

DEADZONE LAST STAND



Fernando Luis Amaya Jiménez
2ºDAM

Índice:

Índice:	1
1. Descripción del Proyecto.	2
2. Forma, mecánicas, contexto, arte, target del Juego.	2
3. Justificación del Proyecto	4
4. Alcance del Proyecto	5
5. Stack Tecnológico	5
6. Valoración de Alternativas Existentes	5
7. Objetivos.	7
7.1. Objetivos principales del jugador:	7
7.2. Objetivos secundarios del jugador:	7
8. Requisitos.	7
8.1. Requisitos principales:	7
8.2. Requisitos no principales:	7
8.2. Requisitos de interfaz:	7
9. Casos de uso.	8
1º Caso de uso.	8
2º Caso de uso.	8
3º Caso de uso.	9
10. Pantallas de prototipado en figma.	9
11. Guía de estilo en figma.	12
12. Navegación entre pantalla con figma.	14
13. Modelo y diseño de la base de datos (playfab).	14
14. Manual de Usuario.	17
Paso 1: instalado.	17
Paso 2: Cambiar la ventana de registrarse.	17
Paso 3: Registrarse.	17
Paso 4: Iniciar sesión.	18
Paso 5: Menú Principal.	19
Paso 6: Jugar y armería.	19
Paso 7: Juego.	20
Paso 8: Ejemplo y fin de partida.	22
Paso 9: Ver ranking.	23
Paso 10: Mejorar el Arma.	24
15. Manual de Configuración.	25
15.1. Configuración de despliegue.	25
15.1. Configuración de Administrador.	27
16. Manual de Instalación.	29
16.1. Requisitos del sistema operativo y de hardware:	29
16.2. Instalación:	29
17. Retos.	33
18. Posibles mejoras.	34
19. Conclusión.	34
20. Repositorio con el código fuente.	34

1. Descripción del Proyecto.

Mi proyecto consiste en el desarrollo de un videojuego usando el motor de juego multiplataforma Unity, ambientado en un mundo postapocalíptico donde un brote de hongos ha evolucionado hasta poder controlar a los humanos infectados. Una vez infectado, un humano muere y el hongo toma control de su cuerpo con el objetivo de propagar la infección a otros humanos, alimentando al hongo.

El propósito del explorador es lanzarse a las ruinas cercanas a la ciudad para proteger a los suyos, ya que no paran de salir infectados en las ruinas. Cada partida es, dentro del juego, un día de una semana: el viernes, el cual eligen a un explorador para que salga allá afuera y elimine zombies hasta la muerte. Luego, un grupo grande recogerá todo lo que este explorador haya conseguido.

El juego no tendrá un protagonista fijo. En cada partida, el jugador controlará a un nuevo personaje, el cual es un explorador elegido de una ciudad de humanos no infectados. La mecánica principal del juego consiste en que el jugador debe explorar fuera de la ciudad con el objetivo principal que es eliminar infectados, los cuales, al ser eliminados, pueden proporcionar recursos al jugador. Con esos recursos/ monedas más adelante se podrán obtener cosas a cambio (como armas o mejoras del arma).

Los recursos obtenidos se sumarán a un contador global de la ciudad, compartido entre todos los personajes de un mismo jugador. Con esos recursos recolectados se podrán comprar distintas armas y mejoras, una arma por partida jugada. Una vez se acabe la partida, el arma debe volverse a comprar, ya que la partida acaba cuando el explorador es infectado, pero las mejoras de las armas son permanentes. Cada partida termina con la eliminación del explorador, debido a la constante oleada de infectados que aparecen en el exterior.

El juego contará con un sistema de ranking donde se podrá ver el top 5 mundial de las mejores partidas de cada jugador. Se valorará la partida por la cantidad de kills (eliminaciones) de infectados que haga el jugador. Además, para el Ranking solo se guardará una partida por cada jugador (su mejor partida).

2. Forma, mecánicas, contexto, arte, target del Juego.

Forma	“DeadZone Last Stand” es un videojuego, este videojuego busca la inmersión del jugador en un mundo postapocalíptico donde hay zombies, por lo tanto el jugador estará constantemente en un ambiente de supervivencia.
-------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Mecánica	<p>Controles</p> <ul style="list-style-type: none"> • Movimiento: Teclas WASD. • Apuntar: Ratón. • Disparar: Clic izquierdo del ratón. • Sistema de munición en cada arma, La recarga se hace con la R. • El personaje puede correr al Shift. <p>Enemigos y Obstáculos</p> <ul style="list-style-type: none"> • Oleadas constantes de infectados con variedad de zombies, los cuales atacan al jugador hasta eliminarlo. • Sistema de recompensas aleatorias entre vida, recursos y munición al eliminar un infectado. • La partida acaba solo cuando el explorador es eliminado/infectado. • Los infectados tienen una IA desarrollada con un NavMesh (componente importado) la cual consiste en calcular la superficie para que el infectado siga la ruta más corta entre los obstáculos hasta el jugador. • El mapa tiene varios elementos tanto sólidos (obstáculos) y otros como carreteras y algunas plantas pequeñas las cuales podemos pasar por encima. <p>Jugador</p> <ul style="list-style-type: none"> • Sistema de compra y mejora de armas. • Sistema de vida, contador de kills de infectados, contador de recursos. • Ranking visible del top 5 mejores partidas de los jugadores (a cada jugador se le guarda su mejor partida y esta saldrá en el Ranking). Si el jugador no aparece en el ranking igualmente podrá ver su posición mundial ya que es visible. • Sistema de iniciar sesión y login para guardar los datos de ese jugador y así no empiece de cero cada vez que juega. <p>Objetivo</p> <ul style="list-style-type: none"> • El objetivo del jugador es mejorar las armas al máximo para descubrir cuál es la mejor para acabar en una partida con el mayor número de infectados y así poder posicionarse en el ranking mundial.
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Contexto	<p>“DeadZone Last Stand” trata de un mundo postapocalíptico donde un brote de hongos ha evolucionado hasta el punto de poder controlar a los humanos infectados. Cuando una persona se infecta, muere, y el hongo toma el control de su cuerpo con el objetivo de seguir propagándose a través de otros humanos. Estos cuerpos controlados por el hongo atacan a los que siguen vivos para alimentarse y multiplicarse, volviendo cada vez más peligrosa la situación fuera de las murallas.</p> <p>La ciudad donde habitan los humanos no infectados es el último lugar seguro, rodeado por ruinas infestadas de infectados. Cada viernes se elige a un explorador para que salga a estas ruinas y elimine a tantos enemigos como pueda antes de ser alcanzado y devorado. Su objetivo no es sobrevivir, sino proteger a los suyos ganando tiempo y recolectando recursos que el grupo de rescate recogerá después de su muerte.</p>
Arte	<p>“DeadZone Last Stand” se propone inicialmente de arte en 2 Dimensiones con una vista 100% desde arriba, gama de colores amplia (negro, gris, verde oscuro, amarillo, verde, marrón oscuro y blanco). Estos son los más destacados, los colores han sido elegidos para que el jugador se inmersa en un mundo postapocalíptico lleno de zombies.</p> <p>La elección de asset ha sido de coches tanques, casas en ruinas pero con muchos árboles para darle ambientación de poca vida humana.</p> <p>Además tiene varios sonidos como música en el menú, rugido de zombie, disparar, si no te alcanza el dinero al comprar algo hace un sonido que indica error, al recargar, etc. Estos sonidos introducen al jugador en el videojuego.</p>
Target	<p>Para mayores de 14 años, es para ordenador con sistema operativo windows.</p>

3. Justificación del Proyecto

Este videojuego busca ofrecer una experiencia desafiante, donde cada partida tenga un impacto en el progreso del jugador. La mecánica de mejorar mediante la supervivencia y recolección de recursos fomenta a los jugadores a explorar, mejorar estrategias y optimizar el uso de sus recursos y experiencia.

4. Alcance del Proyecto

El proyecto incluirá:

- Desarrollo del motor del juego en Unity.
- Control total del explorador, tanto el movimiento como el apuntado de su arma, además podrá, disparar el arma elegida de esa partida.
- Implementación de sistema aleatorio cuya función sea al eliminar un zombie/infectado este proporciona aleatoriamente vida, recursos o munición para el arma.
- Implementación de IA enemiga, que determine el comportamiento de los infectados (persiguen al jugador todo el tiempo calculando la ruta más corta).
- Ranking global, para fomentar la competencia entre jugadores.
- Tienda de armas, donde podrás comprar un arma de forma temporal para una partida o poder mejorar dicha arma de forma permanente.
- Diseño de mapa grande y atractivo para el jugador con muchos obstáculos para mayor estrategias y así despistar a los infectados.

5. Stack Tecnológico

Para el desarrollo del juego, se utilizará el siguiente stack tecnológico:

- Motor de juego: Unity C# para el desarrollo y la física del juego.
- Para el sistema de iniciar sesión y logueo del videojuego utilizó LoginWithPlayFab.
- Base de Datos: Playfab Player Data almacena datos importantes del jugador como los niveles de su arma, materiales, etc.
- Para hacer el Ranking utilizó otro apartado de Playfab que es DisplayName que guarda el nombre que el jugador tiene para otros jugadores y Player Statistics para guardar el record de kills, porque el PlayerData es privado y para hacer el ranking necesitaba que pude ser público como Player Statistics.
- Playfab también tiene un panel de administrador para administrar a los jugadores.
- Interfaz gráfica: Sprites en 2D.
- Para la tipografía de textos he utilizado TextMeshPro
- Sistemas de audio, para que el videojuego emita sonidos e introduzca al jugador en el videojuego.
- Implementación de NavMesh, esto sirve para crear la IA de los zombies y calcular el recorrido más corto hasta el jugador.

6. Valoración de Alternativas Existentes

Existen otros juegos de supervivencia con temática de zombis, pero la mayoría se centran en la acción con un personaje único como puede ser the last of us, este juego es un modo historia de unos únicos personajes que al morir vuelven hacia atrás. Nuestro juego se diferencia por su enfoque en la supervivencia a corto plazo, donde cada personaje está destinado a morir, pero su sacrificio ayudará al crecimiento de la ciudad y a mejorar las habilidades para futuras partidas, a parte también tendrá una historia aunque mucho más

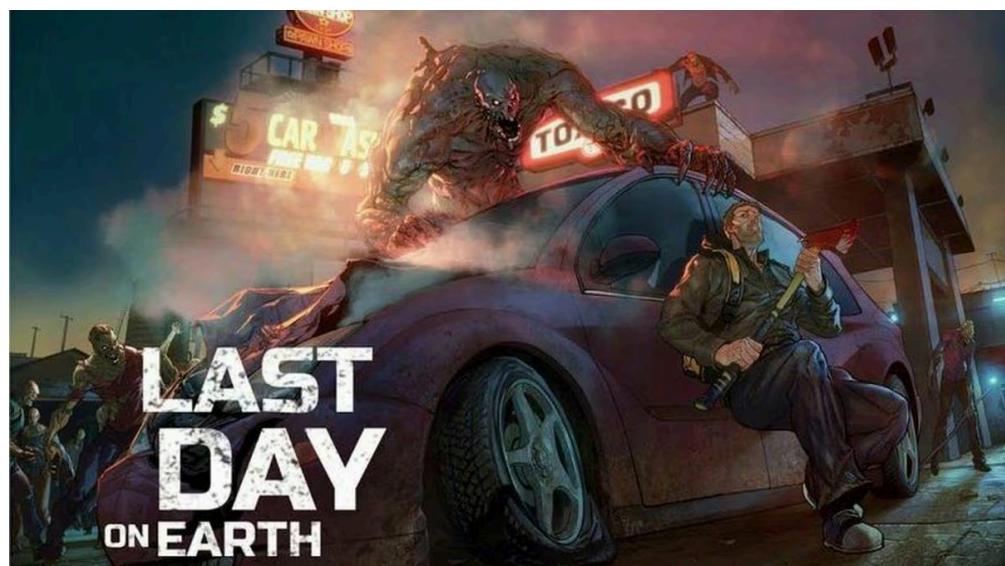
corta que la del juego the last of us, pero lo que sí tendrá el videojuego es la competitividad entre jugadores ya que el propio juego va a mostrar a los jugadores con mejores partidas.

Otras competencias de DeadZone Last Stand serían:

- Last Stand: Aftermath es un juego donde controlas a un infectado que sabe que va a morir, así que sale al exterior a matar zombies y conseguir recursos antes de que le llegue la hora. Cada partida es distinta y el objetivo es ayudar a los que quedan vivos.



- Last Day on Earth: Survival es un juego de supervivencia donde tienes que recolectar recursos, construir tu base y enfrentarte a zombies y otros jugadores. El mundo está lleno de peligro y la clave es sobrevivir el mayor tiempo posible.



Al final, DeadZone Last Stand, al estar ambientado en un mundo lleno de zombis, entra dentro de un catálogo bastante amplio de videojuegos con temáticas similares, ya que su

objetivo principal también es la eliminación de infectados. Hay muchos juegos parecidos por el enfoque en la acción y la supervivencia contra hordas de zombis.

7. Objetivos.

7.1. Objetivos principales del jugador:

- Pasarlo bien jugando.
- Eliminar la mayor cantidad de zombies en una partida, para así posicionarse en un mejor puesto en el ranking.
- Conseguir la mayor cantidad de materiales en las partidas que juegue para poder así mejorar sus armas.

7.2. Objetivos secundarios del jugador:

- Mejorar todas las armas del juego y así llegar al límite del juego.
- Sobrevivir el mayor tiempo posible en cada partida.
- Conseguir munición y vida en cada partida para aguantar más.

8. Requisitos.

8.1. Requisitos principales:

- El jugador puede registrarse e iniciar sesión y que aparezca en PlayFab.
- Navegación entre pantallas y escenas.
- El jugador puede elegir el arma y utilizarla contra los zombies.
- El arma debe de disparar y recargar.
- Los zombies deben perseguir al jugador hasta eliminarlo o sean eliminados.
- Los zombies al ser eliminados se destruyen.
- El jugador al eliminar un zombie le tiene que llegar una recompensa ya sea: materiales, vida o munición.
- Respawn infinito de zombies.

8.2. Requisitos no principales:

- Que exista un ranking para saber cuales son los mejores jugadores.
- Sistema de mejorado de armas.
- Sistema de compra de armas.
- Muestra cual es la posición actual del jugador en el ranking mundial.

8.2. Requisitos de interfaz:

- Tener un escenario grande y divertido

- La vista del jugador será 100% desde arriba.
- Al cambiar de arma cambia el asset del personaje.
- Sonidos como disparos, gruñidos de zombies, etc.
- Tener unos paneles de menús bonitos, intuitivos y además sin sobrecargarlos.

9. Casos de uso.

1º Caso de uso.

Elemento	Descripción
Actor	Jugador
Objetivo	Eliminar a un zombie
Precondición	Tener balas en el arma
Flujo principal	<ul style="list-style-type: none"> - El jugador se acerca al zombie. - Le apunta con el ratón. - Hacer click izquierdo y disparar. - Le da y le quita vida. - Repite hasta que el zombie se queda sin vida. - El zombie se elimina y da puntos al jugador.
Flujo alternativo	<ul style="list-style-type: none"> - Falla y no le da → El zombie no pierde vida. - No tiene balas → El arma no dispara → El zombie no muere. - Se acerca mucho → El zombie elimina al jugador.

2º Caso de uso.

Elemento	Descripción
Actor	Jugador
Objetivo	Recoger materiales
Precondición	Que haya zombies cerca
Flujo principal	<ul style="list-style-type: none"> - El jugador dispara al zombie. - El jugador elimina al zombie. - Le puede tocar: curarse, reponer munición o recoger materiales.

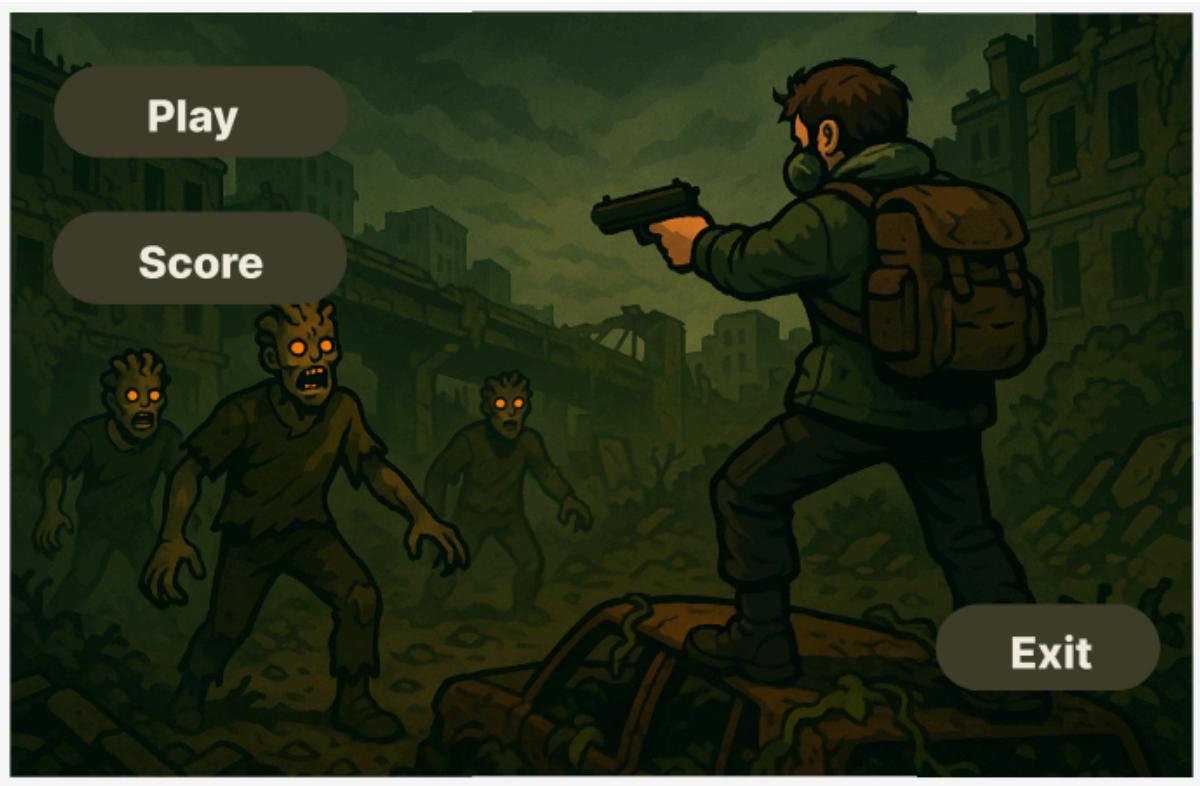
Flujo alternativo	<ul style="list-style-type: none"> - Hay demasiados zombies cerca → Acorralan al jugador → Lo eliminan. - Elimina al zombie pero le toca vida o munición en lugar de materiales.
-------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

3º Caso de uso.

Elemento	Descripción
Actor	Zombie
Objetivo	Eliminar al jugador
Precondición	El jugador esté bajo de vida y falle los disparos
Flujo principal	<ul style="list-style-type: none"> - El zombie se acerca al jugador. - Ataca al jugador. - Elimina al jugador.
Flujo alternativo	<ul style="list-style-type: none"> - El jugador tiene mucha vida → El zombie es eliminado. - El jugador elimina antes al zombie.

10. Pantallas de prototipado en figma.

[https://www.figma.com/design/r7jWQ0E4Nss5VX9feKjCOJ/Untitled?node-id=0-1&p=f&t=cw
YesO6frXJhx3In-0](https://www.figma.com/design/r7jWQ0E4Nss5VX9feKjCOJ/Untitled?node-id=0-1&p=f&t=cwYesO6frXJhx3In-0)



PUNTUACIONES



famajim030@g.educaand.es 30:42 min 38 kills



psalsol161@g.educaand.es 10:12 min 23 kills



juanronda@g.educaand.es 5:29 min 13 kills

EXIT

ARMERÍA

9999



II

gratis



Mejorar

50



1000



Mejorar

600



III

1500



Mejorar

400



I

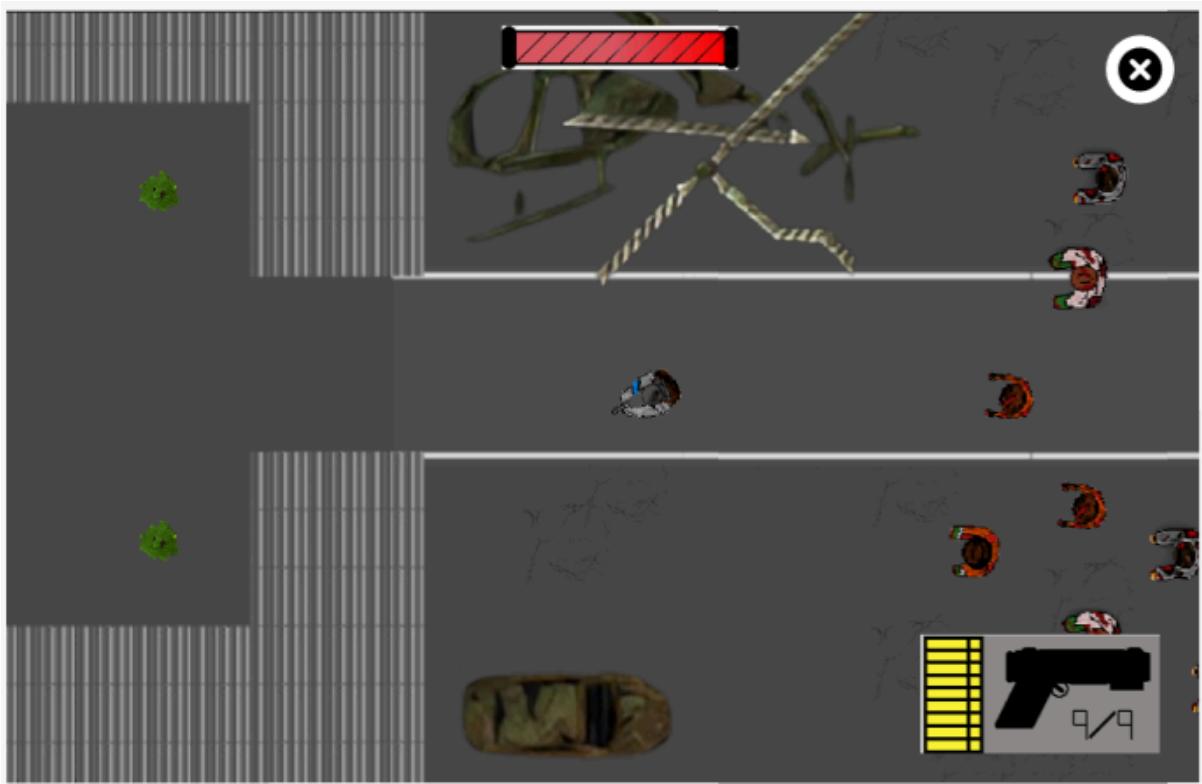
300



Mejorar

150

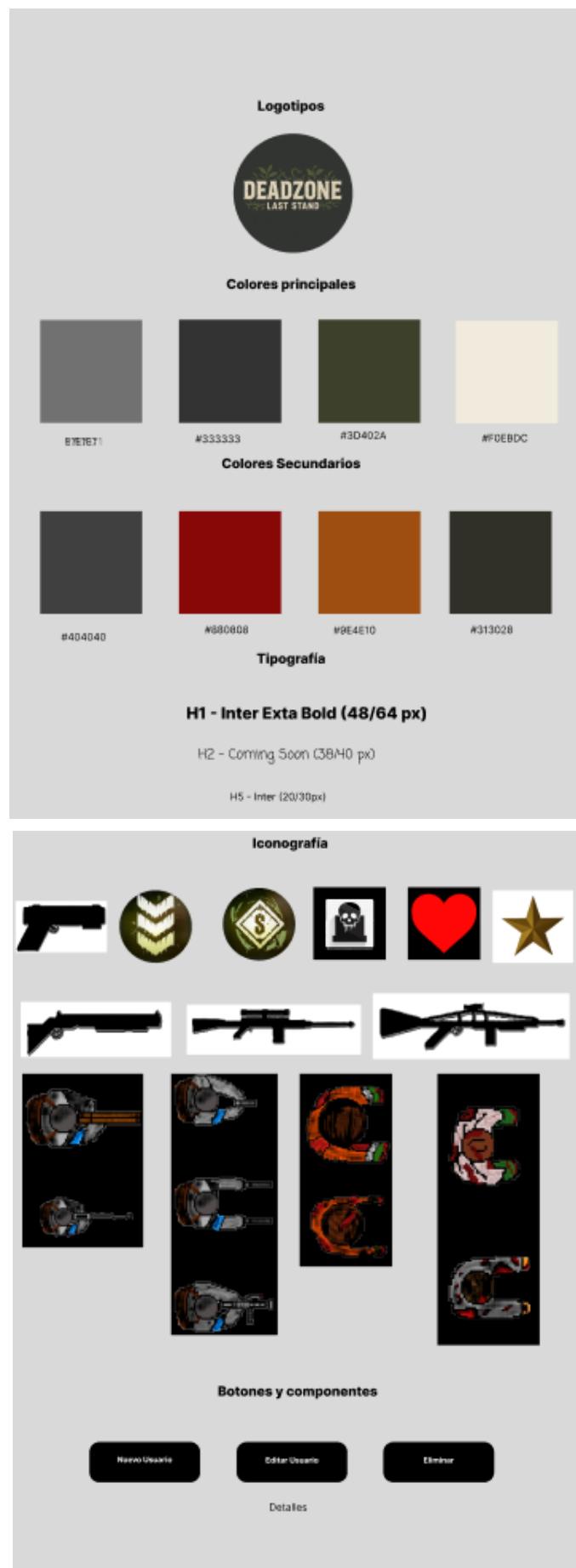




Este diseño al ser el prototipo era la idea principal de cómo se iba a ver el videojuego, el resultado final no son exactamente iguales, ya que a lo largo de la creación del proyecto he añadido distintos cambios para que el diseño sea mejor.

11. Guía de estilo en figma.

https://www.figma.com/design/O2TzDrxvqE2oO2eectFxIS/Untitled?node-id=0-1&p=f&t=LbjIR_OgHp6kDvZMz-0



12. Navegación entre pantalla con figma.

<https://www.figma.com/proto/r7jWQ0E4Nss5VX9feKjCOJ/Untitled?node-id=23-32&starting-point-node-id=23%3A32>

13. Modelo y diseño de la base de datos (playfab).

Para la base de datos he utilizado PlayFab ya que tiene una función de base de datos player data esta base de datos funciona como una base de datos NoSQL de tipo clave - valor asociada a un jugador.

Player data

Key *	Value *	Permissions
PlayerMoney	370	Private
Pistola	443.10.34.60	Private
Escopeta	4.73.14.6.30	Private
Fusil	4.47.31.70.120	Private
Francotirador	4.130.30.5.20	Private
kills	9	Private

+ Add

Read only data

En la imagen anterior se puede ver cómo se almacenan los datos en PlayFab, específicamente en la sección llamada Player Data. Esta funcionalidad permite guardar información personalizada para cada jugador (aunque en mi caso, la estructura es para todos la misma) mediante un sistema de pares clave-valor. Es decir, cada entrada consta de una Key (clave) y un Value (valor), y estos datos están asociados directamente a la cuenta del jugador.

La Key funciona como un identificador único que luego se utiliza desde el código en C# para acceder a ese dato en particular. Por ejemplo, si queremos obtener el dinero del jugador, solo tenemos que hacer una consulta a PlayFab utilizando la clave "PlayerMoney". Por otro lado, el Value es el contenido que queremos guardar, y aunque PlayFab lo almacena como un simple string, yo decidí estructurar varios datos dentro de un mismo value utilizando un formato tipo CSV.

Ejemplo de cómo lo hago para obtener los datos de las armas desde el videojuego:

```

    | referencia
private void CargarArmaDesdeData(string nombre, string data)
{
    string[] valores = data.Split(',');
    if (valores.Length == 5)
    {
        var armaCargada = new arma(
            nombre,
            int.Parse(valores[0]),
            float.Parse(valores[1]),
            float.Parse(valores[2]),
            int.Parse(valores[3]),
            int.Parse(valores[4])
        );
        armasJugador[nombre] = armaCargada;
        ActivarEstrellas(nombre, armaCargada.nivel);
    }
}

```

En este método hay que añadirle el nombre y la data, el nombre lo utiliza para añadirlo al objeto porque requiere el nombre y la data es lo que ha devuelto al buscar la Key esta data la conseguido aquí:

```

    | referencia
private void ProcesarDatosArmas(GetUserDataResult resultado)
{
    armasJugador.Clear();
    bool huboCambios = false;

    string[] nombresArmas = { "Pistola", "Escopeta", "Fusil", "Francotirador" };

    foreach (string nombre in nombresArmas)
    {
        if (resultado.Data != null &amp; resultado.Data.TryGetValue(nombre, out var itemData))
        {
            CargarArmaDesdeData(nombre, itemData.Value);
        }
        else
        {
            InicializarArmaPorDefecto(nombre);
            huboCambios = true;
        }
    }

    if (huboCambios) GuardarArmasEnPlayFab();
}

```

Con este método busca todas las armas mediante el nombre en PlayFab, si existen los datos y la key en PlayFab se llama al método anterior con el *resultado itemData.Value* que este se le ha añadido el value mediante este método *resultado.Data.TryGetValue(nombre, out var itemData)* este método busca mediante resultado que es una forma de optimizar la búsqueda, luego busca mediante la key nombre el value y se lo añade al *itemData.Value*.

Básicamente, lo que hice fue concatenar diferentes valores en una sola cadena de texto, separados por comas (,). De esta forma, puedo guardar múltiples propiedades (como daño, nivel, cadencia, etc.) dentro de un único valor, lo que resulta bastante útil para mantener los datos organizados y reducir la cantidad de claves necesarias. Luego, cuando necesito acceder a estos datos en el juego, utilizó la función `Split(',')` en C# para separar cada elemento del string y obtener un arreglo con todos los valores por separado.

Ejemplo de guardado de armas en playfab Player Data:

```
2 referencias
public void GuardarArmasEnPlayFab()
{
    var datos = new Dictionary<string, string>();

    foreach (var par in armasJugador)
    {
        arma a = par.Value;
        string valor = $"{a.nivel},{a.danio},{a.distancia},{a.cargador},{a.totalBalas}";
        datos.Add(par.Key, valor);
        Debug.Log($"[ArmaManager] Preparando para guardar: {par.Key} = {valor}");
    }

    var request = new UpdateUserDataRequest
    {
        Data = datos
    };

    PlayFabClientAPI.UpdateUserData(request,
        result => Debug.Log("[ArmaManager] Armas guardadas en PlayFab correctamente."),
        error => Debug.LogError("[ArmaManager] Error al guardar armas: " + error.GenerateErrorReport());
}
}
```

Este método como anteriormente explique guarda los datos con un string separados por comas, esto es para guardar los datos o actualizarlos porque los superpone. Utilizó un `Dictionary<string, string>` porque PlayFab (PlayerData) funciona igual.

La estructura se puede crear desde dentro de PlayFab, pero en este videojuego está automatizado para que, cuando no se encuentra una key, se genere automáticamente y se le asigne un valor por defecto.

El valor por defecto se asigna en el método `ProcesarDatosArmas` (anteriormente visto), que al no encontrar un arma, llama al método `InicializarArmaPorDefecto`, al cual se le pasa como parámetro el nombre del arma. Este método, a su vez, llama a otro método encargado de buscar el arma que falta utilizando los datos base, y luego la añade al diccionario `armasJugador`.

Después de que este arma se ha añadido al diccionario, el método `InicializarArmaPorDefecto` llama a `GuardarArmasEnPlayFab`. Este último método guarda todas las armas que se encuentran en el diccionario, y gracias a eso se crea automáticamente la estructura correspondiente en PlayFab, si no existía previamente.

Por otro lado, tanto la key PlayerMoney como Kills se actualizan desde otros métodos específicos dentro del código, ya que no forman parte de la lógica de inicialización automática de armas.

Esta manera me ha resultado bastante práctica, cuando se trata de guardar información sobre armas o estadísticas del jugador, ya que me permite mantener todos los datos de un arma agrupados bajo una sola clave. Además, como todos los valores se guardan como texto(string), se mantiene la compatibilidad con el sistema de almacenamiento de PlayFab sin necesidad de crear estructuras complejas.

14. Manual de Usuario.

Paso 1: instalado.

1.1 Una vez instalado *DeadZone Last Stand* lo primero que va a salir es esta pantalla:



Paso 2: Cambiar la ventana de registrarse.

2.1. Si es la primera vez que entras al videojuego, deberás crear una cuenta. Para ello, haz clic en el botón "Crear cuenta".

2.2. Si ya tienes una cuenta creada, puedes saltar directamente al paso 4.

Paso 3: Registrarse.

3.1. Después de hacer clic en "Crear cuenta", aparecerá la siguiente pantalla:



3.2. Completa los campos con tus datos y luego haz clic en el botón "Registrar". Puedes tomar como ejemplo los siguientes datos para rellenar el formulario:



Paso 4: Iniciar sesión.

4.1. Una vez registrado, el sistema te llevará automáticamente de vuelta a la ventana de inicio de sesión.

4.2. En esta pantalla, introduce los mismos datos con los que te registraste y haz clic en el botón "Login" para acceder al juego.



Paso 5: Menú Principal.

5.1. Una vez que hayas iniciado sesión correctamente, serás dirigido a la siguiente ventana:



5.2. Si lo que deseas es consultar el Ranking de jugadores, dirígete al paso 9.

Paso 6: Jugar y armería.

6.1. Para comenzar a jugar, haz clic en el botón "Jugar".

Una vez pulsado, serás llevado a la siguiente ventana.



6.2. En esta ventana podemos ver un número junto a un ícono de moneda, que representa la cantidad de materiales que tienes disponibles.

6.3. A su lado se encuentra el botón "Volver", que te regresará a la pantalla anterior.

6.4. En la parte inferior se muestran cuatro armas, cada una con su precio en materiales indicado debajo.

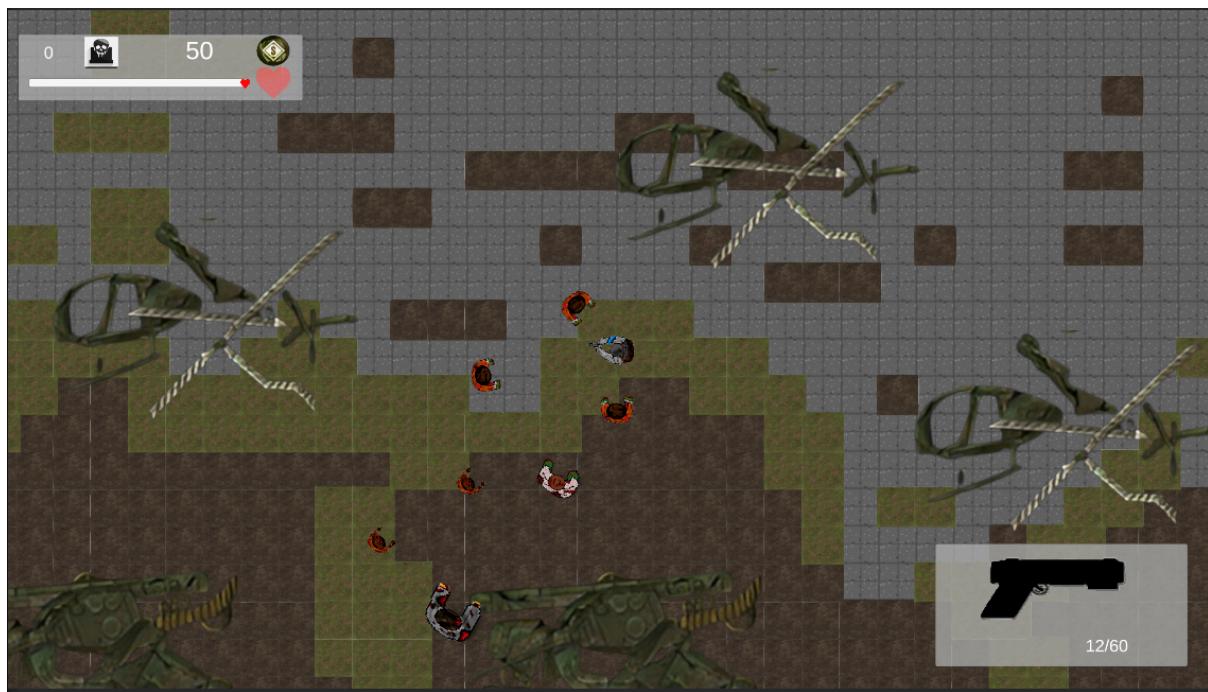
6.5. Si pulsas sobre la imagen de un arma, se descontará automáticamente la cantidad de materiales correspondiente y podrás usar esa arma en la partida, si no tienes suficientes materiales te quedarás en la misma pantalla y no pasará nada.

6.6. Importante: el arma que compres no se guarda permanentemente. Si el jugador muere, deberá volver a comprarla la próxima vez que quiera jugar.

6.7. Además, en esta ventana encontrarás el sistema de mejora de armas, el cual se explica en el paso 10.

Paso 7: Juego.

7.1. Una vez elegido arma nos llevará directamente al mapa de las ruinas el cual está infectado de zombies:



7.2. Como se puede ver en esta pantalla, hay un contador con una calavera al lado, el cual indica las kills (zombies eliminados durante la partida).

7.3. A su derecha está el mismo contador de materiales que vimos en la pantalla anterior, mostrando cuántos tiene actualmente el jugador.

7.4. Justo debajo aparece una barra de vida: si esta llega a 0, la partida terminará automáticamente.

7.5. En la esquina inferior derecha hay un panel que muestra el arma seleccionada y dos contadores:

- La imagen indica el arma equipada.
- El contador de la izquierda representa las balas en el cargador.
- El de la derecha muestra la munición total disponible.
- Cuando el contador izquierdo llegue a 0, debes presionar la tecla R para recargar el arma. Al hacerlo, el cargador se llenará consumiendo balas del contador derecho.

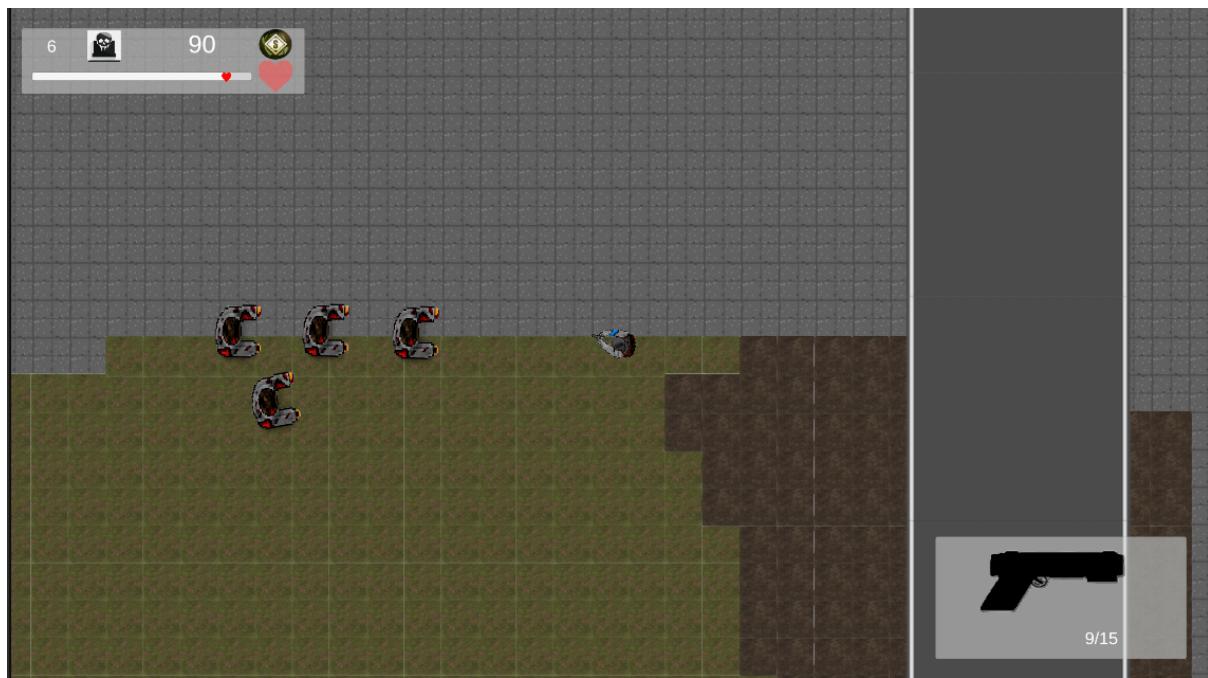
7.6.1. El movimiento del jugador se controla con las teclas W, A, S y D.

7.6.2. La dirección en la que apunta el personaje depende de la posición del ratón, y para disparar se usa el clic izquierdo.

7.7. Si una bala impacta a un zombie, le resta vida. Cuando su vida llega a 0, el zombie es eliminado y otorga un bonus aleatorio, que puede ser vida, materiales o munición.

7.8. Como se muestra en la captura anterior, existen cuatro tipos de zombies, cada uno con características especiales que deberás descubrir durante el juego.

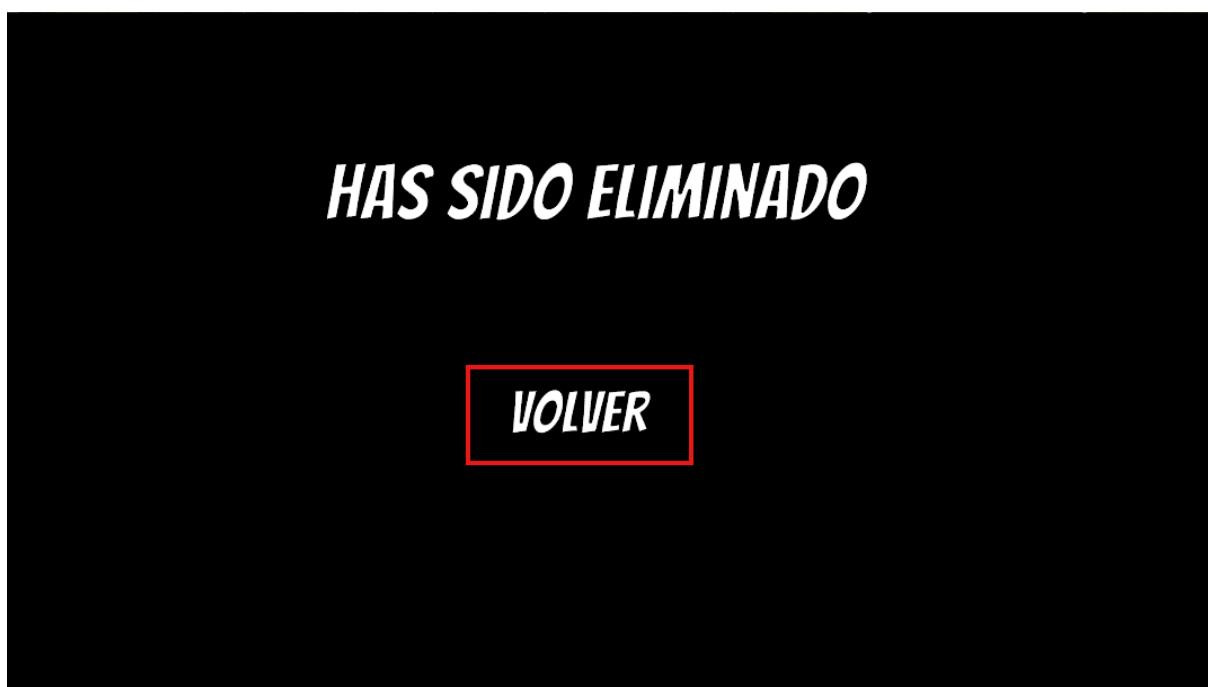
7.9. Por último, en el mapa hay objetos con los que puedes colisionar.



Paso 8: Ejemplo y fin de partida.

8.1. En esta captura se puede ver que eliminé a 6 zombies, gané materiales, además de que gasté munición y realicé una recarga.

8.2. Una vez que el jugador es eliminado, aparecerá una pantalla que indica el fin de la partida, junto con un botón de "Volver".



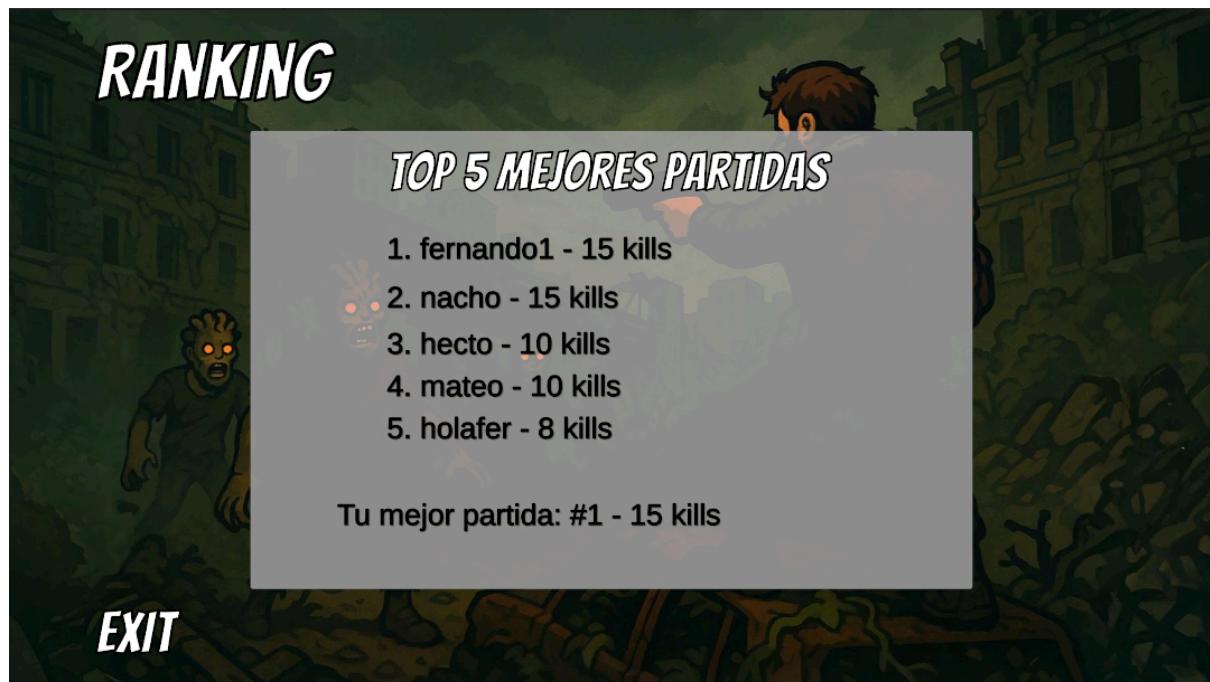
8.3. Este botón te llevará de nuevo a la pantalla que vimos en el paso 5.

Paso 9: Ver ranking.

9.1. Para ver el Top 5 hay que darle al botón de Ranking:



9.2. Una vez dado aparecerá esta pantalla:



9.3. Al darle a volver regresará a la pantalla del paso 5.

Paso 10: Mejorar el Arma.

10.1. Para poder mejorar un arma hay que darle al botón de jugar:



10.2. Una vez dado a jugar para mejorar el arma que deseas dale al botón que hay abajo del arma que pone mejorar y su costo.



10.3. Si el costo es superior a lo que tienes el botón no hará nada, si es inferior se mejorará el arma se gastara los materiales y además saldrá una estrella indicando que se ha mejorado:



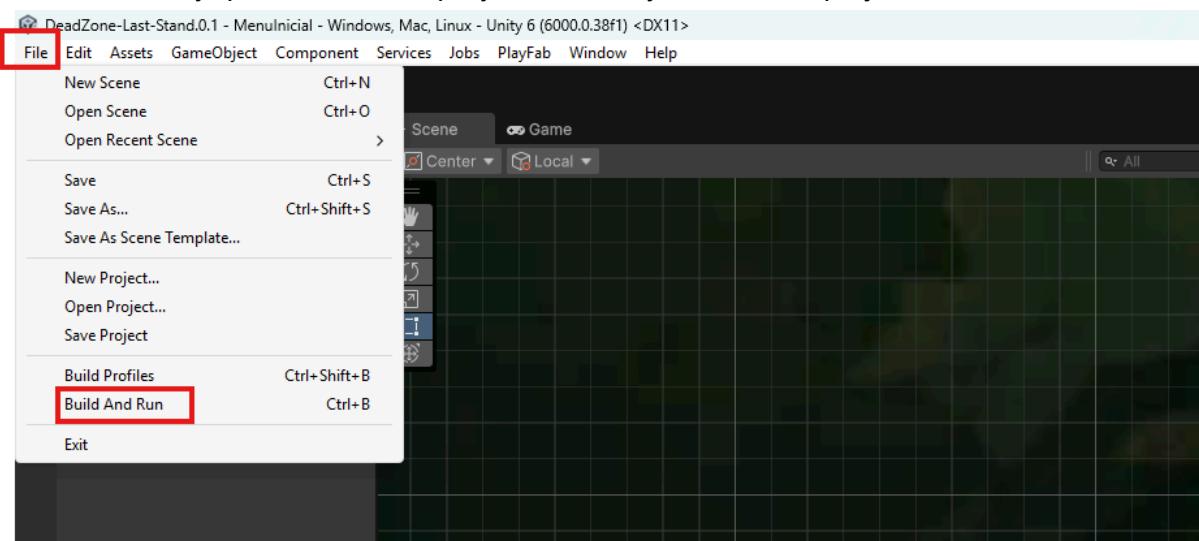
10.4. Esta mejora de arma si es permanente, no es igual que cuando compras un arma para una partida.

15. Manual de Configuración.

15.1. Configuración de despliegue.

La configuración para el despliegue del videojuego DeadZone Last Stand es la siguiente:

- Hay que comprobar si la misma aplicación está conectada a un game de playfab, si esta, hay que ir dentro del proyecto en unity a files build project and run:



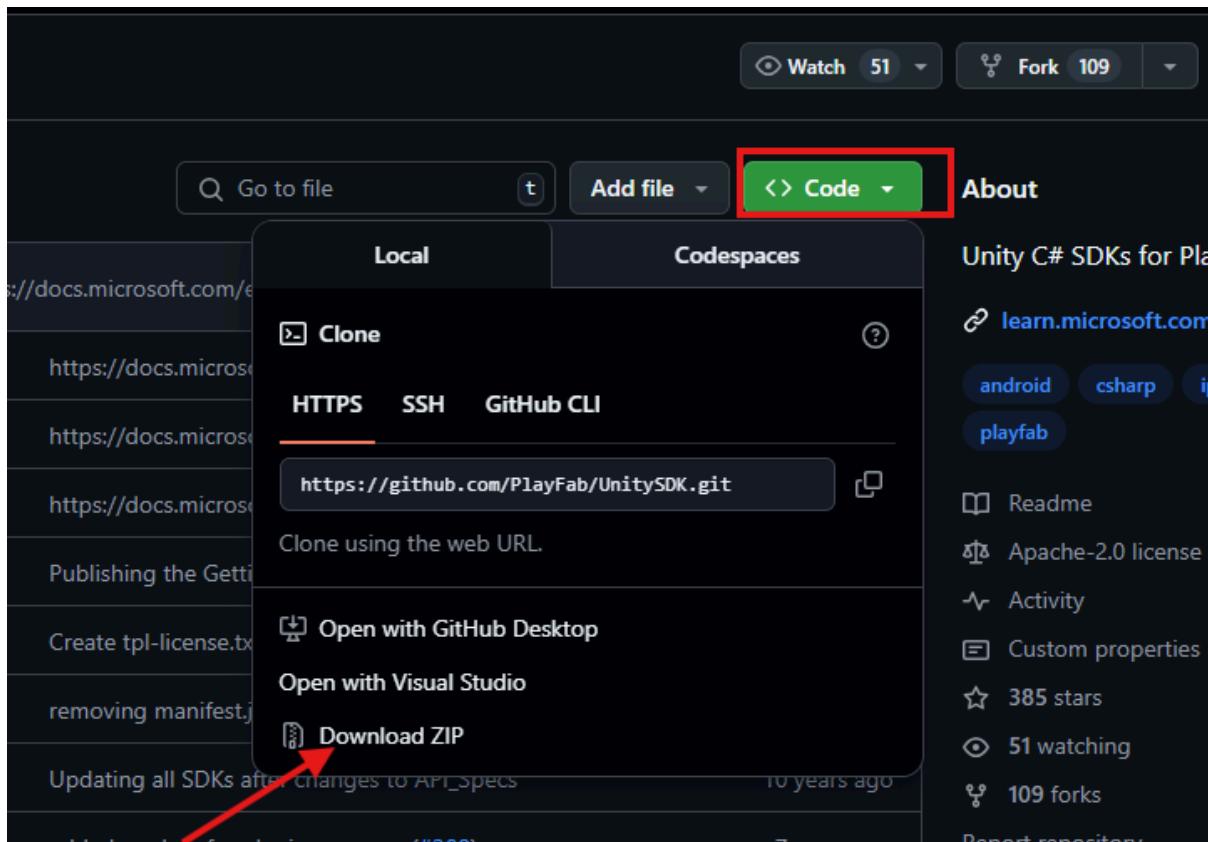
- Esto nos creará el videojuego con su ejecutable, Esa carpeta es la que hay que pasar a la hora del despliegue multiplataforma a los usuarios que deseen jugar, la forma más fácil de compartirlo es comprimiendo esa carpeta en un .zip y

subiéndose a un github en un repositorio público para que todo el mundo pueda acceder a él y pueda descargar el videojuego.

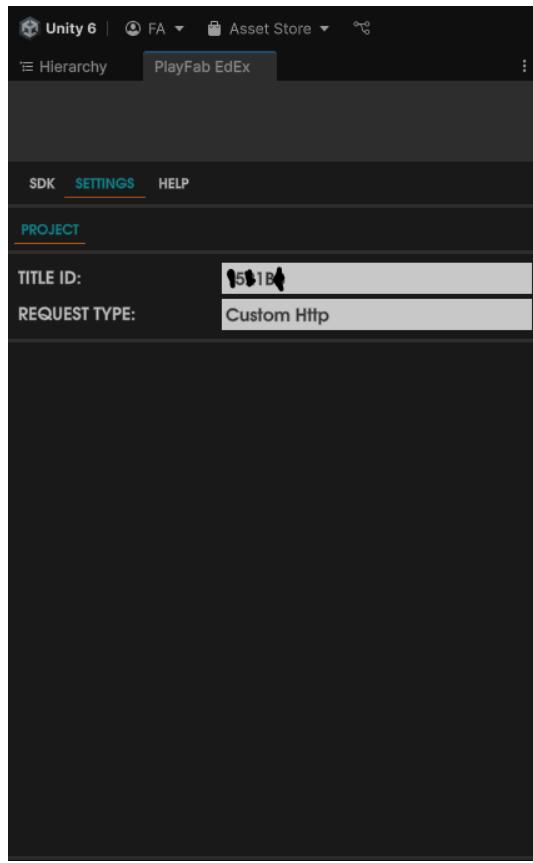
- Si no está conectado a PlayFab habrá que conectarlo y después hacer el build project and run explicado anteriormente.

Para conectar PlayFab a Unity hay que seguir los siguientes pasos:

1. Hay que descargar desde internet el SDK de PlayFab: Aquí se puede descargar <https://github.com/PlayFab/UnitySDK>



2. Una vez descargada se descomprime la carpeta, la carpeta descomprimida se arrastra al proyecto de Unity, y se importa y hecho esto saldrá el panel donde hay que llenar el id del game de PlayFab:



3. De donde sacas este ID, debes de tener una cuenta creada en PlayFab y un game el game viene ya creado por defecto al lado aparecerá el ID que buscas.

The screenshot shows the PlayFab Studio interface. At the top, there's a navigation bar with icons for notifications, account status, and a search bar labeled 'FE'. Below it is a header with 'My Studios and Titles' and a 'New studio' button. The main area lists studios, with one named 'estudioAmaya' highlighted by a red box. Inside this box, a game titled 'My Game' is shown with its ID '151B' circled in red at the bottom right. Other details like 'Development' and a dropdown menu are visible. At the bottom of the page, there's a 'Manage cookies' button.

15.1. Configuración de Administrador.

El administrador deberá tener una cuenta de PlayFab asociada al game del juego, el administrador dentro del game tiene muchas opciones de editar usuarios buscarlos ver lo

que cada usuario tiene en la base de datos etc. Ejemplos de pantallas de PlayFab donde el administrador más va a usar, por ejemplo buscar un player.

- Para buscar un player dentro de game le damos al apartado de players y le damos al botón Search:

Type	Name	Last login	Created	Country/region	VTD
👤	7803EE8BB5E57934 edu	2:42 PM	Today	Spain	\$0.00
👤	190FB0E83FB52386 fernando1	2:39 PM	1 day ago	Spain	\$0.00
👤	27C9E470A26A6E83 andres	2:29 PM	Today	Spain	\$0.00
👤	F9F10FBAACEFBDEC luis	1:27 PM	Today	Spain	\$0.00
👤	C2325A8638B1DFD nacho	11:22 AM	Today	Spain	\$0.00
👤	968DC4027EE3654E mateo	11:16 AM	Today	Spain	\$0.00

- Otro ejemplo sería ver los datos de un player, para ello pinchamos en el player y tenemos varias opciones para ver todo sobre ese jugador.

En Overview tenemos toda la inf del registro aparte podemos eliminar ese player y editarlo dándole a Save, Luego en Player Data se puede ver todo lo que guarda ese jugador:

Player data

	Key *	Value *	Permissions
<input type="button" value="Delete"/>	PlayerMoney	200	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>
<input type="button" value="Delete"/>	Pistola	2,13,10,14,60	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>
<input type="button" value="Delete"/>	Escopeta	1,25,6,6,30	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>
<input type="button" value="Delete"/>	Fusil	1,15,15,30,120	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>
<input type="button" value="Delete"/>	Francotirador	1,50,30,5,20	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>
<input type="button" value="Delete"/>	kills	16	<input type="button" value="Edit"/> Private <input type="button" value="Change"/>

16. Manual de Instalación.

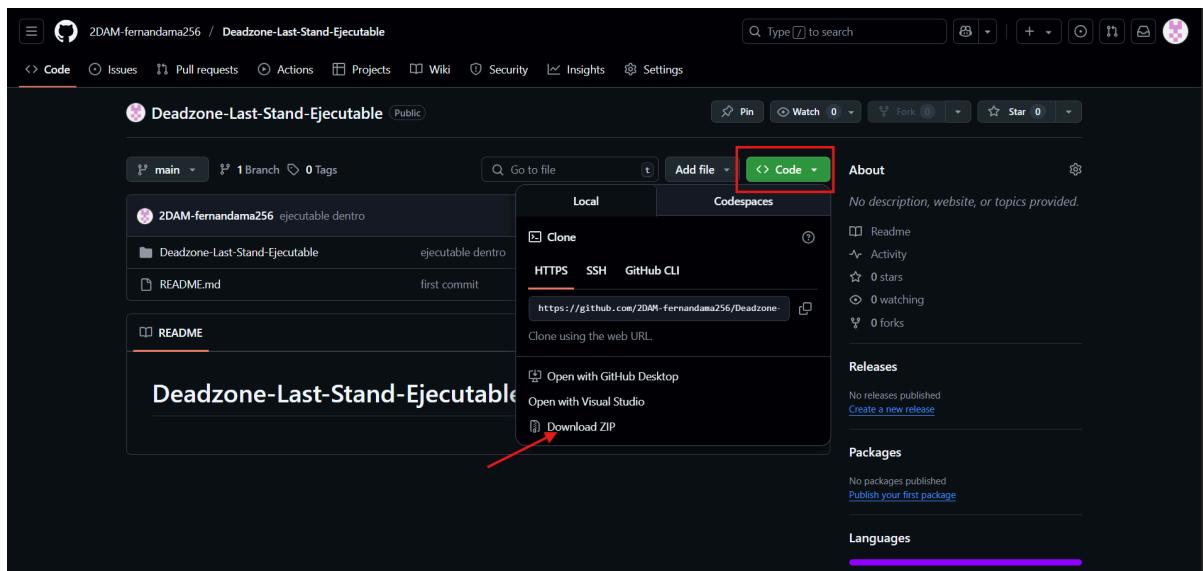
16.1. Requisitos del sistema operativo y de hardware:

Para jugar DeadZone Last Stand no necesitas un equipo muy potente. Con tener conexión a internet, un procesador Intel i3, gráfica integrada, al menos 4 GB de RAM y mínimo dos núcleos, es más que suficiente para disfrutarlo, estos son los mínimos lo recomendable sería tener estas características más superiores, para así disfrutar mejor del videojuego.

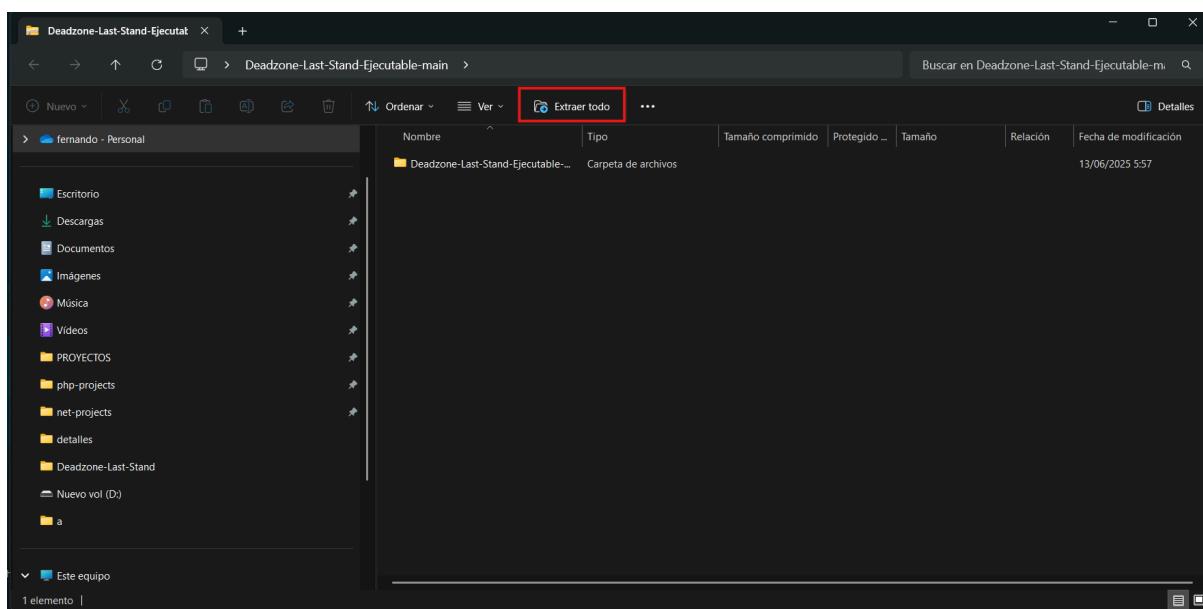
El juego está disponible solo para PC y requiere Windows 10 o superior para funcionar correctamente.

16.2. Instalación:

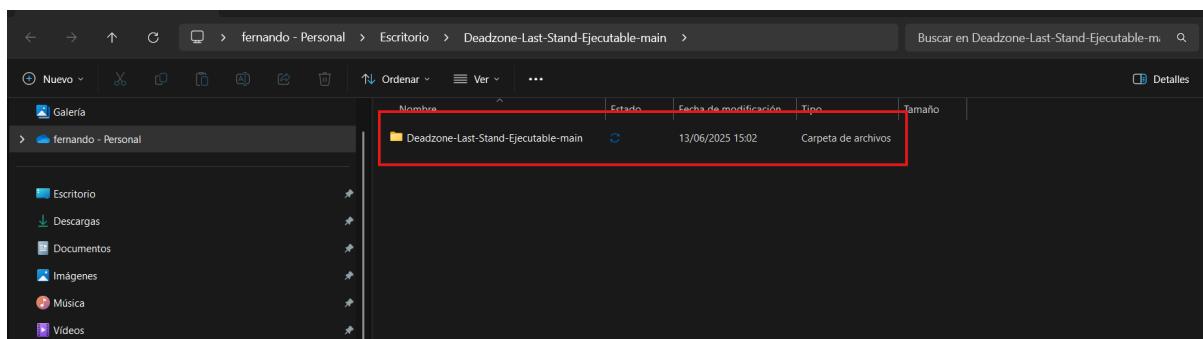
1. Para instalar DeadZone Last Stand, lo primero que hay que hacer es ir a este repositorio: <https://github.com/2DAM-fernandama256/Deadzone-Last-Stand-Ejecutable>, dentro de ese repositorio hay que darle al siguiente botón (el orden seria code y luego Download ZIP):



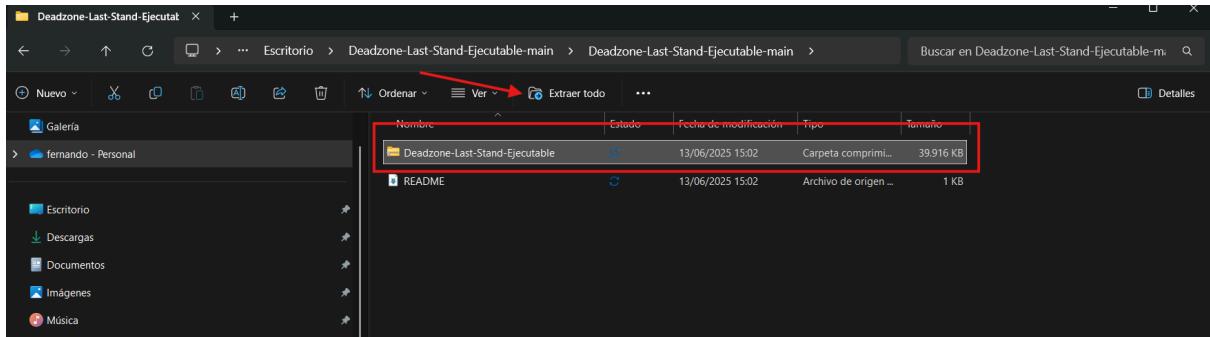
2. Una vez descargado el .zip hay que descomprimirlo, para eso tienes que abrir el .zip y darle a extraer todo:



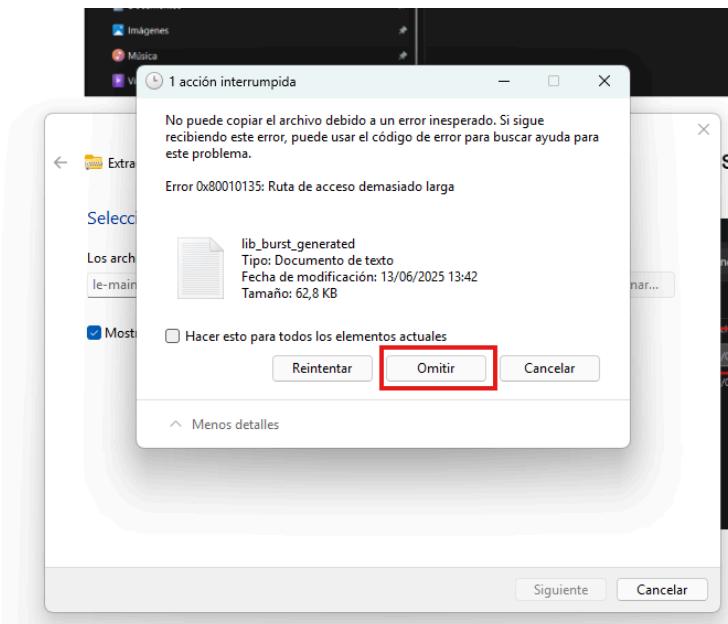
3. Cuando esté descomprimido, entra en esa carpeta y va a salir esto así que vuelve a entrar en la carpeta que sale:



4. Ahora dentro de esa carpeta encontraras otro .zip descomprimelo:

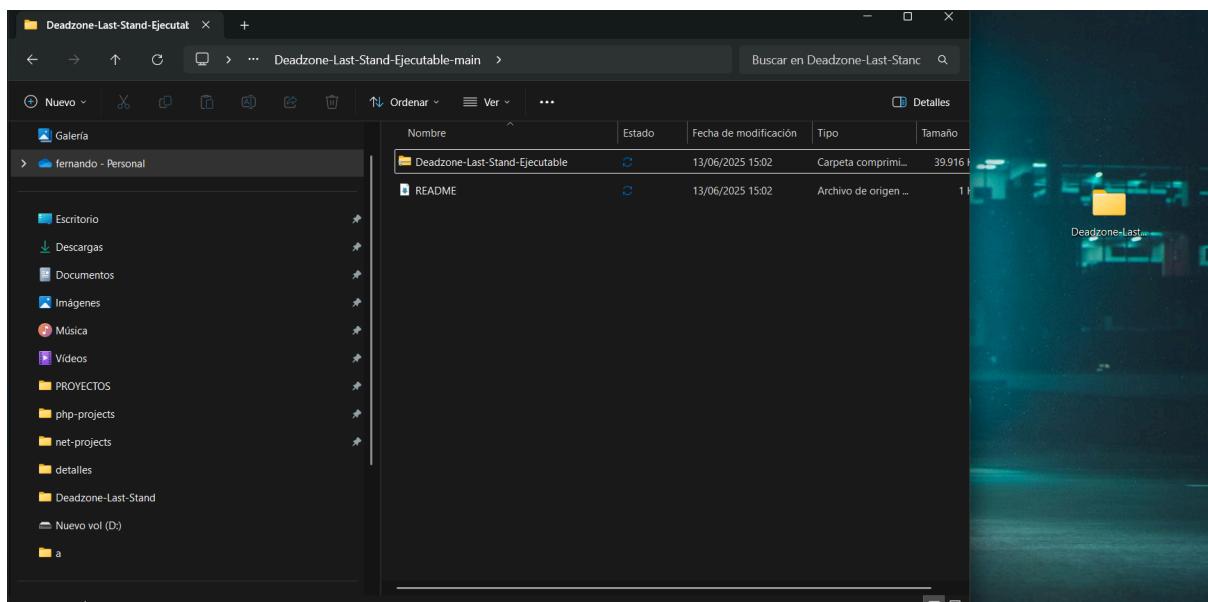


5. Al intentar descomprimirlo aparece el siguiente mensaje, hay que omitir:



Aparecerán otros dos mensajes más como este, simplemente hay que omitir y ya.

6. Una vez hecho esto podemos coger ese archivo descomprimido y sacarlo fuera de la carpeta en mi caso lo voy a dejar en el escritorio(lo arrastras y ya):



7. Tendrás tres carpetas 2 carpetas de ellas tendrán puesto en el nombre main, esas dos carpetas las puedes eliminar, la otra carpeta es la buena (la carpeta que tiene las cruces se pueden eliminar y la carpeta que está dentro de un cuadro es la que hay que dejar):

📁 Deadzone-Last-Stand-Ejecutable	🕒	13/06/2025 15:10	Carpeta de archivos
📁 Deadzone-Last-Stand-Ejecutable-main	🕒	13/06/2025 15:02	Carpeta de archivos
📁 Deadzone-Last-Stand-Ejecutable-main	🕒	13/06/2025 14:58	Carpeta comprimi...

8. La carpeta que se ha quedado abrela, te aparecerá solo una carpeta ábrela y te aparecerá todo esto:

Nombre	Estado	Fecha de modificación	Tipo	Tamaño
📁 D3D12	🕒	13/06/2025 15:10	Carpeta de archivos	
📁 DeadZone-Last-Stand.0.1_BurstDebugInfo...	🕒	13/06/2025 15:10	Carpeta de archivos	
📁 DeadZone-Last-Stand.0.1_Data	🕒	13/06/2025 15:12	Carpeta de archivos	
📁 MonoBleedingEdge	🕒	13/06/2025 15:12	Carpeta de archivos	
⚙️ DeadZone-Last-Stand.0.1	🕒	13/06/2025 15:10	Aplicación	657 KB
⚙️ UnityCrashHandler64	🕒	13/06/2025 15:10	Aplicación	1.496 KB
UnityPlayer.dll	🕒	13/06/2025 15:10	Extensión de la ap...	32.827 KB

Lo resultado es el ejecutable si lo abres te entra directamente en el videojuego, por lo tanto para acceder al videojuego tendrás que hacerlo siempre desde ese ejecutable, NO SAQUES EL EJECUTABLE DE ESA CARPETA, SI NO NO FUNCIONARA.

17. Retos.

A lo largo de la creación del videojuego me he encontrado con diferentes retos:

- El logueo y guardado de datos con PlayFab, a lo largo de la creación de mi proyecto he tenido varios retos por parte de PlayFab ya que para mi es algo nuevo y no sabia como poder guardar las armas y las mejoras de las mismas, ya que Player Data es como un diccionario de key(String) value(String), Al final la solución fue guardar ese String como un CSV pasando los valores a string y además separándolos entre comas „, „, para así luego leerlo con un split(„, „)
- La importación del NavMesh: Al intentar importar el NavMesh desde Unity no me funcionaba, no me calculaba la superficie en la que el zombie se debería de mover, la solución fue investigar por internet hasta darme cuenta de que Unity te importa el NavMesh pero para videojuegos 3D por lo tanto lo tuve que importar manualmente un NavMesh que estaba hecho para el 2D desde github, lo descargue utilice los script importados y me funciono todo.
- La navegación entre escenas, la navegación entre escenas en unity tiene una particularidad y es que al cambiar de escena la escena anterior los prefabs quedan totalmente eliminados, la solución a esto es: este código:

```
④ Mensaje de Unity | 0 referencias
private void Awake()
{
    if (Instance == null)
    {
        Instance = this;
        DontDestroyOnLoad(gameObject);
    }
    else
    {
        Destroy(gameObject); // Evita duplicados
    }
}
```

Este código lo que hace es crear instancias de esa clase por lo tanto no se destruye, además evita duplicados.

- Para poder hacer un Ranking con PlayFab, no puedes hacerlo desde player data, porque el player data es Data privada de un jugador, por lo tanto, la solución fue guardarla en Player Statistics para guardar las kills(la mejor partida del jugador) y además añadir en el registrar un campo de nombre para que salga en el ranking, este campo de nombre lo tuve que guardar en PlayFab en DisplayName.

18. Posibles mejoras.

Las mejoras que le añadiría a mi juego serían:

- Cambiar la vista del juego en 3D con más animaciones.
- Acentuar más la diferenciación entre armas.
- Tener más mapas diferentes y en 3D con la vista desde arriba.
- Tener una mejor historia que respalde al videojuego.
- Poder agregar amigos y tener un ranking entre amigos.

19. Conclusión.

Hacer este videojuego con Unity ha sido una experiencia muy divertida y también un gran reto, ya que he encontrado muchas funciones de Unity que no conocía y he tenido que aprender a usarlas sobre la marcha.

Además, trabajar con PlayFab ha sido todo un reto. Nunca había utilizado esta herramienta y me ha costado entender cómo funciona, sobre todo todo lo relacionado con el Player Data, registro e iniciar sesión. Aun así, gracias a esto he aprendido muchísimo,sobre Unity, y además sobre cómo funciona un videojuego por dentro.

En general, este proyecto me ha servido para mejorar como programador, aprender cosas nuevas y darme cuenta de que cualquier proyecto que se me ocurra lo puedo llegar a desarrollar. Ha sido complicado, pero al final muy satisfactorio ver todo el trabajo hecho y funcionando.

20. Repositorio con el código fuente.

<https://github.com/2DAM-fernandama256/Deadzone-Last-Stand.git>