

3Dslug



19 de junio de 2022

JORGE ARENAS SORIANO

Contenido

[Presentación del proyecto 2](#_Toc103701605)

[¿En qué consiste? 2](#_Toc103701606)

[Opciones de dificultad. 2](#_Toc103701607)

[Estudio de mercado. 2](#_Toc103701608)

[Valor del producto. 4](#_Toc103701609)

[Planificación de tareas y estimación de costes. 5](#_Toc103701610)

[Planificación y organización de tareas. 5](#_Toc103701611)

[Uso de gitflow como flujo de trabajo. 6](#_Toc103701612)

[Estimación de costes. 6](#_Toc103701613)

[Financiación del proyecto 7](#_Toc103701614)

[Herramientas usadas 7](#_Toc103701615)

[Gestión de riesgos 8](#_Toc103701616)

[Análisis de la solución. 9](#_Toc103701617)

[Análisis de requisitos. 9](#_Toc103701618)

[Requisitos funcionales 9](#_Toc103701619)

[Requisitos no funcionales 10](#_Toc103701620)

[Requisitos de información. 10](#_Toc103701621)

[Análisis de escenarios (Casos de uso) 11](#_Toc103701622)

[Diseño de la solución 12](#_Toc103701623)

[Controles de usuario 13](#_Toc103701624)

[Diseño gráfico 13](#_Toc103701625)

[Impacto del diseño en los usuarios 14](#_Toc103701626)

[Diagrama de clases 14](#_Toc103701627)

[Persistencia de la información 16](#_Toc103701628)

[Decisión del tipo de almacenamiento de datos. 17](#_Toc103701629)

# Presentación del proyecto

## ¿En qué consiste?

Un videojuego de supervivencia por rondas en las cuales aparecerán enemigos en función de la ronda en la que se encuentre el jugador y la dificultad escogida por el mismo. Por tanto, el objetivo de este juego es sobrevivir el mayor número de rondas posibles llegando a la ronda número 100 para ganar. El jugador dispone de una barra de salud de 0 a 100 que empezará al completo. Cuando los enemigos golpeen al jugador este perderá vida, en función del daño que cause, que podrá recuperar cogiendo los diferentes botiquines que aparecerán en puntos preestablecidos del mapa. Si la salud del jugador llega a 0, la partida termina automáticamente indicando el número de rondas que has superado. Así mismo, el jugador dispondrá también de una bolsa de granadas. Las granadas se obtienen con una probabilidad del 10% al matar a un enemigo y serán lanzadas por el jugador en la dirección a la que mire el personaje, eliminando a todo enemigo dentro del rango cubierto por la explosión eléctrica.

Además de esto, se pretende añadir una funcionalidad de compra de armas por puntos conseguidos al eliminar enemigos. Se colocarán siluetas de diferentes armas por todo el mapa de forma que, al acercarse el jugador, podrá comprarlas si dispone de los puntos suficientes. Facilitando de esta forma la supervivencia frente a enemigos cada vez más fuertes y diversos.

### Opciones de dificultad.

El juego dispone de 3 modos de dificultad (Fácil, Medio y Difícil) que funcionarán de la siguiente manera:

* **Modo Fácil:** Se generarán enemigos entre el número de ronda y el doble. Por ejemplo: en la ronda 5 se generarán entre 5 y 10 enemigos. El precio de las armas será el preestablecido.
* **Modo Medio:** Se generarán enemigos entre el doble del número de ronda y el cuádruple. Por ejemplo: en la ronda 5 se generarán entre 10 y 20 enemigos. El precio de las armas será un 25% más caro.
* **Modo Difícil:** Se generarán enemigos entre el triple del número de ronda y el séxtuple. Por ejemplo: en la ronda 5 se generarán entre 15 y 30 enemigos. El precio de las armas será un 50% más caro.

En todos los modos de dificultad la salud y el daño de los enemigos serán el mismo, tan solo cambiará el número de enemigos generados y el precio de las armas.

## Estudio de mercado.

Un estudio de mercado consiste en una iniciativa empresarial con el fin de hacerse una idea sobre la viabilidad comercial de una actividad económica. Este estudio tiene un objetivo económico, es decir, su objetivo es generar beneficio económico con la actividad económica cuya validez queremos probar en el mercado.

Esta investigación busca anticipar la respuesta de los clientes potenciales y la competencia ante un producto o servicio concreto para probar los productos o servicios, saber cómo mejorarlos, de qué manera posicionarlos en el mercado, etc.

De esta manera, con un estudio de mercado bien realizado, se conocerá el perfil y el comportamiento de los clientes, la situación del mercado o industria a la que va dedicada el producto, cómo trabaja la competencia, etc. Incluso se pueden llegar a descubrir nuevas necesidades de nuestro público objetivo en las que no se había pensado previamente. Un estudio de mercado se apoya sobre 4 pilares fundamentales:

* **La información del sector:** 3Dslug cae directamente en el sector de los videojuegos, concretamente en la parte del sector dedicada a teléfonos inteligentes.

El sector de los videojuegos es, a día de hoy, uno de los más importantes para la economía mundial debido al amplísimo público que obtiene, desde los más pequeños de la casa hasta, en muchos casos, los no tan pequeños.

Esta industria ha avanzado a pasos agigantados desde la creación del primer videojuego en el año 1952. Tiempo en el que los videojuegos eran proyectos tan sencillos como puede ser una línea que el jugador puede mover de un lado a otro de la pantalla con el objetivo de bloquear una pelota que va rebotando infinitamente por los bordes de la misma.

Actualmente, en 2022, la industria del videojuego ha crecido tanto, que son capaces de cautivar a cualquier usuario con su calidad de gráficos, sus mecánicas, el increíble realismo que algunos presentan, el sentimiento que desarrollan sus personajes y una capacidad de inmersión cada día más impresionante.

Por desgracia, la mayoría de teléfonos inteligentes no están capacitados para ejecutar, con un rendimiento óptimo, unos juegos de tal calidad. Por tanto, a la hora de desarrollar videojuegos para un teléfono inteligente, existirán una serie de limitaciones que obligará a los desarrolladores a trabajar en proyectos menos ambiciosos. Aun así, con **3Dslug** intentaremos romper un poco esos límites creando un estilo de juego nunca antes visto en estos dispositivos. Para que el proyecto sea rentable, es necesario que funcione en el mayor número de dispositivos posible, por lo que el principal objetivo será optimizar el rendimiento todo lo posible.

* **Conocer al público objetivo (target):** Para crear un público objetivo, normalmente se toman en cuenta su edad, sexo, ubicación, formación educativa, poder adquisitivo, clase social y hábitos de consumo.

En el caso de **3Dslug**, al tratarse de un videojuego, estará destinado a todos los públicos con la única limitación del código PEGI, un mecanismo de autorregulación diseñado por la industria para dotar a sus productos de información orientativa sobre la edad adecuada para su consumo integrado por dos tipos de iconos descriptores, uno relativo a la edad recomendada y otro al contenido específico susceptible de análisis.

**3Dslug** presentará el icono de violencia y miedo, por lo que se empleará el código PEGI-16 para restringir el videojuego a menores de 16 años.

* **Conocer a la competencia actual:** La competencia en la industria del videojuego es muy elevada, puesto que el número de videojuegos que existen a día de hoy roza el infinito. Por suerte, la mayor competencia al estilo de juego de **3Dslug** pertenece al sector de las consolas y el PC, por lo que en teléfonos inteligentes la competencia es mucho menor.

Un videojuego de consolas muy famoso que podría hacer competencia directa con **3Dslug** es “Call of Duty” en su modo “zombies”, puesto que el estilo de juego es extremadamente similar (supervivencia por rondas con compra de armas y mejora de atributos).

La idea de **3Dslug** es desarrollar una funcionalidad muy similar a los zombies del call of duty para teléfonos inteligentes, de forma que un jugador aficionado a este estilo de juego, pueda llevar su experiencia y diversión allá donde quiera.

* **Análisis DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades):** Se trata de una sencilla herramienta de análisis estratégico muy extendida en la toma de decisiones de todo tipo de organizaciones y empresas. Sus siglas significan:
  + **Debilidades**: Se trata de las desventajas que tenemos respecto a nuestros competidores, cosas que deberíamos mejorar.
  + **Amenazas**: Entrada de nuevos competidores al mercado o un mercado saturado son amenazas para la empresa.
  + **Fortalezas**: Qué es lo que hacemos mejor que nuestros competidores, en qué nos diferenciamos de ellos.
  + **Oportunidades**: Nichos del mercado no ocupados, estrategias para mejorar la eficiencia y reducir los costes.

Se ha realizado un análisis DAFO sobre **3Dslug** obteniendo los siguientes resultados:

Una gran desventaja es la falta de conocimiento sobre diseño visual tanto de personajes, como de objetos y animaciones. Por lo que se hará uso de Assets públicos creados por diseñadores y publicados en la web de Unity que incluirán diseños y animaciones. Aunque en ocasiones, será necesario diseñar objetos y animaciones que no estén preestablecidas en los assets para hacer uso de ellos.

Debido a que la competencia pertenece al sector de las consolas, las amenazas para **3Dslug** apenas influyen y goza de exclusividad en teléfonos inteligentes por el momento. Lo cual es a la vez su mayor fortaleza y oportunidad. El mayor problema al que se enfrenta **3Dslug** es, sin duda, la variedad de dispositivos móviles que existen y la dificultad que tendrán la mayoría de estos para otorgar un rendimiento óptimo al juego.

## Valor del producto.

El valor de mercado es el valor que un producto (bien o servicio) tiene como consecuencia de la aplicación de la ley de la oferta y la demanda, es decir, lo que normalmente pagaría un comprador por ese producto en condiciones normales de mercado.

Actualmente el mercado de los videojuegos es muy variado entre dispositivos móviles y consolas. El precio de un videojuego de la nueva generación de consolas ronda los 80€, sin embargo, en dispositivos móviles rara vez alcanzan los 10€.

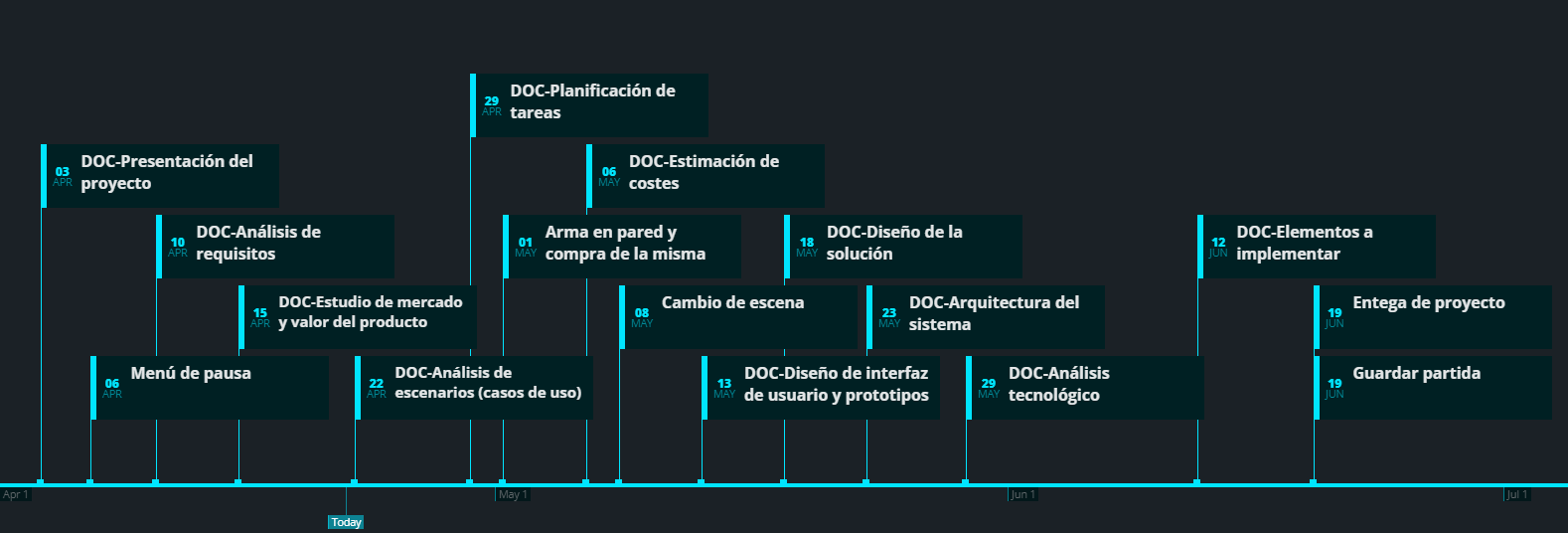
Por otro lado, como se pudo observar en el análisis DAFO, **3Dslug** luchará contra el problema del rendimiento en los diferentes dispositivos móviles, por lo que existirán muchos usuarios que quieran jugarlo pero, sin embargo, no puedan. Por este motivo, **3Dslug** verá reducido su público a usuarios con un poder adquisitivo ligeramente mayor.

Debido a esto último, el precio que se asignará a **3Dslug** será de 0’99€ con la intención de obtener el mayor número de ventas posible. Es bastante probable que con el precio establecido no se alcance una cantidad suficiente para compensar los gastos, por lo que **3Dslug** tratará de negociar con otras empresas o videojuegos para mostrar su publicidad durante las partidas.

Por otro lado, sabemos que la publicidad puede llegar a ser molesta para algunos usuarios, por lo que trataremos de importar la posibilidad de eliminarlos de forma definitiva mediante un pago extra de 3’99€, quedando de esta manera un precio final del juego completo sin anuncios de 4’98€

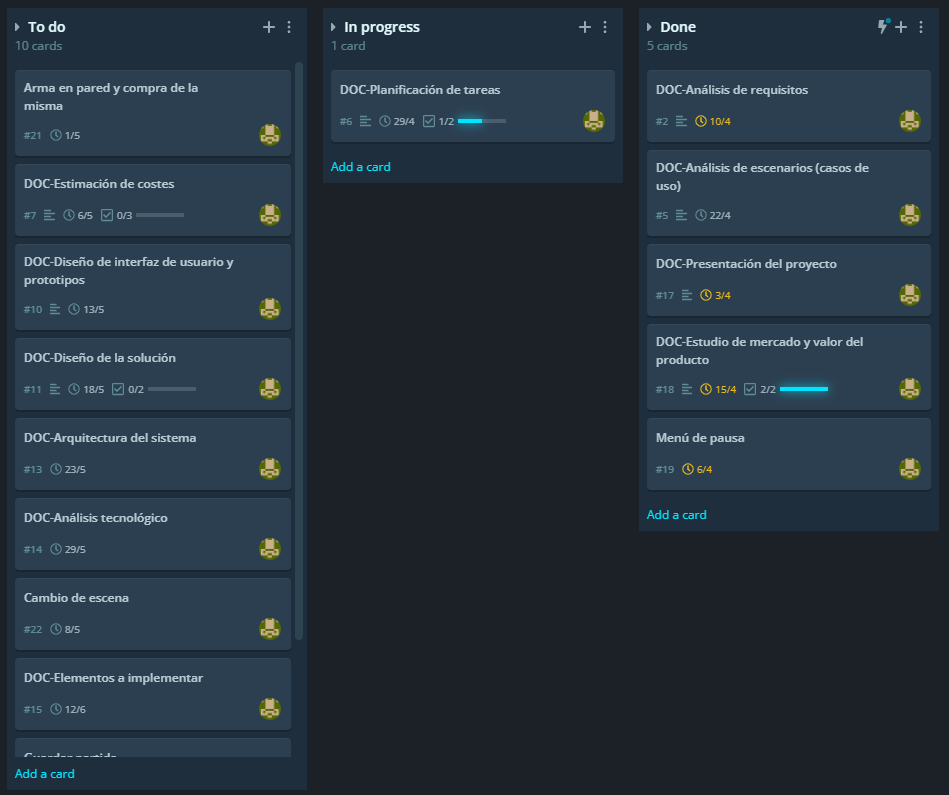
# Planificación de tareas y estimación de costes.

## Planificación y organización de tareas.

Para la planificación y organización de tareas se ha utilizado GitKraken boards, que ofrece la posibilidad de crear un tablero Kanban para poder organizar las tareas del proyecto. Además, estas tareas se pueden descomponer en tareas más pequeñas en modo de CheckList. GitKraken otorga la posibilidad de establecer una fecha en la que terminar cada tarea, pudiendo luego acceder a una vista Timeline en la que se observa de forma muy clara como van a avanzar nuestras tareas en el tiempo. Este es el timeline para el proyecto **3Dslug**:

En la imagen se puede observar como algunas tareas han pasado de su fecha de entrega. Por ahora todas las tareas se han entregado a tiempo y ninguna está retrasada.

Estas tareas se representan además en un tablero dividido en 3 columnas, las tareas pendientes de hacer (To do), las que están en proceso (In progress), y las que ya están realizadas (Done). Presentándose el siguiente tablero:



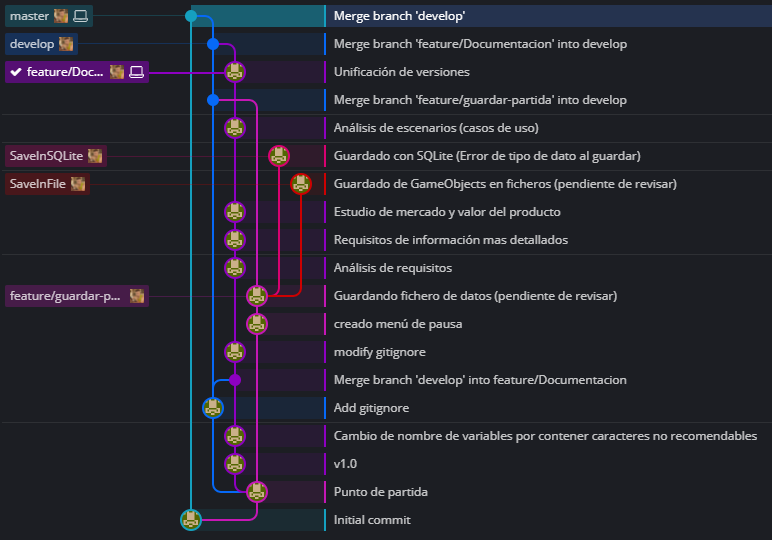
Tambien se puede visualizar el tablero en su versión pública en el repositorio de GitHub del proyecto. <https://github.com/Jarenass97/3DSlug/projects/2>

### Uso de gitflow como flujo de trabajo.

Gitflow es un modelo alternativo de creación de ramas en Git en el que se utilizan ramas de función y varias ramas principales. En comparación con el desarrollo basado en troncos, Gitflow tiene diversas ramas de más duración y mayores confirmaciones. Según este modelo, los desarrolladores crean una rama de función y retrasan su fusión con la rama principal del tronco hasta que la función está completa. Estas ramas de función de larga duración requieren más colaboración para la fusión y tienen mayor riesgo de desviarse de la rama troncal. También pueden introducir actualizaciones conflictivas.

Gitflow puede utilizarse en proyectos que tienen un ciclo de publicación programado, así como para la práctica recomendada de DevOps de entrega continua. Este flujo de trabajo no añade ningún concepto o comando nuevo, aparte de los que se necesitan para el flujo de trabajo de ramas de función. Lo que hace es asignar funciones muy específicas a las distintas ramas y definir cómo y cuándo deben estas interactuar.

En el repositorio de **3Dslug** se utiliza este flujo de trabajo:



Se pueden observar las ramas master y develop y como principalmente abren o cierran otras ramas, siendo muy limitado el uso de commits.

## Estimación de costes.

La estimación de costes es un proceso que consiste en desarrollar una aproximación a los recursos monetarios necesarios para completar las actividades del proyecto. Se trata de una predicción basada en la información que tenemos disponible en un momento dado.

En el desarrollo de software es muy complicado determinar el tiempo y esfuerzo que tomará realizar un proyecto. Esto se debe a que la estimación de costes de software no es nada fácil y los seres humanos somos bastante malos prediciendo resultados absolutos. No hay dos proyectos software iguales, por lo que no hay una guía que seguir acerca de cómo realizar estas estimaciones. Hay que tener en cuenta la cantidad de parámetros que conforman la existencia de los proyectos software, frecuentemente un problema que a priori parece sencillo, se vuelve mucho más complejo cuando se aborda desde el punto de vista técnico.

Las variables que se combinan en el dispendio final para la creación de un juego son casi siempre las mismas: desarrollo (incluye herramientas como hardware y software), márketing/publicidad, contratación de personal y distribución. Éstas son, habitualmente, las 4 constantes que suman al desembolso total que han de hacer las compañías para alumbrar un título.

Sin embargo, es muy raro que las mismas nos ofrezcan datos "oficiales" de los costes y, de hacerlo, es poco común que éstos incluyan esas 4 variables. Es por esta razón que es harto complejo el dar una respuesta exacta, contundente e informada a una pregunta tan sencilla como ¿cuánto cuesta hacer un videojuego?

De media, un videojuego puede suponer una partida de 13 a 52 millones de euros para las cuatro variables nombradas anteriormente, dependiendo de la relevancia del producto. Pero esas estimaciones están hechas basadas en equipos de desarrollo profesionales y experimentados. **3Dslug** carece de esa experiencia y, por tanto, no puede apuntar tan alto.

Es por eso que se aprovecha al máximo la disponibilidad de assets gratuitos para desarrollar el videojuego gastando lo menos posible.

**3Dslug** es desarrollado por una sola persona que estima unos beneficios iguales o superiores a 20€ por cada hora dedicada al desarrollo. Se estima una duración de 90 horas para el desarrollo más 35 horas de documentación e investigación debido a la poca experiencia en el sector, siendo un total de 125 horas. Por tanto, puesto que todos los diseños y software necesarios para el desarrollo serán gratuitos, los costes a cubrir ascienden a 125 horas x 20€ = 2500€

A este valor estimado se añadirán también los costes de despliegue en la Play Store de Google.

### Financiación del proyecto

Para financiar **3Dslug**, puesto que el único coste estimado es el sueldo del desarrollador, será financiado a través de las ventas realizadas en la play store de google y de los pagos por publicidad de las empresas interesadas. Este método de financiación puede abocar al fracaso al depender en gran parte de la captación de compradores de manera que es posible no cubrir los costes estimados.

## Herramientas usadas

Las herramientas usadas para el desarrollo de **3Dslug** serán:

* Para el diseño de escenas del videojuego y sus personajes, El motor gráfico de Unity, una herramienta de desarrollo de videojuegos creada por la empresa Unity Technologies.
* Para el desarrollo de código se utilizará visual studio 2022, el mejor IDE completo para desarrolladores de c# en Windows. Completamente equipado con una buena matriz de herramientas y características para elevar y mejorar todas las etapas del desarrollo de software.
* Para las tareas de ofimática, como la documentación, se emplea Microsoft Office 2016, un conjunto de aplicaciones que realizan tareas ofimáticas, es decir, que permiten automatizar y perfeccionar las actividades habituales de una oficina.
* Para la ejecución y prueba del videojuego se utilizará un Xiaomi 11 con Android 12 como S.O.
* Para el control de código fuente se hará uso de Git y GitHub. Para gestionar estas herramientas se utilizará GitKraken, que además incorpora GitKraken Boards para gestionar las incidencias a través de un tablero kanban donde ver las tareas por hacer, en proceso y finalizadas.

## Gestión de riesgos

Se entiende por prevención el conjunto de actividades o medidas adoptadas en todas las fases de la actividad de la empresa con el fin de evitar o disminuir los riesgos derivados del trabajo. Estos riesgos del trabajo son riesgos laborales, que es la posibilidad de que un trabajador sufra un determinado daño derivado del trabajo. Estos daños pueden ser enfermedades, patologías o lesiones. Para más información acudir a la Ley 31/1995, de 8 de noviembre de 1995, de prevención de riesgos laborales o LPRL.

En este caso, al ser una empresa de informática hay que prestar especial atención a:

* Dolores de espalda y otros trastornos musculo esqueléticos.
* Fatiga visual.

La prevención de riesgos en **3Dslug** se centrará principalmente en:

* Tener un puesto de trabajo adecuado, silla adecuada, escritorio a altura correcta, teclado separado del monitor, buena iluminación y ventilación. La luz debe ser en su mayoría natural y debe entrar por un lado del trabajador, ya que es perjudicial que la luz de por detrás y se refleje en el monitor
* Utilizar herramientas ergonómicas: pantallas con eye-care, teclados con una franja para apoyar adecuadamente la muñeca.
* Recomendación de levantarse y estirarse al menos una vez cada dos horas.

La gestión de riesgos es una actividad de protección dentro de la gestión de proyectos, encargada de identificar, mitigar y monitorizar los riesgos que pudieran afectar a la ejecución y viabilidad del proyecto.

Estos riesgos pueden ser de 3 tipos:

* **Riesgos del proyecto:** Ponen en peligro al plan, si se producen supondrá un mayor esfuerzo y dinero.
* **Riesgos técnicos:** Ponen en peligro la calidad del producto final.
* **Riesgos del negocio:** Ponen en peligro la realización del proyecto, si se cumplen, el proyecto se cancela.

Las principales causas que incrementan el nivel de riesgo en un proyecto son:

* Caer en alguno de los errores típicos.
* Desarrollar sin metodología.
* No tener una correcta estimación, evaluación y administración de los riesgos.

En **3Dslug**, se ha establecido una política clara en cuanto a los riesgos a los que nos enfrentamos:

|  |  |
| --- | --- |
| RIESGO | ACCIÓN |
| Falta de organización del equipo | Uso de GitKraken boards para organizar el trabajo y las tareas |
| Mala elección de herramientas | Se intentará estar al día sobre nuevas herramientas más útiles que las usadas actualmente. |
| Pérdida de código fuente o archivos | Se hará uso del Sistema de Gestión de Versiones Git para tener el proyecto completo en la nube de forma que siempre tengamos acceso a él a través de la web. También se instará a pushear los cambios de forma reiterada para evitar el mayor número de pérdidas. |
| Mala calidad del producto final | Se pospone el lanzamiento y se revisan todas las posibilidades de mejora posibles para otorgar el mejor resultado posible. |
| Retraso en una entrega | Se debe trabajar hasta terminarla. Habrá que descontar al cliente el precio ofertado debido al retraso. **3Dslug** cumplirá siempre su palabra. |
| Interfaz o código mal implementados | Se vuelve a diseñar e implementar aquello que sea necesario |
| El producto depende de las normativas del gobierno | Estar siempre al tanto de estas normativas para estar siempre bajo el amparo de la ley |

# Análisis de la solución.

## Análisis de requisitos.

### Requisitos funcionales

Un requisito funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas.

Los requisitos funcionales del usuario de **3Dslug** son los siguientes:

1. Manejar al personaje.
2. Posicionar la cámara.
3. Atacar a los enemigos. Habrá dos tipos de ataque (cuerpo a cuerpo y a distancia) que cambiarán en función del arma equipada.
4. Lanzar objetos arrojadizos.
5. Recoger botiquines dispersos por el mapa.
6. Comprar armas empleando los puntos conseguidos al abatir enemigos.
7. Cambiar entre un total de 2 armas que se guardarán en el inventario. Al comprar un arma nueva, esta ocupará el espacio de la que tenga equipada en ese momento, dejándola caer al suelo.
8. Pasar de un escenario a otro sobreviviendo a un número determinado de rondas (Lo decidiré mas adelante) con la salud, puntos y armas conseguidos hasta el momento, pero perdiendo todas las armas que queden tiradas en el suelo.
9. Recoger las armas que se hayan dejado caer al comprar nuevas, siguiendo la misma normativa de intercambio por el arma equipada.
10. Parar el juego en cualquier momento.
11. Guardar el estado de la partida en cualquier momento. De forma que al volver a abrir el juego, el usuario pueda retomar la partida por donde la dejó, sin perder sus puntos ni armas.

### Requisitos no funcionales

Un requisito no funcional o atributo de calidad es, en la ingeniería de sistemas y la ingeniería de software, un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que estos corresponden a los requisitos funcionales.

Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento. Por esto, suelen denominarse atributos de calidad de un sistema. Queda entonces el requisito no funcional, que son las restricciones o condiciones que impone el cliente al programa que necesita, por ejemplo, el tiempo de entrega del programa, el lenguaje o la cantidad de usuarios.

Entonces, los requisitos no funcionales de nuestra aplicación serán los siguientes:

1. El Front-End se desarrollará en Unity, utilizando assets públicos prediseñados y animaciones descargadas y diseñadas. Se utilizarán Prefabs con aquellos objetos que lo precisen, como pueden ser los enemigos, las armas o incluso el personaje jugador.
2. El Back-End se desarrollará en visual studio utilizando lenguaje C# apropiado para juegos de Unity, utilizando los métodos heredados de la clase MonoBehaviour. Estos scripts irán asignados a los GameObjects o Prefabs a los que vaya destinado su código.
3. Para guardar el estado de la partida, se generará un fichero de datos que será almacenado en una carpeta interna del móvil. (Estudiable utilizar firebase)
4. Se utilizará base de datos No-SQL con firebase para guardar un top de los usuarios que hayan conseguido las mejores puntuaciones.
5. Se pedirá un nombre al usuario al terminar la partida para guardarlo en el top de los mejores indicado anteriormente.

### Requisitos de información.

Los requisitos de información son aquellos que representan entidades e información relevante con las que el producto software va a operar.

En el caso de **3Dslug**, Estos requisitos son:

**Usuario →**

* Nombre otorgado por teclado al terminar la partida.
* Puntuación obtenida durante la partida, contando los puntos con los que cuente al terminar más la suma de todos los gastos que haya hecho el jugador al comprar armas. Por ejemplo, si tiene 6000 puntos acumulados al terminar la partida y ha gastado 24000 puntos en armas, el puntaje final será 6000+24000 = 30000 ptos.

**Partida →**

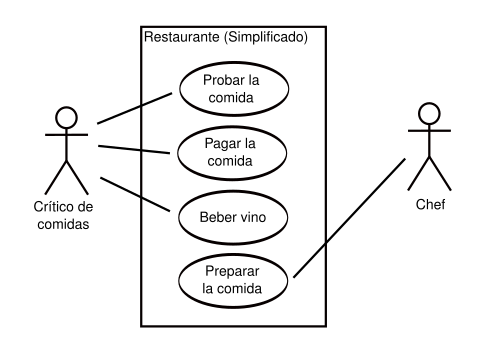
* Se guardará el estado de la escena en el momento de presionar la opción de guardar partida. De forma que, al volver a iniciar el juego, el usuario pueda retomar la partida tal como la dejó.

## Análisis de escenarios (Casos de uso)

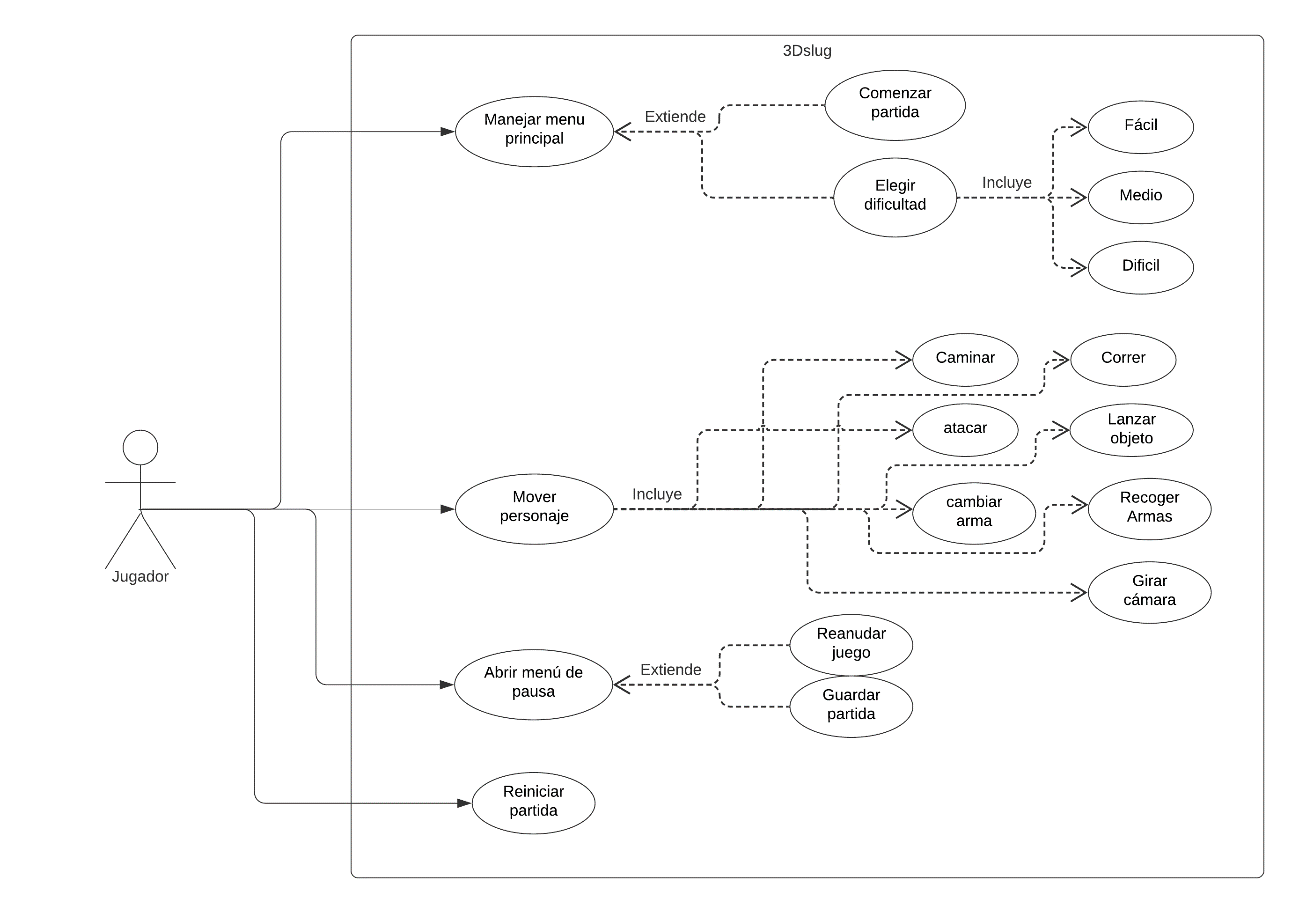
Un caso de uso es la descripción de una acción o actividad. Un diagrama de casos de uso es una descripción de las actividades que deberá realizar alguien o algo con el sistema desarrollado para llevar a cabo algún proceso

En el contexto de desarrollo de videojuegos, un diagrama de casos de uso representa a un sistema o subsistema como un conjunto de interacciones que se desarrollarán entre casos de uso y sus actores en respuesta a un evento que inicia el actor. En este caso, el Jugador.

Estos diagramas de uso son útiles para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. A continuación, un ejemplo de un diagrama de casos de uso básico.



A continuación, se expone el diagrama de casos de uso de **3Dslug**, que albergará los requisitos funcionales obtenidos en la fase de análisis de requisitos.



# Diseño de la solución

A continuación, hablaremos sobre el diseño de la solución. Este proceso juega un papel determinante en el desarrollo, ya que permite a los ingenieros de software producir diversos modelos que:

* Caracterizan la solución a implementar.
* Pueden ser analizados y evaluados con el fin de determinar si se satisfacen los requisitos
* Facilitan el examen y evaluación de alternativas.
* Sirven para planificar las siguientes actividades de desarrollo.

Este proceso es la actividad del ciclo de vida del software en la cual se analizan los requisitos de la solución para producir una descripción de la estructura interna del software que sirva de base para su construcción.

El objetivo de la interfaz de usuario es mantener la interacción con los mismos de la forma más atractiva, centrando el diseño en ellos. Las herramientas principales son los recursos gráficos, los pictogramas, los estereotipos y la simbología.

Es por ello, que toda interfaz de usuario sigue estos 6 principios:

* **Familiaridad con el usuario:** Se utilizan términos y conceptos que se toman de la experiencia de las personas que más utilizan el sistema
* **Consistencia:** La interfaz debe ser consistente en el sentido de que las operaciones comparables se activan de la misma forma
* **Mínima sorpresa:** El comportamiento del sistema no debe provocar sorpresa en los usuarios
* **Guía al usuario:** la interfaz debe dar feedback significativo al usuario
* **Diversidad de usuarios:** La interfaz debe proveer características de interacción apropiada para los diferentes tipos de usuarios

## Controles de usuario

El diseño tomado para los controles de **3Dslug** se basa en el uso de un joystick digital izquierdo para mover al personaje, un joystick digital derecho para girar la cámara y una serie de botones para las diferentes acciones posibles como: saltar, correr, cambiar de arma, golpear o disparar y lanzar objetos. Además, dispondrá de una parte del HUD informativos sobre el estado de la partida como puede ser el número de enemigos que quedan vivos, la ronda en la que se encuentra el jugador, su barra de vida, el número de granadas que tiene, etc.



## Diseño gráfico

El diseño gráfico tiene un papel fundamental en el proceso, ya que, a través de programas y un previo proceso de bocetaje, se plasman las ideas de manera precisa y clara, por lo que debe de comunicarse con los miembros de los diferentes departamentos para saber lo que han hecho en su trabajo, ligarlo a su invención creativa y reflejar la estética del tema para que todo esté en concordancia.

El diseño gráfico aporta los conocimientos necesarios para:

* que el usuario pueda tener una interactividad funcional con el producto a través de la creación de íconos, menús, etc.
* llevar a cabo la correcta creación o selección de una tipografía adecuada que cumpla con los criterios técnico-visuales y que sea del agrado y comprensión del usuario.
* dar un orden visual en formas que interactúen correctamente y den como resultado algo estético y fácil de digerir, visualmente hablando.
* proporcionar una paleta de colores que comunique armonía, coherencia y que sea atractiva.
* trabajar en el tratamiento de imágenes.
* crear ilustraciones internas y externas.
* dar seguimiento en medios impresos y digitales.

Por ello, es importante que el diseñador tenga los conocimientos básicos sobre la profesión y que tenga un lenguaje visual evolucionado para dar una solución funcional al problema de comunicación visual.

Y es que el diseño gráfico viste aquellos códigos de programación, guía a la animación para que visualmente sea atractiva, comunica visualmente lo que los guiones o historias nos dictan y da funcionalidad para que el mensaje sea recibido.

En **3Dslug** casi todo lo relacionado con diseño gráfico se ha descargado de la web de assets de Unity de forma gratuita en forma de paquetes como puede ser el jugador, los enemigos, las escenas, armas, etc.

Aun así, no todo se ha utilizado tal y como venía descargado. Se han añadido luces, efectos o pequeños detalles según han ido siendo necesarios, como por ejemplo la animación de ataque del jugador, que no estaba prediseñada y ha tenido que diseñarse desde cero.

Además de esto, se han combinado diferentes assets como puede ser la granada y el efecto de electricidad que genera para dar más sensación de realismo al explotar.

## Impacto del diseño en los usuarios

Los diseños y modelos escogidos para un videojuego influyen directamente en la percepción del usuario del mismo. No es lo mismo un videojuego de guerra como call of duty que uno de plataformas como el Mario Bross. Esto significa que no todos los juegos están dirigidos necesariamente a todos los públicos, estableciendo límites de edad para los niños en juegos susceptibles para su edad o informando sobre posibles riesgos en personas con algún tipo de problema que pueda verse afectado por el videojuego, como puede ser:

* la epilepsia por exceso de cambios de luz
* problemas de corazón por sustos inesperados o tensión
* dolor de cabeza por exceso de tiempo de juego, etc.

**3Dslug** es un videojuego principalmente violento y con algunas situaciones de terror debido al ambiente tétrico generado en alguna escena, por lo que puede no ser apto para cierto público con enfermedades de corazón o personas que no superen una edad mínima establecida. Esta edad mínima se ha establecido en los 16 años basándose en las reglas de edad PEGI.

## Diagrama de clases

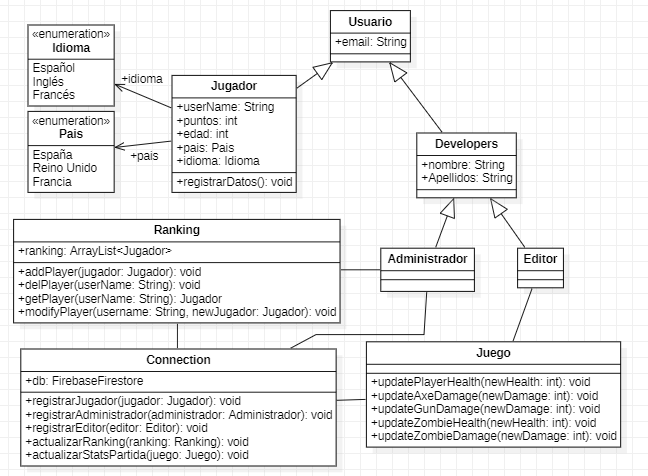
Un diagrama de clases en Lenguaje Unificado de Modelado (UML) es un tipo de estructura estática que describe la estructura de un sistema mostrando las clases de este, sus atributos, operaciones (o métodos) y las relaciones entre los objetos.

En cuanto a las clases, podemos definir las mismas como una plantilla para la creación de objetos de datos según un modelo predefinido. Cada clase es un modelo que define un conjunto de variables (estado) y métodos para operar con dichos datos (comportamiento). Cada objeto creado a partir de la clase se conoce como instancia de la clase.

En cuanto a las relaciones entre los objetos, son el tercer pilar fundamental del diagrama de clases. Pueden ser binarias o de orden superior. Si dos clases están relacionadas significa que esas clases tienen algo que ver entre sí.

Puesto que hablamos de un videojuego, las clases son asignadas a diferentes objects que ejercen las funciones designadas en la clase. Todas estas clases asignadas heredan de MonoBehavior y adquieren 2 métodos esenciales para el funcionamiento del videojuego: **Start y update.** Start se ejecuta solo en el primer fotograma en que se instancie el object mientras que update se ejecuta en cada uno de los fotogramas, pudiendo desarrollar en ese método, por ejemplo, el movimiento del jugador.

Además del juego, se propone de cara a una futura ampliación el diseño de una app también para móvil en la que podrán entrar los desarrolladores del juego bajo los roles de administración de jugadores o edición de las stats del juego como la salud del jugador o el daño producido con cada ataque, quedando como resultado el siguiente diagrama de clases:



## Persistencia de la información

Se denomina persistencia a la capacidad de guardar la información de un programa para poder volver a utilizarla en otro momento. Esto puede significar guardar los datos en un fichero (de texto, binario, csv) o guardar los datos en una base de datos (y sus distintas alternativas)

Para el desarrollo de 3Dslug, utilizaremos la base de datos NoSQL de Google Firebase y, además, un guardado en ficheros para almacenar los datos de la partida en el propio almacenamiento del teléfono.

**NoSQL** significa “not only SQL”, es decir, no sólo SQL. No es un modelo antagónico, si no un enriquecimiento y complemento útil de las tradicionales bases de datos SQL relacionales.

NoSQL plantea modelos de datos específicos de esquemas flexibles que se adaptan a los requisitos de las aplicaciones más modernas. Este tipo de bases de datos surgió debido a las limitaciones y problemas de las bases de datos relacionales, que no son capaces de hacer frente a las exigencias del desarrollo moderno. En cambio, las bases de datos NoSQL utilizan novedades, como los servidores en la nube y estructuras de datos muy potentes y flexibles.

El funcionamiento de estas bases de datos parte de la premisa de no usar tablas tradicionales y rígidas para almacenar los datos. En su lugar, organizan grandes volúmenes de datos con técnicas más flexibles como por ejemplo documentos y pares de clave valor. Una de las particularidades de los sistemas NoSQL es el escalamiento horizontal. Para entender este concepto es necesario saber que las bases de datos tradicionales escalan de manera vertical, es decir, toda su capacidad de rendimiento se basa en un solo servidor, por lo que para aumentar su capacidad hay que invertir en un servidor más potente (una opción muy cara en el largo plazo). El escalamiento horizontal supone que las soluciones NoSQL distribuyen sus datos en varios servidores, por lo que si necesitamos manejar más volumen de datos con un servidor austero sería suficiente. De esta manera, pueden almacenar grandes cantidades de datos a un precio menor.

### Decisión del tipo de almacenamiento de datos.

En primer lugar, se ha elegido Firebase por su facilidad para almacenar datos desde teléfonos inteligentes y su flexibilidad a la hora de crear colecciones. Por otro lado, se ha escogido utilizar almacenamiento en ficheros puesto que los datos de la partida guardada solo le interesan al propio usuario, de forma que no es necesario un uso de base de datos y se aprovecha una metodología más rápida y menos dependiente de una red a internet.

Para ello se analizan una serie de conceptos sobre la persistencia de datos:

* **Integridad:** Es la garantía de que los datos almacenados mantendrán su exactitud y consistencia en el tiempo. En NoSQL no es necesario definir el tipo de dato a almacenar, de forma que permite almacenar datos de una forma extremadamente flexible. Se da prioridad especialmente al acceso a los datos.
* **Escalabilidad:** La capacidad de crecimiento de la base de datos, especialmente ágil en las bases de datos NoSQL facilitando la expansión de la base de datos de una forma rápida y barata.
* **Velocidad:** Capacidad de escribir y leer en una base de datos en un periodo de tiempo. En NoSQL se suele contar con mecanismos de búsqueda sumamente rápida para conseguir un dato específico entre millones. La principal ventaja en la velocidad es que puedes diseñar la base de datos en función de las consultas esperadas.
* **Consistencia vs Redundancia:** En SQL, consiste en asegurarse de que un único dato esté una única vez en toda la base de datos. Mientras que en NoSQL, la redundancia es repetir adrede los datos a conveniencia en varias partes de la base de datos. De forma que se pueden almacenar los mismos datos de un usuario dentro de cualquier colección de forma que pueda pertenecer a distintos objetos, como por ejemplo al administrador.

En definitiva, la justificación de la elección de NoSQL se basa en lo siguiente:

* Necesidad de ver información en tiempo real para los cambios del ranking. La mejor opción es el cloud Storage de firebase.
* Puesto que el proyecto aún está en las primeras etapas de desarrollo, NoSQL facilitará la modificación de los datos a la hora de incluir al administrador de datos y editor de funcionalidad.
* No requerimos de operaciones atómicas que cambien muchas entidades, En principio el jugador hará un insert de sus datos con su puntuación al terminar la partida, mientras que el administrador solo podrá modificar datos de estos usuarios y el editor podrá modificar valores dentro del juego.
* Si el videojuego es bien recibido por el público, será necesario manejar un volumen de datos mayor que facilitará la escalabilidad horizontal de NoSQL.
* Con el objetivo de alcanzar la máxima velocidad posible de lectura y escritura para la actualización del ranking y los datos pertinentes, se aprovecharán los mecanismos que ofrece NoSQL.

Finalmente la estructura de la base de datos de 3Dslug queda de una forma similar a la siguiente:

