

# Technical Design Document

## VRCADE

## **Contents**

[Coding Standards / Style Guides](#)

[Variables](#)

[Methods](#)

[XR Rig Configuration / VR Player](#)

[Game Flow Diagrams](#)

[Login Scene](#)

[Home](#)

[Lobby](#)

[Air Hockey](#)

[Basketball](#)

[Archery](#)

[Server Communication \(Score, Credit, Time tracking\)](#)

[Unity Web Requests](#)

[Add Credit UI](#)

[PHP Server - Rest API](#)

[Custom Teleporter](#)

[Login System](#)

[Home scene selection menu](#)

[Avatar Selection](#)

[Scoreboards](#)

# Coding Standards / Style Guides

Reference:

- CamelCase
- lowerCamelCase
- snake\_case

## Variables

Standard variables will all use **lowerCamelCase**

Example:

```
string randomRoomName = "Room_" + Random.Range(0, 10000);
```

With the exception of if there are multiple similar variables like the example below. Here we will use an underscore to clearly differentiate the variable.

Example:

```
string randomValue_Lobby  
string randomValue_AirHockey  
string randomValue_Basketball  
string randomValue_Archery
```

## Methods

Methods are differentiated by using **CamelCase**

```
public void OnEnterRoomButtonClicked()  
{  
    // code here  
}
```

For more information on C# Style Guides please visit:

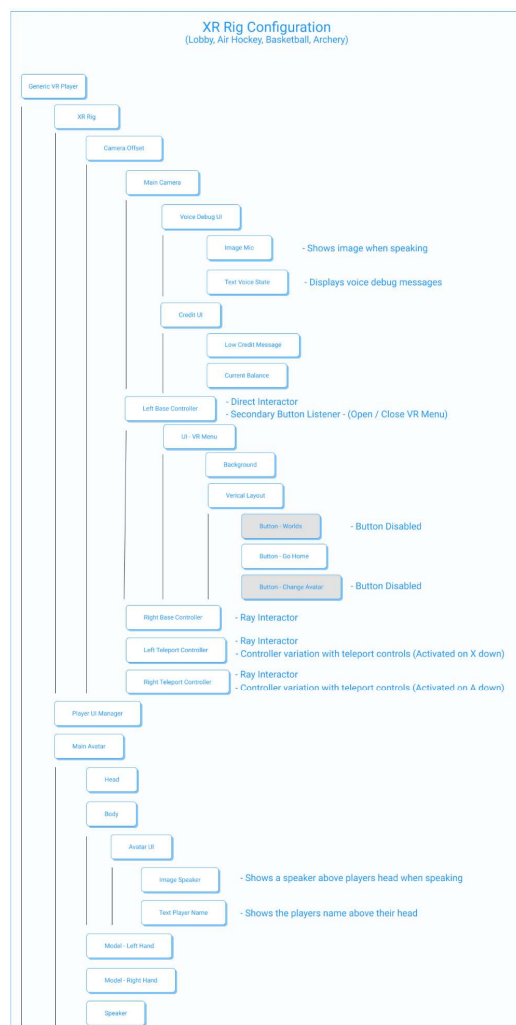
<https://google.github.io/styleguide/csharp-style.html>

# XR Rig Configuration / VR Player

The Network VR player (shown below) is a standard device based XR Rig with additional features added to allow us to use both Direct and Ray based interactions along with showing various UI elements and menus.

The diagram below shows the hierarchy of the standard Network VR Player.

The Login and Home scene both contain variants of this original model and the oldy difference is that the unnecessary components are turned off for these scenes.



[https://drive.google.com/file/d/1lbpr0gV6\\_sMH7lplqeQImFhcD01n1g1j/view?usp=sharing](https://drive.google.com/file/d/1lbpr0gV6_sMH7lplqeQImFhcD01n1g1j/view?usp=sharing)

# Game Flow Diagrams

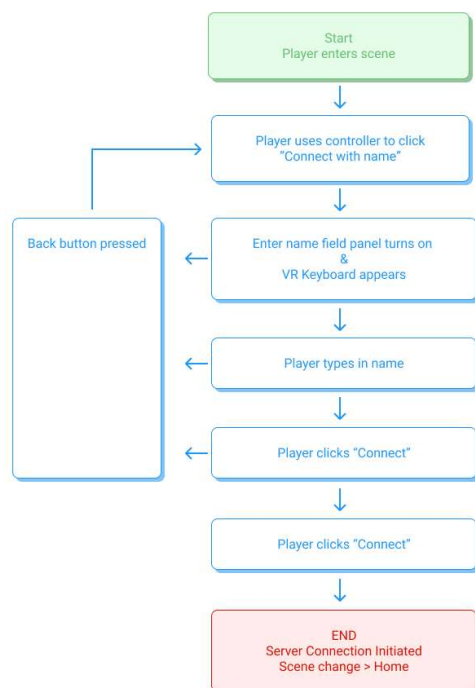
## Login Scene

The login scene is the first scene that the player sees when they enter the game.

The player must enter a name to start the game.

Behind the scenes this is where the server connection is initiated. This name is also the name which is used for player credits, scores and check ins.

### Login Scene – Game Flow Diagram



[https://drive.google.com/file/d/1mojFDWPInVS\\_dalQP5Y89U3oXrebGq\\_O/view?usp=sharing](https://drive.google.com/file/d/1mojFDWPInVS_dalQP5Y89U3oXrebGq_O/view?usp=sharing)



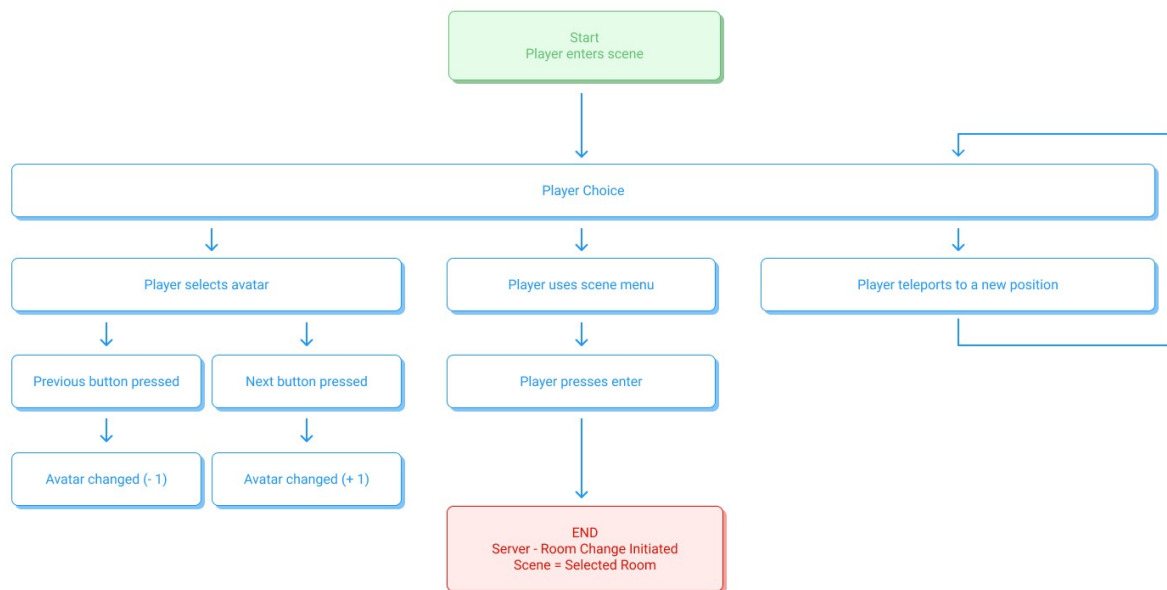
# Home

After a player has logged in they will be directed to the Home Scene.

This scene is a single player only room where the player can select an avatar, get used to VR and select the room which they would like to explore.

The player navigates to the next room by selecting a room on the rooms menu.

## Home Scene – Game Flow Diagram



[https://drive.google.com/file/d/1mojFDWPInVS\\_dalQP5Y89U3oXrebGq\\_O/view?usp=sharing](https://drive.google.com/file/d/1mojFDWPInVS_dalQP5Y89U3oXrebGq_O/view?usp=sharing)

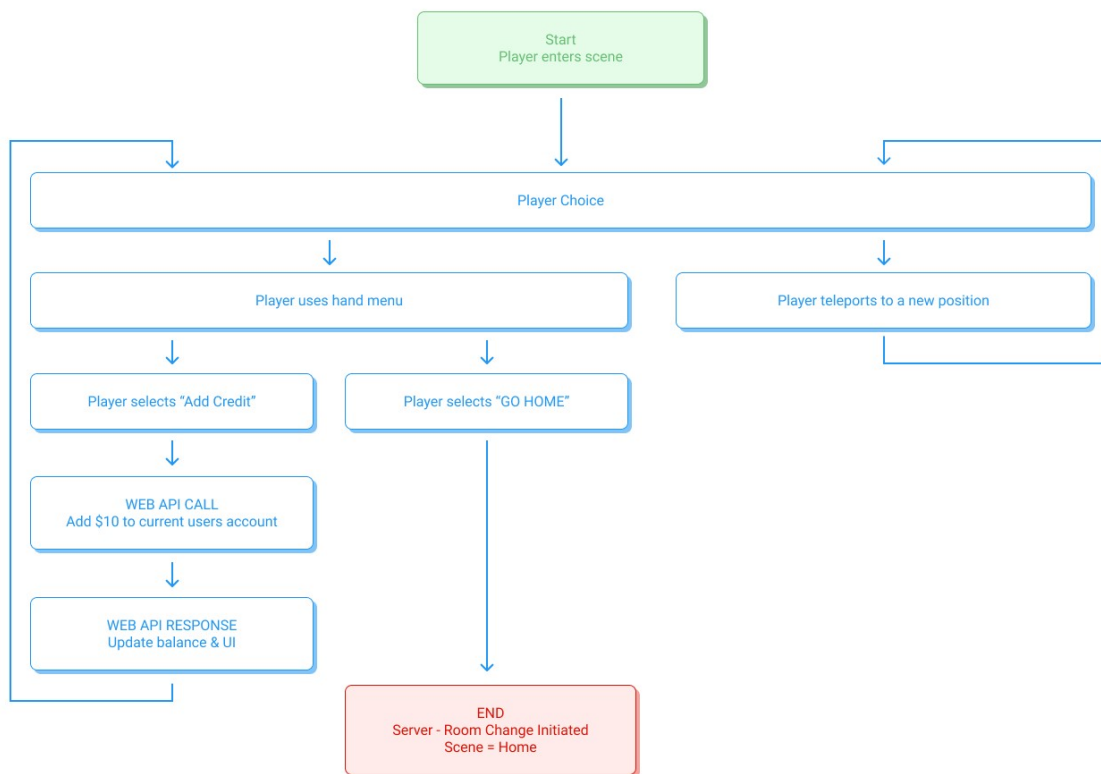


## Lobby

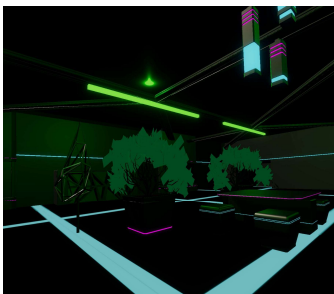
The lobby is the simplest of all the rooms and this area is designed to be a simple hang out space.

When the user is in this room they do not have their credit deducted each minute like the other rooms.

### Lobby - Game Flow Diagram



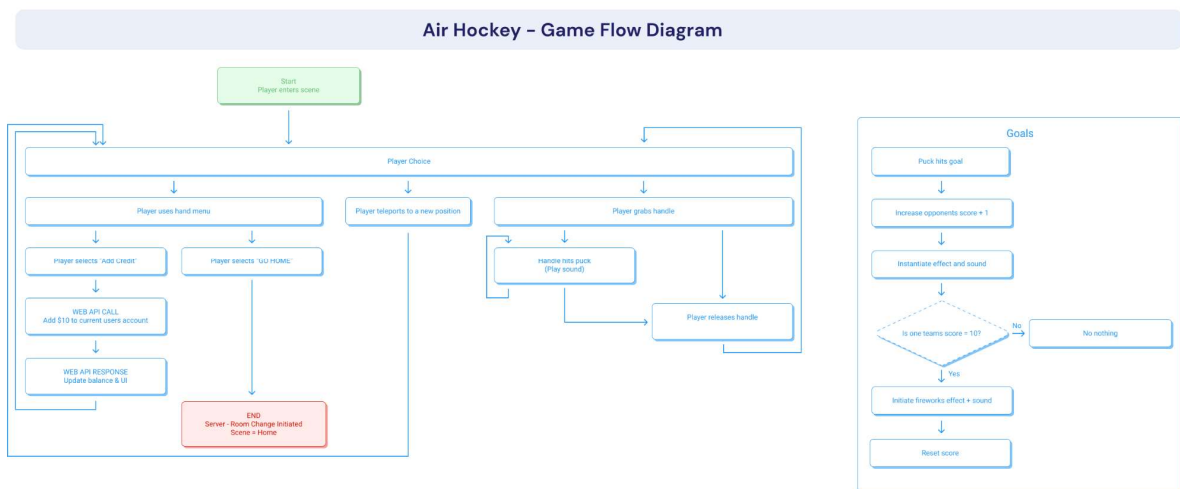
[https://drive.google.com/file/d/1p0rNtPSDGF\\_0bfb8oqvFLmlfq8483Q6V/view?usp=sharing](https://drive.google.com/file/d/1p0rNtPSDGF_0bfb8oqvFLmlfq8483Q6V/view?usp=sharing)



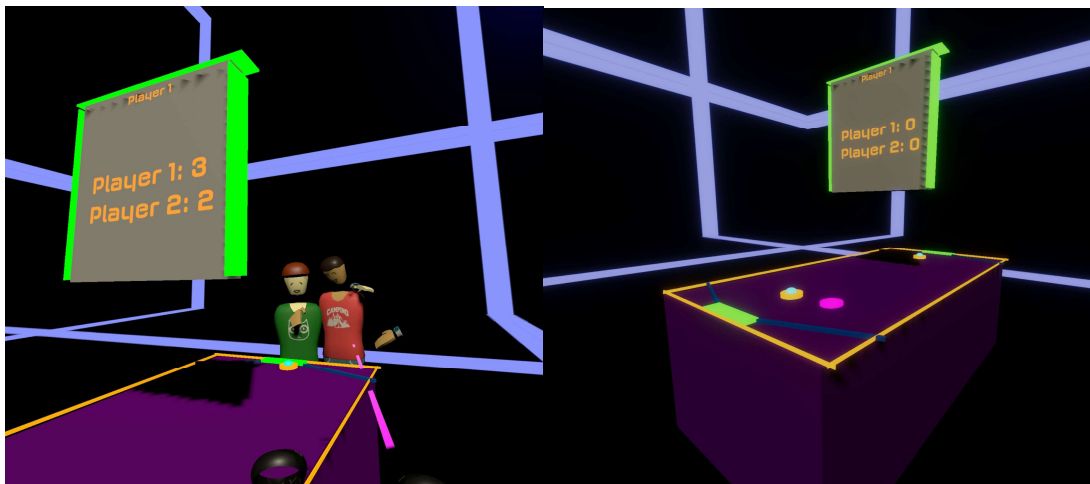
# Air Hockey

Air hockey is a multiplayer room in which there is an air hockey table. The players can each grab a handle and play a multiplayer game of air hockey by hitting the puck into their opponents goal.

The game ends and resets after a fireworks display when one of the players' scores reaches 10. Scores for this game are not recorded.



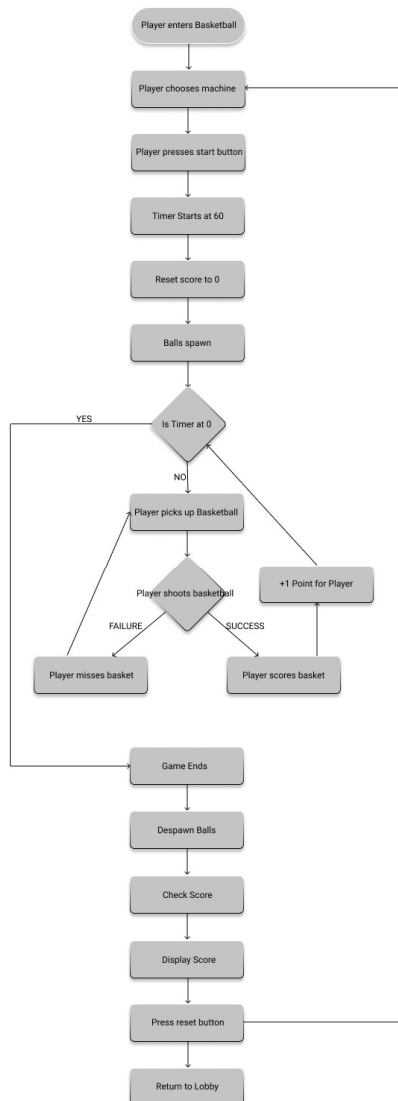
[https://drive.google.com/file/d/1SuOiZE2mrG9m6zLCEOZ\\_6lJuHzeL3pUN/view?usp=sharing](https://drive.google.com/file/d/1SuOiZE2mrG9m6zLCEOZ_6lJuHzeL3pUN/view?usp=sharing)





# Basketball

Basketball is a multiplayer room with several basketball machines. The players can each play at a different machine. Once they start the game, they will have a 60 second timer and 3 balls that spawn. The goal of this game is to score as many points as possible within that time frame. The highest score is recorded for everyone to see. Players can choose to reset the timer and score early if they wish.



<https://drive.google.com/file/d/1voZznQLsaBBvznNONfpYSPL7avszjvXM/view?usp=sharing>

# Archery

The archery game is a multiplayer open air experience where two players can try to hit as many targets as they can within the specified time.

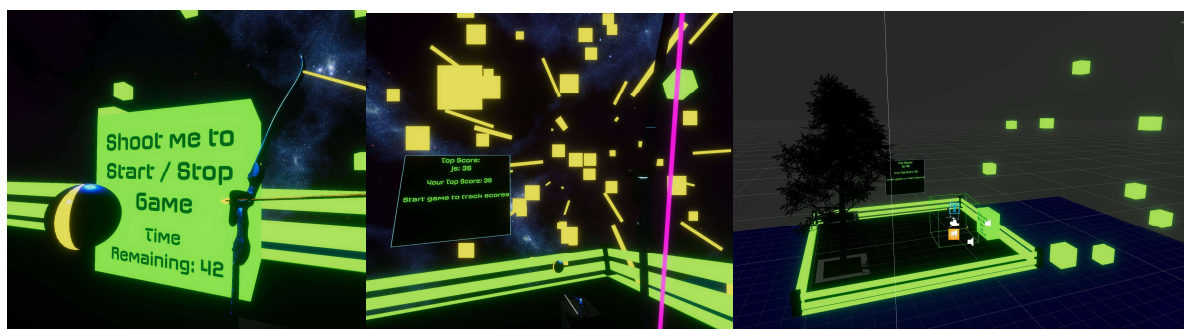
To start the game the player must shoot the Start / Stop game cube.

One this happens the timer's time is set to the length of the hype musc track and the time counter is reduced each second.

When the player fires an arrow at a target and hits that target 1 point is awarded to the player who owned the arrow.

If the players finish a full game (timer = 0) then each player's score is posted to the server and the in game scoreboard is updated.

<https://drive.google.com/file/d/1AYrQx7tNorEfLFXFrzXVQDD-eBeKHRe/view?usp=sharing>

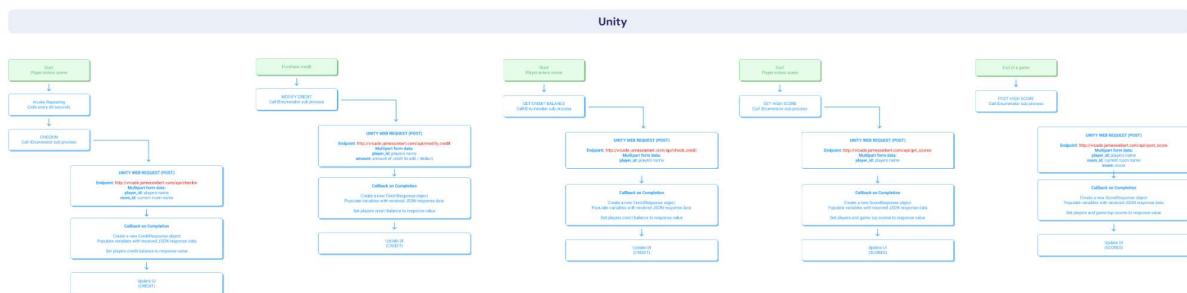


# Server Communication (Score, Credit, Time tracking)

## Unity Web Requests

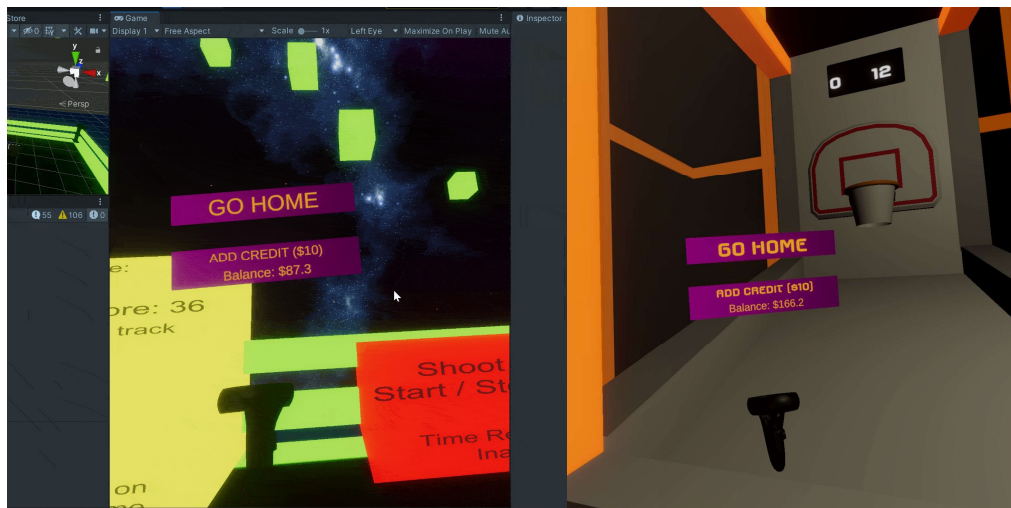
These API calls are made from both the game manager in each scene and the player's character and are called when the player enters the room, when a game is finished, when the user checks their credit balance and adds more credit to their account.

The use of IEnumerators allows for these actions to be performed within a subprocess and the game is not blocked whilst waiting for an api response.



<https://drive.google.com/file/d/1qJmXZQM5mrJUvZO9JU7On1R3zbZrJ2B/view?usp=sharing>

## Add Credit UI



# PHP Server - Rest API

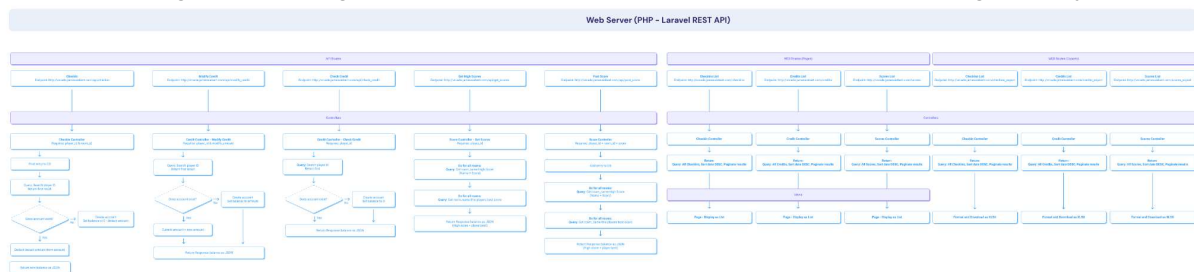
The web server is a PHP based web application built on the laravel framework that handles inbound GET and POST requests.

Once the data has been received and validated by the application various tasks are performed based on where the data was sent (endpoint).

This web application allows humans to read the raw list of database entries along with providing a downloadable XLXS file for further data analysis.

The web application also provides a JSON response back to Unity and the response data is converted back into unity variables which in turn updates player credit balances, scores, UI etc.

Below is a diagram showing all of the endpoints and how the data flows through the system.




[https://drive.google.com/file/d/1EA5JEbeM\\_oT1sEwwFA4IAUNakJbuao3m/view?usp=sharing](https://drive.google.com/file/d/1EA5JEbeM_oT1sEwwFA4IAUNakJbuao3m/view?usp=sharing)


Web API GitHub Repository: <https://github.com/JamesSiebert/vrcade-api.git>

Visit the WEB API Viewer <http://vrcade.jamessiebert.com/>


## VRCADE API

**Info**


This is a very simple API backend for a university project we are developing "VRCADE". This API tracks time usage and fakes an in game purchasing system.

**Check out our game!**

You can find the latest builds and updates and more team info on our [Itch.io Page](#)

**VRCADE OFFICIAL PAGE**

You can find the latest builds and updates and more team info on our [Itch.io Page](#)

**Links**

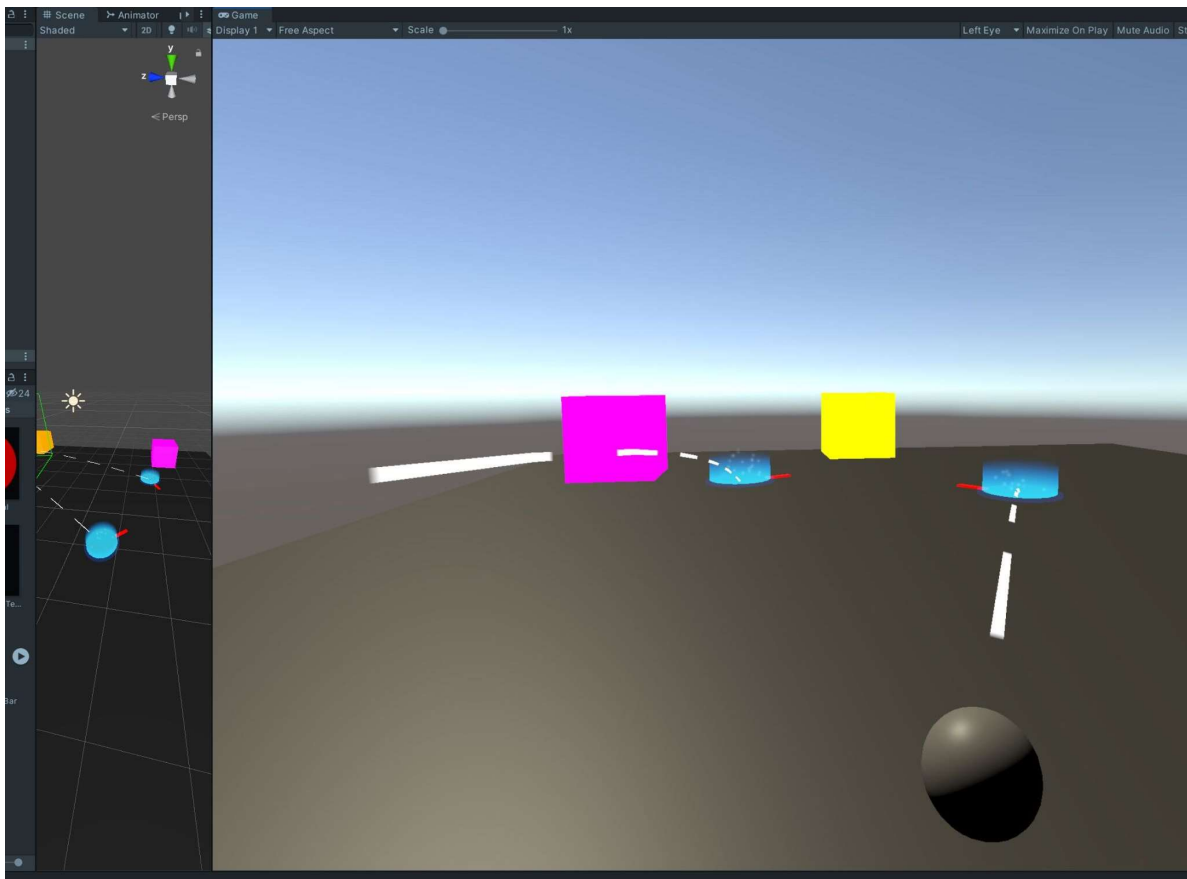
- [View all Checkins List](#)
- [View all Credits](#)
- [View all Scores List](#)
- [Download all checkins as XLSX](#)
- [Download all credits as XLSX](#)
- [Download all scores as XLSX](#)

Laravel v8.33.1 (PHP v7.3.27)

## Custom Teleporter



The custom teleporter is built from the ground up and represents a system I have seen used in Unreal. The teleporter shows the destination and the rotation with the same display. By doing this you reduce one of the steps the player has to do.



# Login System

This is the first scene the player sees when they enter the game.

This area handles the collection of the player's name. This name is used throughout the game as the player's account identifier for scores, credit and time tracking.



## Home scene selection menu

The scene selection menu is ideally a gateway to all of the other scenes.

Then one of the buttons here is clicked Photon Pun2 directs the player to the selected scene with the avatar model they have chosen also assigned.



## Avatar Selection

Avatar selection also happens in the home scene above.

This is just a simple model swap procedure and then an Int is assigned to the player as the players model id.

This id is sent with the player as they move scene to scene.



## Scoreboards

The scoreboards are very simple and only display the values.

Their functionality is governed by the api communications mentioned earlier in this document.

