Reign

Lewis Brown

CST-451 Capstone Project Requirements Document

Grand Canyon University

Instructor: Professor Mark Reha

Revision: 1

Date: 10/4/2019

**ABSTRACT**

Reign is a round based hack and slash video game where the Player will fight rounds of enemies leading up to their inevitable demise. The player will defeat enemies and can pick up weapons and other shields. The player will have the ability to traverse the map, attack enemies, and block enemies attack. Once a round starts multiple enemies will come after the player and once that player has defeated all enemies on the map the round ends and the next round will start. At the start of round 1 the player will start off with a weak sword and a weak shield. In later rounds the enemies will spawn with better weapons and shields. If a player defeats one of those enemies the player will be able to pick up those weapons and use on other enemies. The player will also be able to hold multiple weapons. The player has multiple ways of avoiding attacks such as dodging, running, and blocking attacks.

The work that has been done so far is a 3d models of weapons, and a couple 3d models of shields. The Pause Menu and The Main Menu are also created.

|  |
| --- |
| History and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Revision Notes** |
|  |  | Initial draft for review/discussion |
|  |  |  |
|  |  |  |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

**Integrated Instructor Feedback into Project Documentation**

Yes  No

**TABLE OF CONTENTS**

Functional Requirements 4

Non-Functional Requirements 5

Technical Requirements 6

Logical System Design 7

User Interface Design 8

Reports Design 9

Functional Requirements

**Use Cases**

Use Case 1: Main Menu selection:

Main Flow:

1. The user will move their mouse over play game and left click to start the game
2. The user will move their mouse over Exit game and left click to close the application

Use Case 2: Game Play:

Main Flow:

1. The user can walk forward by pressing and holding down the ‘W’ key
2. The user can walk backward by pressing the and holding down the ‘S’ key
3. The user can walk to the left by pressing the and holding down the ‘A’ key
4. The user can walk to the right by pressing the and holding down the ‘D’ key
5. The user can attack the enemy by pressing the left mouse button
6. The user can block an enemy attack by pressing the Right mouse button
7. The user can jump by pressing the Space Bar
8. The user can run by pressing and holding down the Shift key and holding down the ‘W’ key
9. The user can crouch by pressing the Ctrl key
10. The user can look around by using the mouse
11. The user can pick up a new weapon from a downed enemy by pressing the ‘E’ key while looking at the weapon
12. The user can pick up a new shield from a downed enemy by pressing the ‘E’ key while looking at the shield
13. The user can hold up to 2 weapons
14. The user can switch to primary weapon by pressing the ‘1’ key
15. The user can switch to secondary weapon by pressing the ‘2’ key
16. The user can switch weapons by using the scroll wheel
17. The user can bring up the pause menu by pressing the ‘Esc’ key
18. The Engine will handle the enemies attacking the user
19. The Engine will handle the enemies blocking the users attack
20. The Engine will take the user to the next round if all enemies are defeated
21. If the player is dead the Engine will bring up the game over menu
22. The Engine will handle collision detection so that the user and enemies will not run through solid objects
23. The Engine handles damage calculation so that the user or enemies will slowly take damage after a direct hit see how much damage the enemy or play took then take it out of their health
24. The engine will handle enemy intelligence by having them know the player location and detecting obstacles, so they don’t run into them
25. The Engine will handle Collision detection on enemies and players so the cannot run through each other
26. The engine will handle collision detection for the floor so that the user or enemies wont fall through the floor

Use Case 3: Pause Menu:

Main Flow:

1. The user will go over with their mouse and left click on the controls to bring up the controls menu
2. The user can select continue game by going over it with their mouse to get out of the pause menu and continue playing the game
3. The user can select Exit Game by going over it with the mouse and that will take them back to the main menu
4. The user can also press the ‘Esc’ key to exit out of the pause menu and continue playing the game.

Use Case 4: Controls Menu:

Main Flow:

1. The user can view all the controls in the game
2. The user can go over ‘return’ with the mouse and left click to go back to the pause menu

Use Case 5: Game Over Menu:

Main Menu:

1. The user can go over with their mouse and selecting ‘Try Again?’ to start the game over from the beginning
2. The user can go over with their mouse and selecting ‘Exit Game’ to go back to the main menu selection screen

Non-Functional Requirements

Performance:

* The Engine will need to load all game assets into memory in less than a minute. This is the loading time to get into the game as fast as possible
* The game must run at 60 frames per second to make the game play as smooth as possible
* Low-end standard definition graphics so that the game can run on most low-end computers
* The game should not use more than 3 GB of RAM while it is running
* All damage calculations must happen as soon as the player or enemy gets hit
* The game must work on at least a dual core CPU
* The game must work on a graphics card that uses 2 GB of VRAM

Technical Requirements

Unity: 2018.4.8f1

The Unity game engine is capable of rendering full 3D environments, handling complex animations, and uses C# as a programming language. Also has a store for various 3d models. This engine will be responsible for the following task:

* Rendering the 3d world the player will walk on
* Rendering the character that the player will play as
* Rendering the Enemies
* Rendering the equipment
* Mapping actions to buttons such as walking around by using the WASD keys
* Collision of objects and player
* Handling all in game rules

Visual studio 2019: 16.2

Microsoft’s IDE that is able to create classes that are written in C#. It also supports integration to Unity. This IDE will be used to create these components:

- Will be used for going from round to round

- Used for combat such as hit points and damage

- Calculate damage

- Basic ai for the enemies

- Collision between two objects

- Calculate how many enemies are on the map alive/defeated

Blender: 2.8

Blender is a 3dmodeling software that does 3d modeling and animation. Blender such as unity has its own 3d model store that may be used for this project, but 3d models will most likely be self-made. Animation taken form the store my include models of equipment and character prefabs. Blender will be used to complete these tasks:

- Walking animation

- Attacking animation

- Blocking animation

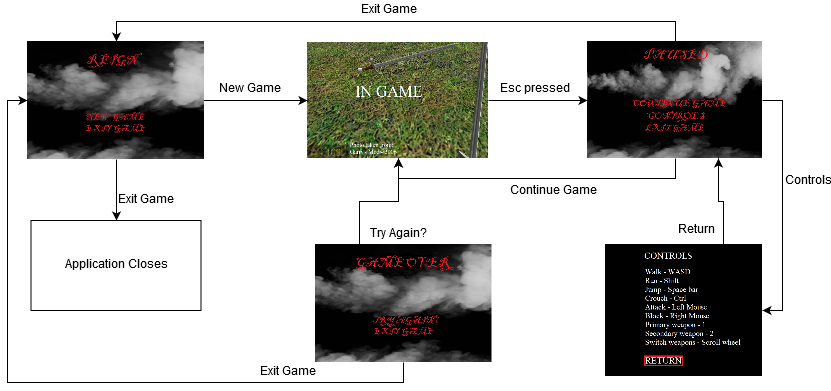
- 3d models of equipment such as sword, and shield

- 3d rendering of the entire world the player can explore

- 3d model of character the player plays as

- 3d model of Enemies

Logical System Design

****

User Interface Design

Main Menu:

New game - starts the game

Exit game – closes the application



Pause menu:

Continue game – continues the game

Controls – brings up the in-game controls

Exit game – brings you back to the main menu



Controls Menu:

Return – brings you back to the pause menu



Game Over Menu:

Try Again? - starts the game over from the beginning

Exit Game - takes you to the main menu



3D models in game

