Software Requirements Specification

for

The AntiDote

Version 1.0

Prepared by Team “The AntiDote”

Deshaboina Yashwanth

Chandrima M J

Abhijith S Babu

Lakshin Kumar R

Srivatsan H

Alaka Jayan

E Sai Tejashwin

Indian Institute of Information Technology Kottayam

Kerala

26-09-2021

Table of Content

Revision History

Name	Date	Reason For Changes	Version
Software Requirements Specification for The AntiDote, A 3D Shooter Game.	26-09-2021	Initial SRS Document	1.0

1. Introduction

1.1. Purpose

This Software Requirement Specifications document is intended to give details about the features of our game, The AntiDote. The document will provide details of the user interface, the game mechanics, dynamics and any other dependencies it requires. It is hoped that this document works as a checklist for us to implement all the features of our project in the order specified in this document.

1.2. Document Conventions

• Priorities for higher-level requirements are inherited by detailed requirements

• Every requirement statement has its own priority.

1.3. Intended Audience and Reading Suggestions

Programmers, Testers and Designers of the project. Software Management Professor.

Reading Suggestions:

This document is intended for developers, project managers, users, testers, and documentation writers.

The next section (Section 2) provides an overall description of the game which includes the product perspective, its functions, user classes along with its characteristics, the operating environment, design and implementation constraints, user documentation, and assumptions and dependencies of the game.

Section 3 gives information on the external interface requirements of the game. This includes user interfaces, hardware interfaces, software interfaces and communication interfaces.

System features and their functional requirements are specified in Section 4. All non-functional requirements regarding safety, security, quality and business attributes of the game are specified in Section 5.

Other additional requirements are given in Section 6.

All the formal terminologies used in this SRS document are defined in Appendix A.

1.4. Product Scope

The goal of the project is to create a game falling in the zombie-niche genre with all its elements intact. The player when playing our game is expected to feel a mix of thrill, awe, and fear. We achieve this with the help of some carefully designed mechanics, environments, and agents.

The mechanics we use to make the game relevant are the inclusion of vaccine awareness and zombies who hold the ability to harm the player, possibly affecting their health in a substantial way, the numbers of which increase gradually as time passes. The first vaccines of a very hostile, mutated virus have failed and have caused side-effects making people extremely aggressive, throwing the world into chaos. Three years later, the surviving and healthy researchers have succeeded in creating an antidote and want you, a mercenary, to deliver the antidote to an important place for distribution. The player must try to overcome the dangerous world with the sedative-weapons given by the army, to immobilize and escape from the mutated humans.

1.5. References

This Software Requirements Specification is prepared by taking the references of Library Management System (LMS) prepared by Dr. Divya Sindhu Lekha at Indian Institute of Information Technology Kottayam.

Secondly, we used the sample SRS Document Shared by Dr. Divya Sindhu Lekha at Indian Institute of Information Technology Kottayam.

Link: https://lmsone.iiitkottayam.ac.in/mod/assign/view.php?id=1719

2. Overall description

2.1 Product perspective

We're developing a 3D survival, first-person shooter that is meant to be played by a single player with future plans to improve it and possibly improve functionally and mechanics over time. This is a stand-alone game and it is not a continuation of previous games. This game is to be played on PCs and uses its peripherals for input and outputs. It requires certain hardware to run smoothly. The game has an easily understandable UI and gameplay and decent level design for the player to know the ins and outs of the game.

2.2 Product functions

This being a survival, the player is represented in the game by a small character who can be controlled and be able to move through the map.

The following functions have been implemented in the game:

• Movement:

The player is a 3D Character who can be controlled via the keyboard and can be moved up, left, down and right using WASD movement keys and can jump using the Spacebar Key and Crouch using Left Control key.

• Power ups:

As the player traverses through the map, he/she can collect various bonuses and special powers hidden throughout the game. After each round, they can be collected and will be added to the player's inventory.

• HP (Health Points):

The character has certain HP or health points up to which he/she can take a certain amount of damage from enemies before dying or losing their life.

• Title screen:

The first viewable screen upon starting the application where the player can view/change the game options and see their collection of treasures.

• Pause screen;

The player can pause the game at any time and will be greeted with this screen showing options to control the game.

• Audio:

The game also includes audio using the unreal Audio engine using audio-editing software like Audacity to go along with the visuals to provide immersion for the player.

• Enemies: The game also has monsters who will be prowling along the map waiting to harm anyone who crosses paths with them.

2.3 User classes and characteristics:

Our control scheme is designed to be intuitive, and our gameplay innovative and unfamiliar. Therefore, experience with games will not be a significant factor in determining who can play; players of any skill level should be able to pick it up.

However, as with any game with a large enough fanbase, there will be a natural divide between casual and hardcore players. These two classes will naturally differentiate themselves by using the gameplay mechanics and their knowledge of the game.

The abilities and characteristics that distinguish a hardcore player are:

• The knowledge of the map and its major hot spots.

• Ability to shoot Enemies with the AntiDote Vials using a Gun.

• Plan escape accordingly without getting infected with the virus.

• Upon Spawning you get a access to a single basic weapon. Multiple weapons or Loadouts will be considered while building through out the project and once added will be set as an element of game progression.

2.4 Operating environment:

The game will be Operable on x86- 64-bit operating systems (for windows it's Win64) Environment only. The game can run on pretty much any computer having Windows OS and does not require much in terms of hardware. This game will not run on mobile phones.

Minimum hardware requirements:

• A dual-core CPU

• At least 4 GB of RAM

• 8-10 GB HDD space

• 2GB GPU performance

• At least 1 GB of visual memory

2.5 Design and implementation constraints:

There are very few limitations in terms of design and implementation in this game. Major constraints will be the size of the game and its visuals. Since we want the game to be playable on any machine, we have to limit the size and amount of GPU intensive graphics and animations. Secondly, the space occupied on the HDD should not exceed a lot more than the minimum requirement as we want everyone to experience this game to the fullest.

2.6 User documentation:

Once the game is installed, the user will be offered a user manual which the player can refer to understand the game mechanics and controls. The user manual will be a PDF with all the details about the game. There will also be a credit note containing the name of the creators and their contributions to the game. A document containing a set of terms and conditions will also be provided.

2.7 Assumptions and dependencies:

We assume that the user has a machine that meets the minimum requirements:

4 GB RAM

8-10 GB Disk Space

2 GB GPU performance

1 GB VRAM (Video memory)

Operable on x86 - 64-bit operating systems (for windows it's Win64)

The unreal engine launcher must be installed in the PC.We can assume that all the intended features and functions work properly across all operating systems without any hitch. We also assume that the user does not tamper with the product to customise anything or to pirate the game.

3. External interface requirements

3.1 User interface

• Upon starting the user will be greeted with the title screen where they will have options to start the game, change settings and view the profile all of which can be done using the mouse.

• Once inside the game the users can use the keyboard and mouse to play it.

• The user can pause the game at any time, which will be a pop-up window with buttons to resume, restart, settings or quit. All of these can be accessed using the mouse.

3.2 Hardware Interfaces

Any PC that supports Winx32 and Winx64 will be able to run this program.

3.3 Software Interfaces

The game uses C++ and Microsoft DirectX graphics API to generate the interface to run the game during the execution. The program utilises the inbuilt game libraries made by Epic Games for Unreal Engine 4 in C++ and can be found in the Unreal Engine API Documentation.

3.4 Communication Interfaces

Since it is a single player game, it will not interact with any other interfaces other than Unreal Engine 4 plugins.

4. System features

4.1 Starting a new game:

4.1.1 Description and priority:

Upon starting the game for the first time, the user is required to create a new game to start playing. This is a high priority.

4.1.2 Stimulus/Response sequences:

"New Game" Button click: The game starts a new game and prompts the

user to enter their name.

"Select Avatar" Button click: User can select their avatar from any of the

pre-existing avatars.

"Submit" Button click: New game will be created with the name provided

and will be forwarded to the next screen.

4.1.3 Functional requirements:

Function: Starts a new game.

Inputs: The name of the user as text and the avatar.

Source: From the field on the screen.

Outputs: Starts a new game with the name of the user.

Action: User input is validated, and a new game is started.

4.2 Continuing the game:

4.2.1 Description and priority:

Once the user has created a game, they can continue playing it by selecting it in the title screen. This is a high priority.

4.2.2 Stimulus/Response sequences:

"Start Game" Button click: The game continues from the last saved point.

4.2.3 Functional requirements:

Function: To continue the game.
Description: Allows users to keep playing the game from the last point.
Inputs: Click the Start Game button.
Source: From the title screen.
Output: The game starts.

Function: InitialiseGameEvent [New Game]
Description: Initialise the game by assigning the player's position, health, destination, etc.

Function: PauseGameEvent [Pause Game]
Description: Allows the user to pause the game
Input: Click the Pause Game button.
Source: From the screen.
Output: The game pauses.

Function: Spawn PlayerController/Spawn Actor

Description: The player will be spawned.

Function: Actor Physics(At Event TickRate)
Description: Allows the player to walk/run in different directions.

Input: A	Output: Left
Input: S	Output: Back
Input: W	Output: Front
Input: D	Output: Right
Input: <spacebar key>	Output: Jump
Input: Left Ctrl key	Output: Crouch

[NOTE: the user inputs here are the common controls used in many games and has been a placeholder for our game as well, adjustments could be made in the future according to feedback from playtesting].

4.3 Game settings:

4.3.1 Description and Priority:

The user has the option to change the settings of the game to suit

their preferences. This is a low priority.

4.3.2 Stimulus/Response sequences:

“Settings” Button click: The game forwards to the settings screen.

“Master Volume” Button click: Gives option to adjust game audio volume.

“SFX Volume” Button click: Gives option to adjust special effects audio volume.

“Key Binds” Button click: Gives option to customise the key controls.

“Quit” Button click: Directs back to title screen.

4.3.3 Functional requirements:

Function: Change settings for the game.

Description: Allows users to change the settings of the game.

Inputs: Click the settings button.

Click the audio button. [Slider Bar] [Seperate for Master Volume and Music]

Source: From the settings screen.

Output: The game applies the settings selected by the user.

Action: Adjusts the audio and keybinding settings according to the user's preference.

Pre-condition: The game has default settings applied.

Post-condition: The new settings are applied.

4.4 Gameplay:

4.4.1 Description and priority:

The user can now play the game. They can use the keyboard to control the Player inside the game. This is a high priority.

4.4.2 Stimulus/Response sequences:

"W, A, S, D" keyboard strokes: Moves the Player Run forward, left, backward, right respectively.

Right click the mouse to shoot The AntiDote vials towards enemies.

4.4.3 Functional requirements:

Function: Player movement inside the game.

Input: Keyboard strokes and mouse clicks.

Source: From the game play screen.

Output: Movement of Player.

Action: The game allows the Player to move freely inside the map and shoot the enemies.

4.5 Pause Menu:

4.5.1 Description and priority:

The user can pause the game anytime they're playing using the Esc key. This is a medium priority.

4.5.2 Stimulus/Response sequences:

"Esc" key press: The game pauses.

"Resume" Button click: The game resumes.

"Retry" Button click: The game restarts from the previous checkpoint(if any).

"Settings" Button click: Forwards to the settings screen.

"Quit" Button click: The game quits the level and the player is returned to the main screen.

4.5.3 Functional requirements:

Function: Pauses the game.

Description: The user can pause the game to change settings, retry or to quit the game.

Inputs: Keyboard stroke.

Source: Anytime during the game play.

Output: The game pauses.

Action: The game pauses to give the users some options.

4.6 Enemy Spawning:

4.6.1 Description and priority:

The game spawns an enemy around the map that will harm the actor when in contact range. This is a high priority.

4.6.2 Stimulus/Response sequences:

“<Level>” Level Sequences : The game loads the level and it starts to spawn enemies.

4.6.3 Functional requirements:

Function: To harm the user actor.

Description: This enemy tries to stop the actor from completing the level by harming it.

Inputs: Click the level button.

Output: The actor may/may not get killed.

4.7 Quit menu:

4.7.1 Description and priority:

The user can quit the game using this menu. This is medium priority.

4.7.2 Stimulus/Response sequences:

“Quit” Button click: Shows a pop-up confirming quit.

“Yes” Button click: The game quits to desktop.

“No” Button click: Directed back towards the title screen.

4.7.3 Functional requirements:

Function: The game quits to desktop.

Description: The user can quit the game anytime by clicking this button.

Inputs: Click the quit button.

Click the yes button.

Click the no button.

Sources: From the title screen.

Output: The game either quits or directs back to title screen.

5. Other Non-Functional Requirements:

5.1 Performance requirements:

Based on the capabilities of the current personal computers, performance should not be an issue. However, computers 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 computers, regardless of its hardware specification. The functionality of the game will be simplistic enough, but not trivial, and graphics will not be overly detailed so that the system does not slow down.

5.2 Safety requirements:

• The game will not affect or damage other programs installed on the users' computers.

• In the event of the game crashing, the game will load from the previous saved data without any errors.

• It will provide error handling methods to support critical functions.

• Termination of the game will result in a safe system state.

5.3 Security requirements:

The game will not ask for any personal information from the user (the name asked might not necessarily be their real name, any name is accepted) and will thus be unable to compromise such information. There is no user authentication required to play the game. The user simply has to install the game in order to start playing. It is the responsibility of the user to prevent any unauthorized use of their computer.

5.4 Software Quality Attributes:

• To ensure reliability and correctness, the game will respond to the user's commands in a timely manner. When the user makes a keyboard stroke or a mouse click, the game should respond instantaneously.

• For adaptability and flexibility, the game will automatically save the user's progress after every level completion. That way, in the event of game crashing or accidental termination, the user can resume game play at a reasonable starting point.

• In terms of usability, the GUI will be very intuitive for the player to use. The beginning level of the game will also slowly introduce each of the unique commands available to the user as well as any other game mechanics that the user needs to know to complete the level. In later levels there will be no introductions or hints for the game play. This method ensures that the player gradually learns the game mechanics and also enjoys a challenging game play experience. Our game focuses on both ease of learning and ease of use, without leaning towards one or the other.

• For performance, the game will perform very well on almost all of the computers that have Windows operating system. It may suffer with computers that may not have a dedicated video card.

• The game will be free to use. This ensures availability to all the users.

• For portability, once the game has been downloaded, it can be copied into any storage device and transported.

5.5 Business rules:

• The user may not tamper with the code of the game for their own benefit.

• Pirating the game is not allowed.

• The users should download the game only from reliable sources.

6. Other requirements:

There is no additional requirement.

Appendix A: Glossary

PC: Personal Computer

GUI: Graphical User Interface

SFX: Sound Effects

GPU: Graphics Processing Unit

VRAM: Virtual RAM

API: Application Program Interface

HDD: Hard Disk Drive

HP: Health Points

Appendix B: To Be Determined List

1. Different maps for levels.

2. Different player avatar selection images.

3. Monster spawn locations and pathfinding.