**Catch! VR Design Doc**

Jason Captor

**Game Design**

**Game Concept**

*[Explain your Game concept in detail. Give a clear description of what the game will be including genre and perspective, what the game mechanics/core loop will be, and with an explanation of the story/setting]*

Catch! VR (referred to as Catch) is a game where the player is tasked with defending themself from an endless supply of falling weapons, using a pair of swords to deflect the incoming weapons. As the falling weapons are endless, the game only ends when the player runs out of health, but the player can regain health if a potion hits them. Players increase their score by deflecting weapons or potions away. Once the game ends, their score is displayed.

**Assess the Brief**

*[Go through the game project brief point by point and explain how this game concept will address these points]*

**Features**

* *Game has a Main Menu with Play button and Quit button, as a minimum*

On starting the game, players will be shown a menu with the game title, and buttons to start, quit or set the difficulty of the game.

* *The Game has a pause menu functionality available during gameplay*

The game will feature a wrist-mounted pause menu accessible via the menu button on their controller(s). Players will select options using either a joystick or by pointing with their other hand.

* *The Game features an end screen with the option to return to the Main Menu or play again*

The game will feature a pedestal which, during gameplay will track score and health, and once the game ends will show score and buttons to return to main menu or play again.

* *Game must build and run successfully for 3 different VR headsets*

Catch will export to HP Reverb G2, Oculus Quest 2 and Oculus Rift S.

**Gameplay**

* *Objects fall from above the player, the player has a weapon or implement that destroys these objects on collision, and this weapon follows the tracking of one of the controllers so that, as an example, the player could swing around a sword and cut fruit falling from the sky.*

Weapons will fall from a random point within a spawn ‘ring’ above and slightly in front of the player, the player will have two swords paired to their controllers to deflect incoming weapons, but rather than destroying the falling objects, the weapons will bounce away harmlessly.

* *Each item destroyed gives the player score*

Each item deflected increases the player’s score.

* *If one of these objects hits the ground, the player loses 'health' or some other attribute, and if this attribute reaches 0, then the game is over and the player receives their score on an end screen*

When objects hit the ground directly below the spawn ring, players lose health. If objects hit the ground around the player after being deflected, the player gains points.

**Aesthetics**

* *The game must be made with a consistent visual style and use assets that are not simply basic unity engine objects (no grey boxes!). What we're looking for: a consistent theme to the visuals. What we're not looking for: AAA level fidelity. Just have a theme but don't make it a burden on yourself.*

The game will be styled fairly simplistically to match the level of gameplay, with models and textures for the falling weapons, potions and the player’s swords. There will also be a simple pedestal nearby for displaying UI elements in world space. The walls and floor will also be textured in a simple, non-distracting way.

* *The game must feature some audio at least for the gameplay activities (sound effects for breaking objects and losing 'health') but does not strictly need to extend to the GUI (menu sounds)*

The game will trigger a sound effect when players knock away objects (a ‘clang’ of sword on sword), a sound effect for weapons hitting the ‘score zone’ (a shatter or crash), and a sound for potions hitting the ‘score zone’ (a bubbling, health gained sound).

**Platform and Documentation**

**Platform**

*[The Game you are making must be released on at least 3 different VR/AR platforms. Decide on the Development and Release platforms. What engine will you be developing the game with? What VR/AR platforms will you be releasing on? Please explain your decision.*

*Ensure the Platforms chosen meets the following requirements: Supports multi-platform development, Has built-in GUI support]*

I will be creating my game in Unity as it is one of the simplest 3D engines to develop for, and has built-in support for VR and GUI. Larger scale engines like Unreal would be too demanding for some systems, and smaller engines like GameMaker do not have the tools for VR support.

I plan to launch this game on the HP Reverb G2, the Oculus Quest 2, and Oculus Rift S. I chose these 3 as the HP Reverb is what I will primarily be testing in, and the others are common and popular VR headsets which should give me a broad audience without imposing too many hardware-related restrictions. All 3 of these units are compatible with paired hand controllers which will be required to play this game.

**Documentation**

*[What software/approach will you use to document the game and it's planning, development, testing, fixing, and review?*

*Why have you chosen these programs/solutions for these purposes?]*

I will be using Microsoft Word for writing my planning documents, testing forms and review documents, as it is broadly accessible, and this will become especially important once I start sending out testing forms.

I will be using GitHub to track changes made to the project over time, and for version control.

In terms of documenting development, I will be writing my documentation in 4 main stages: An initial, planning document setting out the goals and processes to design the game (this one), an in-progress stage with documents tracking asset attribution and bugs, a testing and deployment stage, and finally a review stage once the game is complete, to reflect on the project as a whole.

**GUI Design**

*[Provide a visual mockup and demonstration of the planned graphical user interface – the in-game visuals that will be used to convey important details to the player such as health, current ammunition and resource levels, etc. Please explain what each element will do]*

**Main Menu:**



Start button takes you directly to the next scene, the main game.

Options button opens the options menu (shown on next page).

Quit button closes the application.

**Options Menu:**

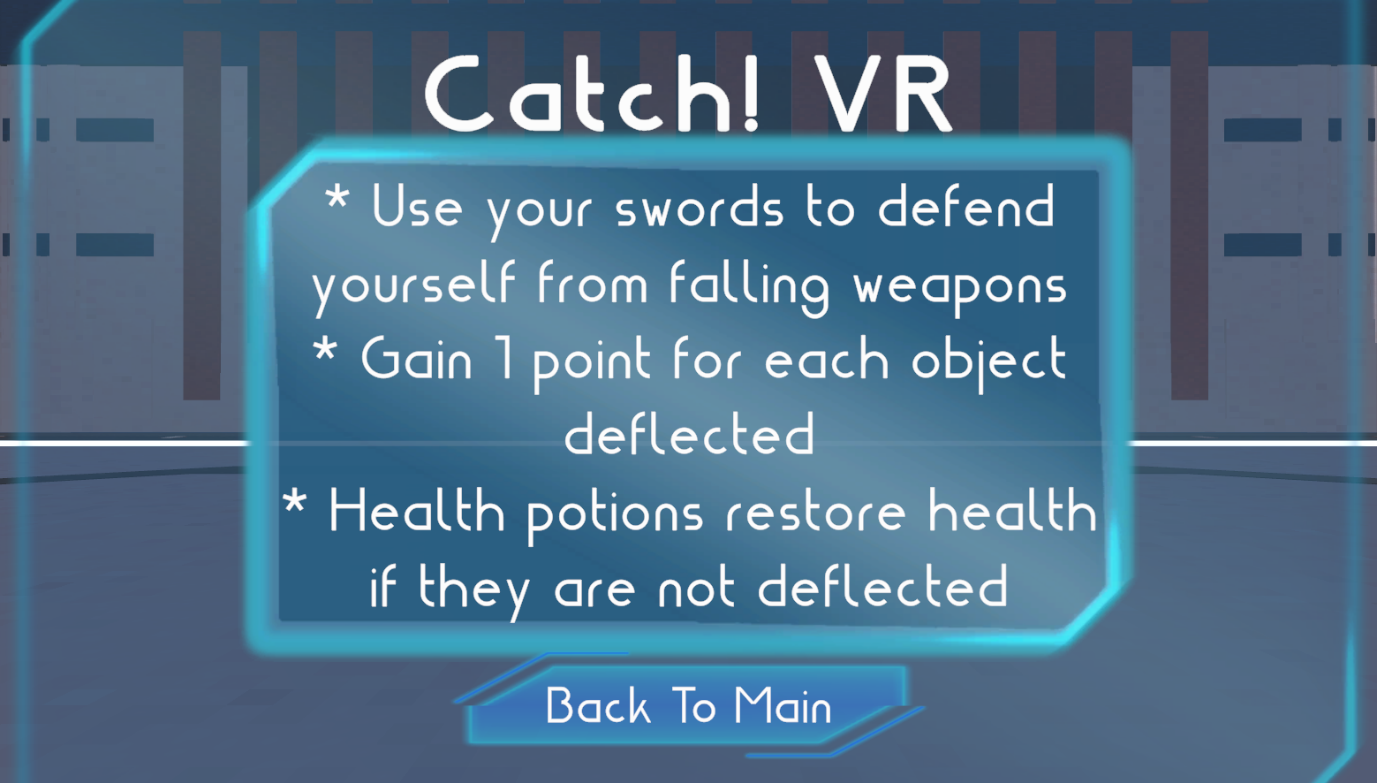
Volume slider controls the volume of sound effects in the game (shown at maximum volume).

Brightness slider adjusts the brightness of the game (shown at maximum brightness).

Resolution selector allows users to choose the resolution the game is displayed at.

How To Play button takes you to the tutorial screen (shown below).

Back To Main button closes the Options menu and takes you back to the main menu (previous).

**Tutorial Screen:** 

Back To Main button closes the tutorial screen and takes you back to the main menu.

**In-Game UI:** 

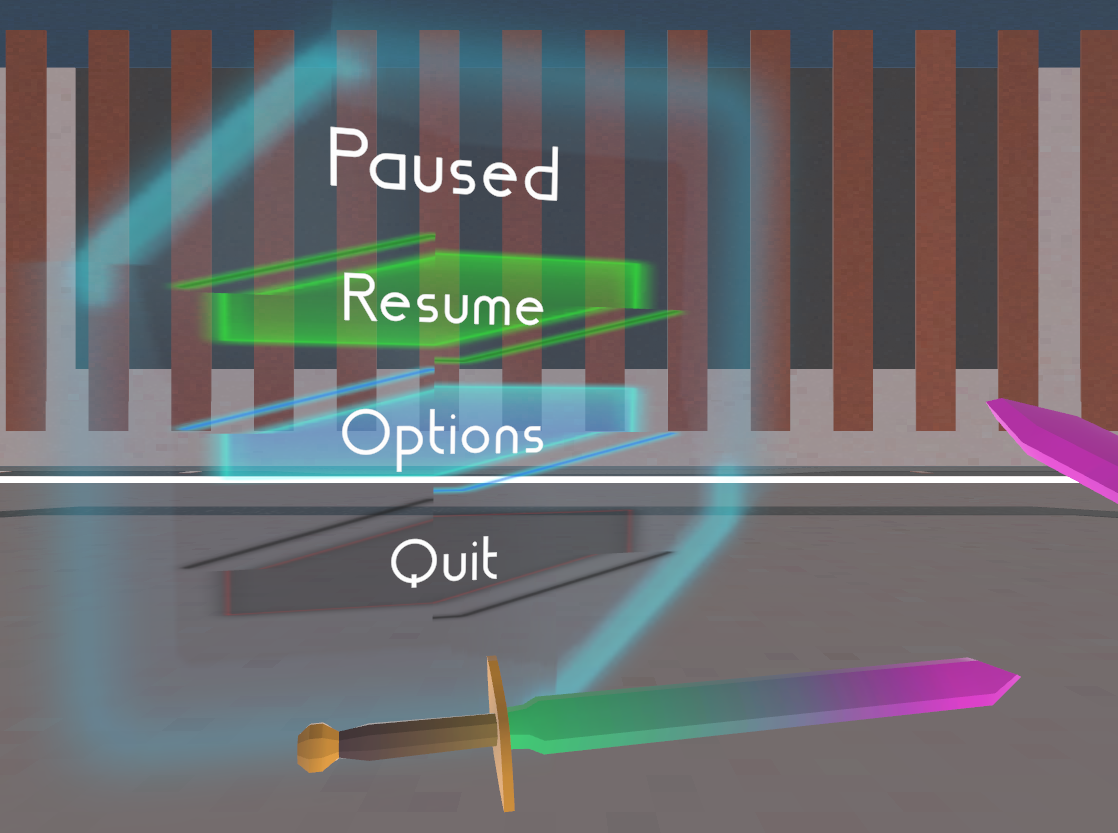
Health and Score update dynamically throughout gameplay but are not directly interactable, and when health reaches zero, this display switches to the ‘game over’ screen.

**Game Over Menu:**

Score display shows your final score (non-interactable).

Main Menu button takes you back to the previous scene with the main menu.

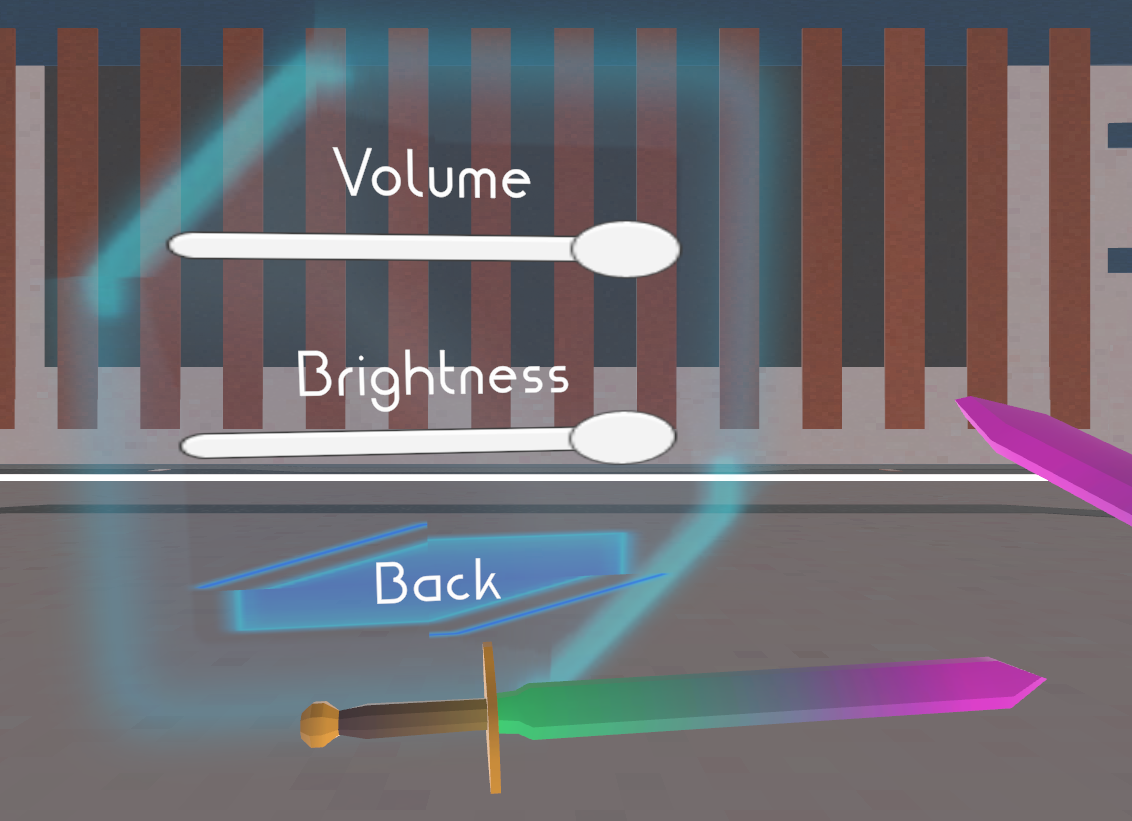
Quit button closes the application.

**Pause Menu:** 

Resume button closes the pause menu and resumes the game.

Options button opens the in-game options menu (below).

Quit button takes you back to the main menu.

**In-Game Options Menu:**

Volume and Brightness sliders function identically to those in the main options menu.

Back button closes the options menu and re-opens the pause menu (above).

**Production Plan**

*[Go through the important components of the game and what tasks will need to be completed in order to make the game. Create a timeline for completing these tasks in whatever format works for you]*

| **Task** | **ETA** |
| --- | --- |
| Design gameplay loop | 18/08/2023 |
| Design main menu | 25/08/2023 |
| Design end menu | 31/08/2023 |
| Implement gameplay loop  Design pause menu | 01/09/2023 |
| Retrieve and implement visual assets  Complete initial documentation | 08/09/2023 |
| Finalise main gameplay loop  Implement audio assets | 15/09/2023 |
| Implement menus | 22/09/2023 |
| Debugging | 03/11/2023 |