

### Functional Requirements:

**Req-1:** No death zone should be placed so as to make its respective level impossible to complete.

## Title Screen

The title screen is the screen the player will see every time upon playing the game. Through this interface, the player can start the game and also can see the game instruction to play.

### Functional Requirements:

**Req-1:** The title screen must load and appear every time the game is launched.

**Req-2:** If the player quits the game during any stage of a level, they must be returned to the title screen

**Req-3:** If the player presses the exit button, the game will end and return the player to the phone's regular interface.

**Req-4:** If the player completes the game, the game will end and return the player to the title screen.

# Other Nonfunctional Requirements

## Performance Requirements

Based on the capabilities of current system, performance should not be an issue. However, system with weaker hardware may incur some difficulties and potentially run slower. The game design will be tailored in order to give an enjoyable experience on all system, regardless of hardware. The functionality of the game will be simplistic enough but not trivial, and the graphics will not be overly detailed so the system does not become slow down.

## Safety Requirements

This game will not affect or damage any of the other applications installed on the system. The game will also not cause any overheating of the system; therefore, the computer's internal components will not be damaged. This game should not be played when the player's attention is divided among multiple tasks to prevent potential harm to the player.

## Security Requirements

This game will not ask for any personal information from the player and will thus be unable to compromise such information. There is no player authentication required to play the game. The player simply has to download the application in order to start playing this game. That being said, anyone who has access to the player's computer will have the ability to play the game. If any unauthorized player acquires the original player's computer, that unauthorized player will be able to play the game. It is the responsibility of the player to make sure that no unauthorized player/ person will have access to his or her phone.

## Software Quality Attributes

To ensure reliability and correctness, the game will respond to the player's commands in a timely manner. When the player makes the character jump the gravity and velocity components would response immediately, the player should see the effects of this command within milliseconds not 10 seconds after the effect. For adaptability and flexibility, this game will not save the player's progress after every level completion that way, in the event that the player's system crashed during game-play, the player cannot resume game-play at a reasonable starting point.

In terms of usability, the graphical user interface will be very intuitive for the player to use. The beginning levels of the game will also slowly introduce each of the unique commands available to the player as well as any other game mechanics the player may need to know to complete the level in more complicated and elaborate level, these introductions and hints will not be available because the player will have already been introduced to all the tools needed to complete the level. This method will ensure that the player gradually learns all the game mechanics and also enjoys a challenging game-play experience. Our game focuses on both ease of use and ease of learning while not leaning towards one or the other. This should be a game that any player can pick up and play instantly without much down time in figuring out the controls. The controls will be easy and intuitive to use and the player will not be bombarded by having to use all the controls at once in the beginning levels. The game play will ease the player into learning and using each of the commands by gradually introducing each command as the player progresses through the game and completes each level. By the end of any given level the player should be familiar with and fully able to use the command introduced in that level and all the commands introduced in previous levels.