

DR. ANTONIO BARBA SALVADOR
MARCOS ELADIO SOMOZA CORRAL

PISTOLEROVR

TRABAJO FIN DE GRADO



GRADO EN INGENIERÍA INFORMÁTICA

TABLA DE CONTENIDOS

01

Introducción

02

Inmersión

03

IA

04

Optimización

05

Dashboard

06

Costes

07

Viabilidad

08

Demo

01

Introducción

¿Problema? ¿Solución?
¿Propuesta?



EL PROBLEMA



Pocos videojuegos VR

Según Valve Inc, hay muy pocos nuevos títulos para VR, una ínfima parte de los títulos para PC en su plataforma



Pero... hay jugadores?

Sí, los videojuegos cuentan con más de 3MM de usuarios, con la mayor facturación digital de todos los mercados según el world economic forum



Y de VR ?

Sí, exponencial como el mercado global pero mucho menor, con 173 M de usuarios.

Entonces... porqué no hay más videojuegos VR?

¿POR QUÉ?



Costes de prod.

Al utilizar hardware específico, se incrementan los costes de producción



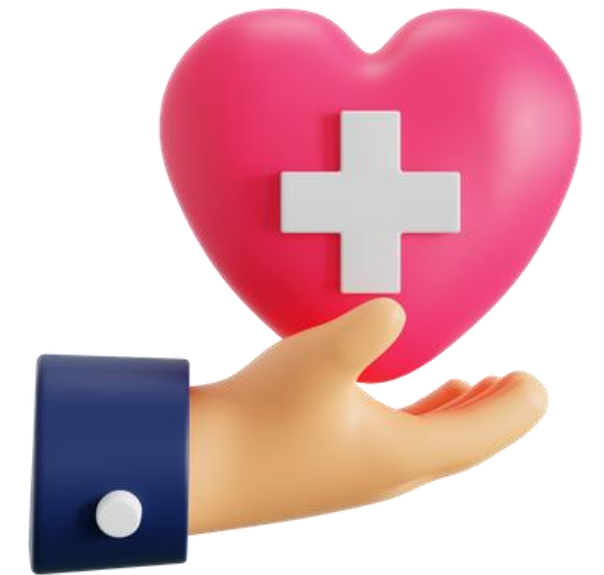
Potencia del HMD

Las gafas VR son mucho menos potentes que los PC's o consolas actuales



FPS

Por la potencia y la naturaleza del VR, se debe optimizar para llegar a unos FPS aceptables



Problemas de salud

Déficit de la función visomotora, estabilidad postural y cinetosis



¿Qué propongo?

Un pmv de un videojuego VR que además de ofrecer solución a la falta de productos, que cumpla con las expectativas de los usuarios ofreciendo una UX satisfactoria.

Además de una documentación de buenas prácticas para el desarrollo en VR para futuros desarrolladores



¿Cómo es?

PistoleroVR es un videojuego FPS por oleadas de acción, donde se deberá sobrevivir a enemigos inteligentes, comprando armas, desbloqueando caminos y recogiendo powerups.

02

Inmersión

¿Cómo crear la
experiencia?



ELEMENTOS

ESCENARIOS 3D DETALLADOS

El escenario es coherente consigo mismo, la iluminación, el esquema de colores y la temática

ELEMENTOS FÍSICOS INTERACTUABLES

La mayoría de elementos cuya version real se percibe como “movibles” son disparables en el juego

ANIMACIÓN PROCEDURAL

El personaje está animado mediante Inverse Kinematics para seguir los mandos y el HMD. Los enemigos también están animados mediante IK procedural, adaptando la posición de sus extremidades al terreno de forma dinámica

SONIDO 3D

Los sonidos con origen (es decir, todo menos la música) se sitúan en el espacio 3D, ayudando al jugador a orientarse

MOVIMIENTO Y ROTACIÓN

Además de que el personaje reaccione a la persona al moverse y rotar en el espacio físico real, este puede mover al jugador, así como rotarlo con los mandos. (además, existen 2 tipos de rotación dependiendo de cada gusto, continua y discontinua)

MÚLTIPLES SISTEMAS DE INTERACCIÓN

TIPOS DE INTERACCIÓN



POKE

Tipo de interacción de “golpecito” con la mano. Uso exclusivo para elementos de la UI

RAY

Tipo de interacción a distancia con un rayo desde la mano. Usado para coger comprar y seleccionar

PHYSICALL

Tipo de interacción de golpeo con la mano. Uso exclusivo para el botón de compra 3D para cosméticos

03

IA

¿Cómo funcionan los
enemigos?



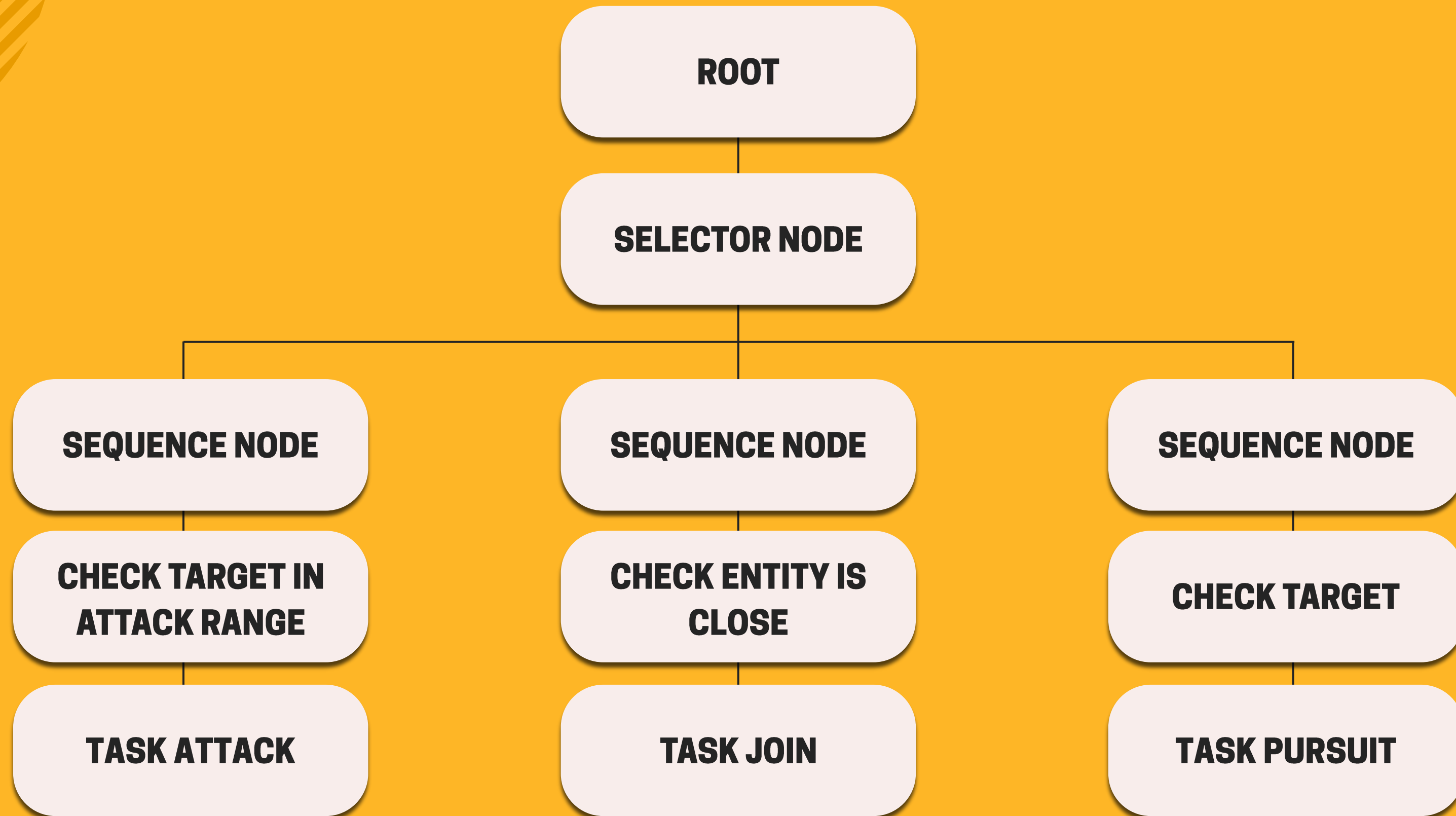
NAVIGATION

Para que los enemigos puedan moverse por el espacio 3D, se ha utilizado el paquete Unity Navigation.

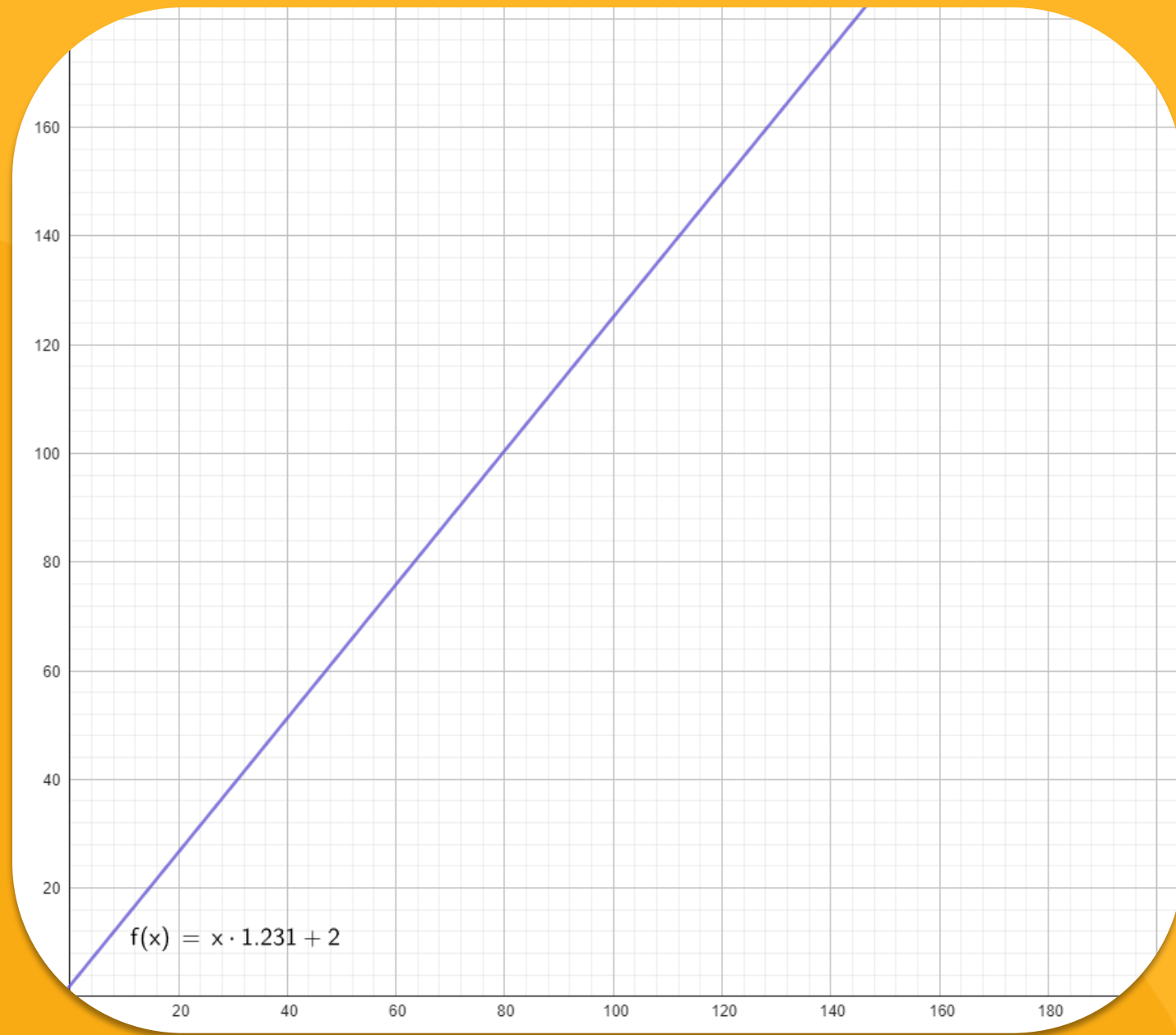
Mediante A* (algoritmo de búsqueda de caminos), encuentra la ruta óptima entre 2 puntos para un agente



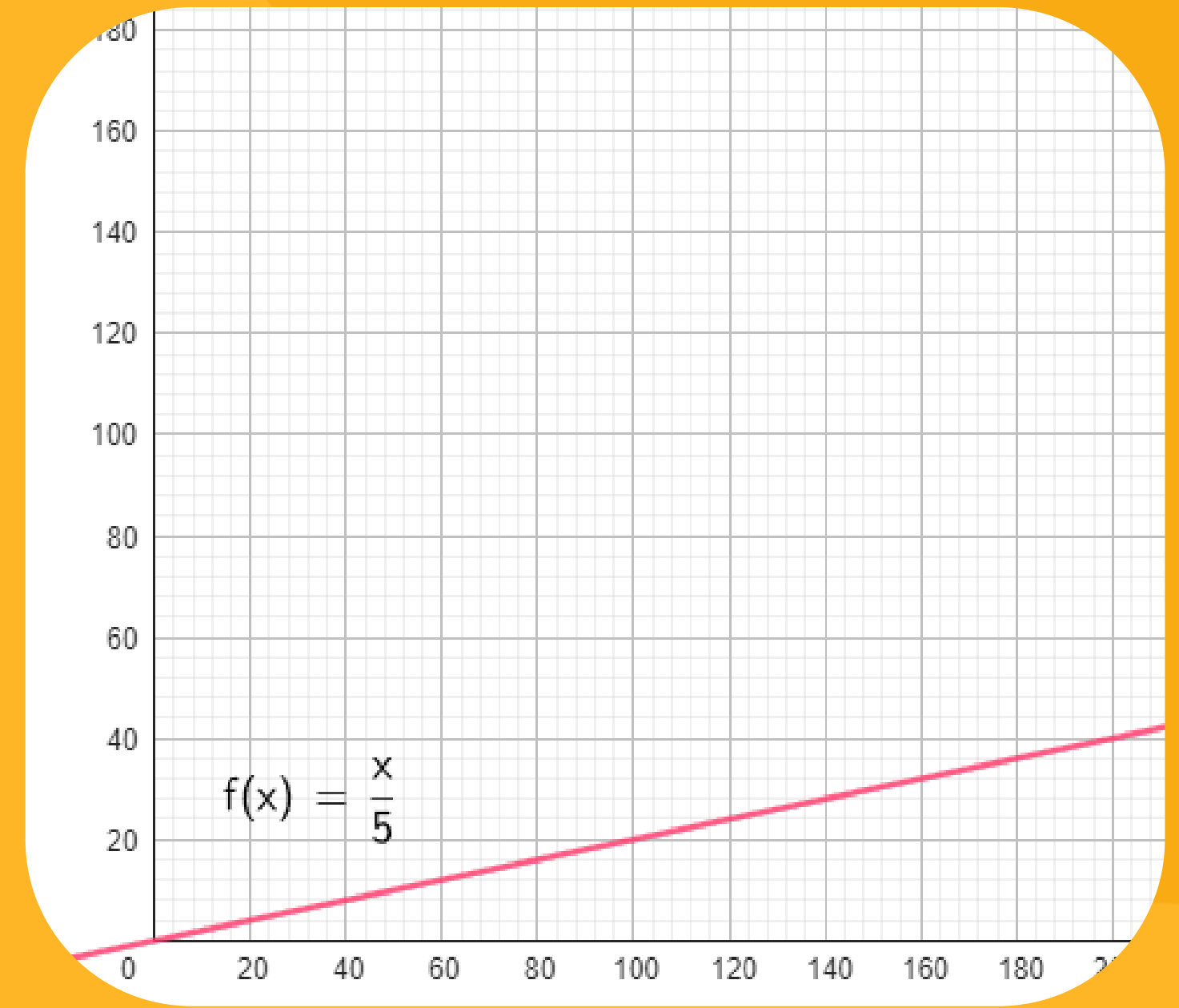
BEHAVIOUR TREE



WAVES MANAGER



Progresión cantidad de enemigos



Progresión hp de enemigos

04

Optimización

¿Cómo se mejoran los
fps?



OBJECT POOLING



Balas

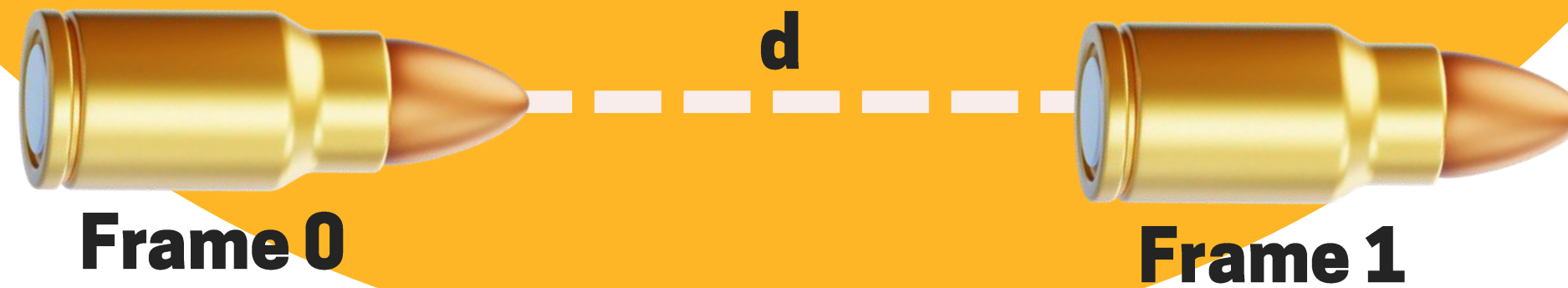
Se ha implementado pools para cada tipo de balas (es decir, como hay 5 armas en total, 5 pools distintos)

Se ha creado un `pool<T>` y pool manager para poder reutilizar el código en las múltiples pools

Enemigos

Así como un pool adicional para los enemigos (los zombies) durante la partida.

RAYCAST SEGMENTS



Para calcular un *hit* de un disparo, se calculan colisiones cada frame entre las posiciones del frame actual y el último. Así, se realizan cálculos más simples y precisos

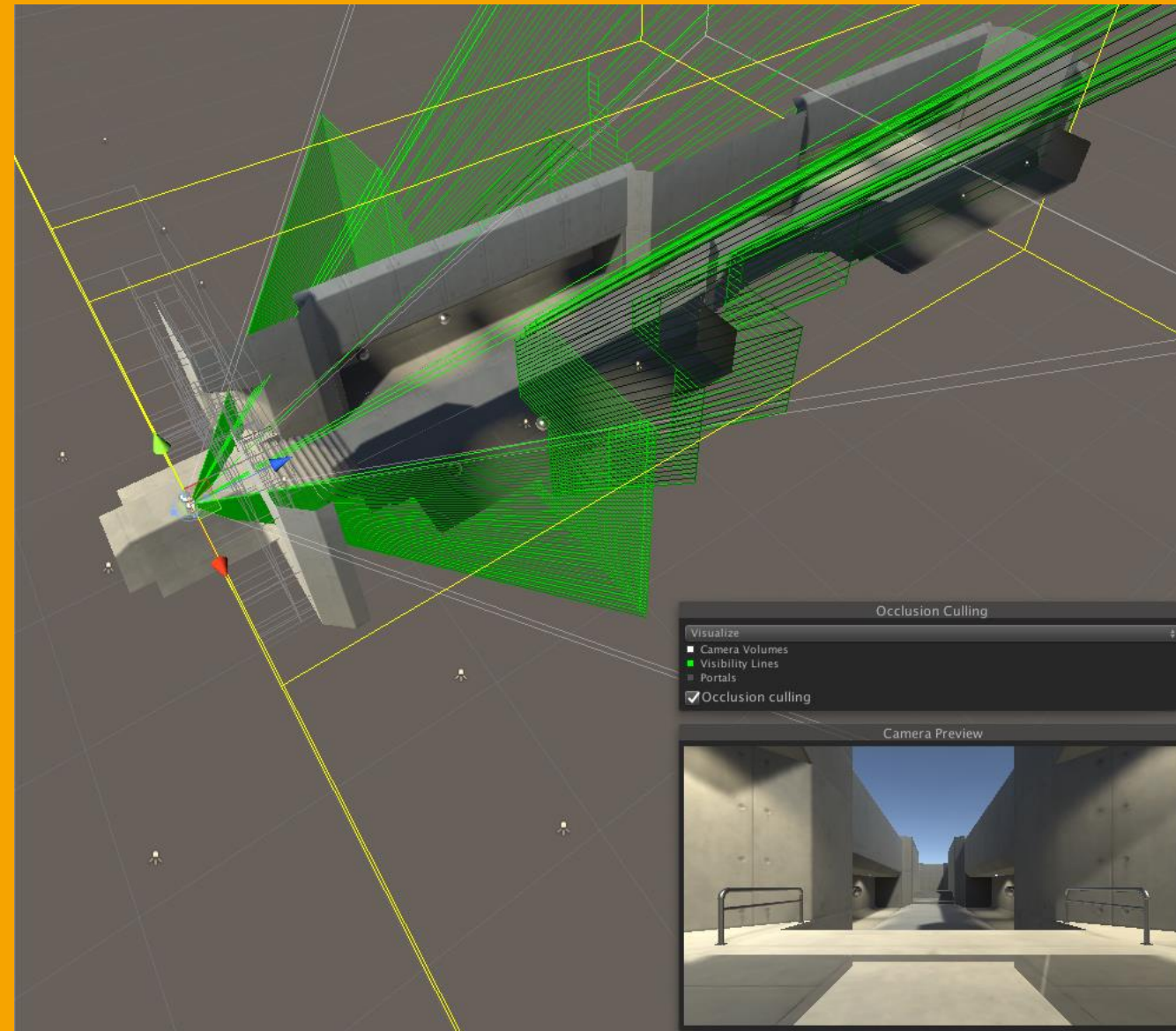
Arquitectura de código

- **Patrón Singleton**
 - GameManager, SoundManager...
- **Patrón Observer**
 - EventsManager
- **Patrón Component**
 - Entities...
- **Patrón ScriptableObject**
 - ShopItem, ShopManager

Otras optimizaciones

Occlusion culling

Técnica de optimización comúnmente utilizada en videojuegos pues desactiva la renderización de aquellos objetos de la escena que no estén en el ángulo de visión de la cámara



Otras optimizaciones

Baking

Técnica donde se precalcula y almacena la iluminación en escenas estáticas, lo que ayuda a mejorar el rendimiento y la calidad visual del juego al eliminar el cálculo en tiempo real de la iluminación.



05

Dashboard

¿Qué se ha utilizado de UGS?



SERVICIOS UTILIZADOS

Authentication

Gestiona el inicio de sesión único de usuarios, otorgando un id único y guardándolo en la base de datos de usuarios.

Cloud Save

Permite almacenar data mediante bbdd no-sql en la nube, asociándola a un id de usuario.




Economy

Gestiona el valor de las monedas utilizadas en el juego para compras en app, asociándolo con un id de usuario.

Cloud Code

Permite la ejecución de código en la nube (js), usado como “backend” para el uso de monedas de economy.

Find player











 production ▼

Players


View all the players from all environments in your game

 Please enter exact player ID

Find player

Player ID		Created	Last login
 1aEHekSOOarbkmNyVuLwID5N35d9		Jun 23, 2023 17:47	Jun 23, 2023 17:47
 21bOrkUiTdRbTqVZMJpPpZgPy6tl		Jun 21, 2023 18:07	Jun 21, 2023 18:07
 5RFTs2Hgu4sbNLMtevdU7AaVbd40		Jun 19, 2023 00:20	Jun 19, 2023 00:20
 F8FmSkhrvWc5j5kzyZbvDeh29iit		Jun 18, 2023 21:31	Jul 7, 2023 23:21
 SKyBELMO5AojV2sJSotwGmQBEVjr		Jun 22, 2023 20:36	Jun 22, 2023 20:36


Authentication

 production ▼













<div>Overview</div> <div>Player details</div>	Last login	Jul 7, 2023, 11:21 PM GMT+2
	Creation date	Jun 18, 2023, 9:31 PM GMT+2
	Linked identities	None

Economy

Cloud Save

<div>Currencies</div> <div>Inventory</div>		
Currency name	Balance	Updated
MONEY 	97	Jul 1, 2023 4:44 PM GMT+2

Economy

Data						
Key	Value	Created	Last Saved	Write Lock	Stored Size	
Hats	[true,false,false,f  <>]	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:19 GMT+2	7caa2e780b7b... 	24 B	...
Kills	18  <>	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:19 GMT+2	3b45edc57df10... 	2 B	...
LastDateTime	"07/07/2023 20:( <>	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:18 GMT+2	6939d90a2f48... 	21 B	...
RewardClaimed	"False"  <>	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:18 GMT+2	751850cbb058... 	7 B	...
Runs	1  <>	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:19 GMT+2	51363a9b60ec... 	1 B	...
SelectedHat	0  <>	Jun 24, 2023 01:13 GMT+2	Jul 7, 2023 21:19 GMT+2	9d8031c62c71... 	1 B	...

Cloud save

Run Code

Details

Script Code

Beautify

```
1  const { CurrenciesApi } = require("@unity-services/economy-2.3");
2  const _ = require("lodash-4.17");
3  const badRequestError = 400;
4  const tooManyRequestsError = 429;
5
6
7
8
9  module.exports = async ({ params, context, logger }) => {
10   try {
11     const { projectId, playerId, accessToken } = context;
12     const economyCurrencyAPI = new CurrenciesApi({ accessToken });
13
14
15     const currencyId = "MONEY";
16     const amount = parseInt(params.amount);
17
18     logger.info(amount);
19
20
21     await subtractCurrency(economyCurrencyAPI, projectId, playerId, currencyId, amount);
22
23     // return the granted currency and amount to calling (Unity) script
24     return { currencyId: currencyId, amount: amount };
25   } catch (error) {
26     transformAndThrowCaughtError(error);
27   }
28 };
29
30
31  async function subtractCurrency(economyCurrencyAPI, projectId, playerId, currencyId, amount) {
32    const currencyModifyBalanceRequest = { currencyId, amount };
33    const requestParameters = { projectId, playerId, currencyId, currencyModifyBalanceRequest };
34    await economyCurrencyAPI.incrementPlayerCurrencyBalance(requestParameters);
35  }
36
37  function transformAndThrowCaughtError(error) {
```

Parameters

Run

Parameter name	Value
amount*	<div>1</div>

Player ID



9MfBFPxalvk1XRUCl6D5hMQQBnes



	RESPONSE	LOGS	REQUEST
1	Received script response in: ~945ms		
2	-----		
3	Response body:		
4	-----		
5	{		
6	"amount": 1,		
7	"currencyId": "MONEY"		
8	}		

Cloud code

06

Costes

¿Cuánto cuesta el pmv?



PRESUPUESTO

BRUTO

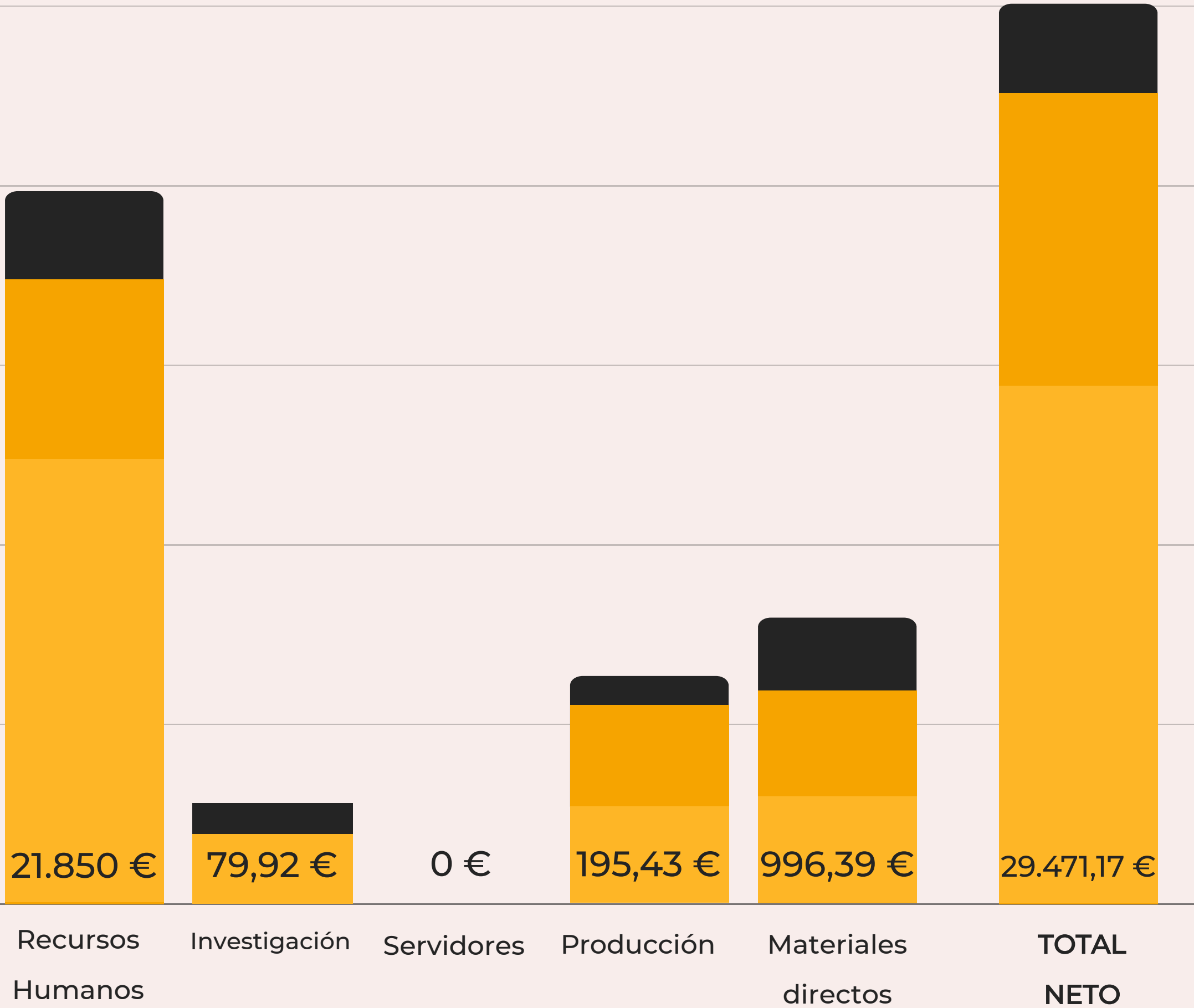
23.222,25 €

IVA

4.876,67 €

NETO

29.471,17 €



07

Viabilidad

¿Qué se ha utilizado de UGS?



Previsión ROI

PISTOLERO VR

Obj. Ventas - 20.000
Precio - 19,99 €

Coste - 40.000 € ✱
ROI - 599,65%

BEAT SABER

Ventas - 4.000.000
Precio - 29,99 €

Coste - 100.000 €
ROI - 83.872,00%

GORILLA TAG

Ventas - 760.000
Precio - 19.50 €

Coste - 40.000 €
ROI - 17.190,00%



RECORDANDO QUE...

● ● -30% de todas las ventas por las plataformas de distribución

● ● 25 € es el precio medio de videojuegos indie VR del mercado

● ● El coste estimado para pasar de pmv a producto final es de ~40.000 €

● ● Abre las puertas a inversores y financiación para futuros proyectos

08

Demo

¿Cómo es jugarlo?

