GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

CREACIÓN DE UN VIDEOJEUEGO INTERACTIVO PARA EDUCAR EN EL USO RESPONSABLE DE LAS REDES SOCIALES

Autor: Ana Leticia Urbistondo Murua

Co-Directores: Mario Castro Ponce, Gregorio López López

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Creación de un videojuego interactivo para educar en el uso responsable de las redes sociales

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico **2019/20** es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Ana Leticia Urbistondo Murua Fecha: 11/ 07/ 2020

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Mario Castro Ponce Fecha: 11/07/2020

Fdo.: Gregorio López López Fecha: 11/07/ 2020

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

CREACIÓN DE UN VIDEOJEUEGO INTERACTIVO PARA EDUCAR EN EL USO RESPONSABLE DE LAS REDES SOCIALES

Autor: Ana Leticia Urbistondo Murua

Co-Directores: Mario Castro Ponce, Gregorio López López

Madrid

CREACIÓN DE UN VIDEOJUEGO INTERACTIVO PARA EDUCAR EN EL USO RESPONSABLE DE LAS REDES SOCIALES

Autor: Urbistondo Murua, Ana Leticia
Director: Castro Ponce, Mario
Director: López López, Gregorio

RESUMEN DEL PROYECTO

Este trabajo de fin de grado consiste en el desarrollo de una aplicación móvil que tiene como objetivo educar y mostrar a los jóvenes usuarios de Internet los peligros y consecuencias que conllevan el *Online Grooming* junto con las técnicas manipulativas que utilizan estos depredadores sexuales en las redes sociales.

Palabras clave: Aplicación, *Online Grooming*, Pedofilia, Internet, Educación

1. Introducción

Debido a los peligros que acechan en Internet, junto con la cantidad de usuarios que caen víctimas de los engaños y manipulaciones de los depredadores sexuales online, con este trabajo se propone desarrollar un videojuego de temática "*juego serio*" [1] que eduque a los jugadores sobre las técnicas manipulativas que utilizan estos criminales en la red y, de esta manera, poder aplicar dichos conocimientos adquiridos gracias a esta aplicación en la vida real.

2. Definición del Proyecto

Este proyecto consiste en el diseño y desarrollo de una novela visual de aplicación móvil, compatible tanto para Android como para iOS, donde se cuenta la historia de un adolescente que utiliza por primera vez su propio dispositivo electrónico, en este caso un ordenador, y se adentra en el mundo de los videojuegos online. En el transcurro de una semana, el protagonista conoce a un usuario de Internet y se hace su amigo, confiando poco a poco en él. Es la responsabilidad del jugador tomar control de las decisiones de este protagonista y, por tanto, de las consecuencias de sus acciones. En el juego, estas decisiones conducen a diferentes finales según la ruta que ha tomado el jugador.

Dado que este trabajo tiene como objetivo educar a jóvenes jugadores sobre las técnicas manipulativas de los depredadores sexuales online, el diseño del contenido del mismo se ha realizado a partir de la información publicada en informes *Online Grooming*, como por ejemplo el informe de *Violencia Viral* [2] desarrollado por *Save the Children*. Asimismo, se ha extraído información del trabajo de fin de grado realizado por una estudiante de la Facultado de Ciencias Humanas y Sociales de la Universidad Pontificia Comillas Patricia Alonso titulado *Online Grooming* [3] que ha resultado a la vez informactivo e inspurador para el desarrollo de esta aplicación.

Finalmente, este proyecto utiliza una base de datos creada gracias a la aplicación de *Google Firebase* [4], donde se realizan escrituras de los datos relevantes de los jugadores (género, edad, nombre) junto con los finales obtenidos dentro de la aplicación y las decisiones tomadas a lo largo de la historia. Esta funcionalidad ofrece la oportunidad a investigadores sociales y de datos de realizar estudios cuantitativos en el futuro sobre los conocimientos que tienen los usuarios partícipes en esta aplicación en relación con *Online Grooming*, convirtiendo esta base de datos en una fuente de información vital para futuros proyectos investigativos.

En conclusión, los objetivos que se han cumplido con este proyecto son los siguientes:

1. Realizar un estudio sobre las amenazas de Internet: realizar un perfil de amenazas y depredadores que existen en Internet e implementarlas en el proyecto.
2. Programación utilizando Unity y diseño de un árbol de decisiones: diseñar la historia en la que se desarrollará el videojuego utilizando un árbol de decisiones.
3. Diseño de una base de datos: diseñar una base de datos donde se almacenarán los datos de los jugadores.

3. Descripción del sistema

Se ha utilizado como entorno de desarrollo y motor de videojuego la plataforma *Unity* [5], implementando a su vez el controlador de diálogo *YarnSpinner* [6] que utiliza guiones de lenguaje *Yarn*. Este motor de diálogo ayuda a los desarrolladores a implementar el árbol de decisiones de una historia de manera sencilla, utilizando ramas de diálogo y guardado de variables en su procedimiento. Esta aplicación utiliza diversas clases desarrolladas en *C#*, también denominados objetos en *Unity*, donde se especifican las funcionalidades principales de este proyecto y, además, establecen su estructura.

Las clases principales que componen esta aplicación son las siguientes: clase *DialogueRunner*, se encarga de diferenciar entre las líneas encontradas dentro del guion de diálogo y de clasificarlas según su estructura (líneas de narración, líneas de diálogo, rama de opciones o comandos); clase *ClassicDialogueUI*, controla todos los componentes dentro de una escena y responde a las instrucciones recibidas de *DialogueRunner;* y, finalmente, la clase *RopeworkManager* [7] define los comandos únicos elaborados por el desarrollador que se encarga de especificar funcionalidades adicionales que se desean implementar en el videojuego.

Esta aplicación, a su vez, interaccionará con otras clases, como por ejemplo *FireBaseConnection, Choices* y *Player*, para realizar la conexión con la base de datos y escribir los datos relevantes del jugador, junto con las decisiones tomadas por éste a lo largo de la historia. Además, dado que se han implementado animaciones encargadas de simular un desvanecimiento entre escenas, se ha establecido la clase *LevelChanger* para realizar la administración y control de dichas animaciones. Por último, se ha añadido la funcionalidad de guardar y cargar partida, funciones especificadas en la clase *Save*, facilitando de esta manera una jugabilidad más sencilla y eficiente para el usuario. En la *Figura 1* de esta sección se puede observar el diagrama de bloques donde se establece la relación de todos los objetos mencionados previamente, junto con sus funciones.

En el desarrollo del árbol de decisiones de la historia, se han utilizado las características principales encontradas dentro de los trabajos investigativos sobre *Online Grooming* previamente estudiados. Para poder diseñar un entorno realista que ayude en la educación del jugador sobre esta temática, se han realizado dos perfiles principales que establecen las características y propiedades fundamentales que debe cumplir e implementar esta historia: perfil de la víctima y el perfil del acosador.

En el perfil de las víctimas se establecen las siguientes propiedades: soledad, falta de comunicación de la familia, un uso elevado de Internet con un dispositivo electrónico propio, falta vigilancia adulta en el uso de Internet y baja autoestima del jugador. Siguiendo estas características, en esta historia el protagonista es un adolescente que, debido al trabajo de sus padres, está mudándose constantemente, por lo que no tiene un gran número de amigos y recurre a Internet para socializarse con otros usuarios.

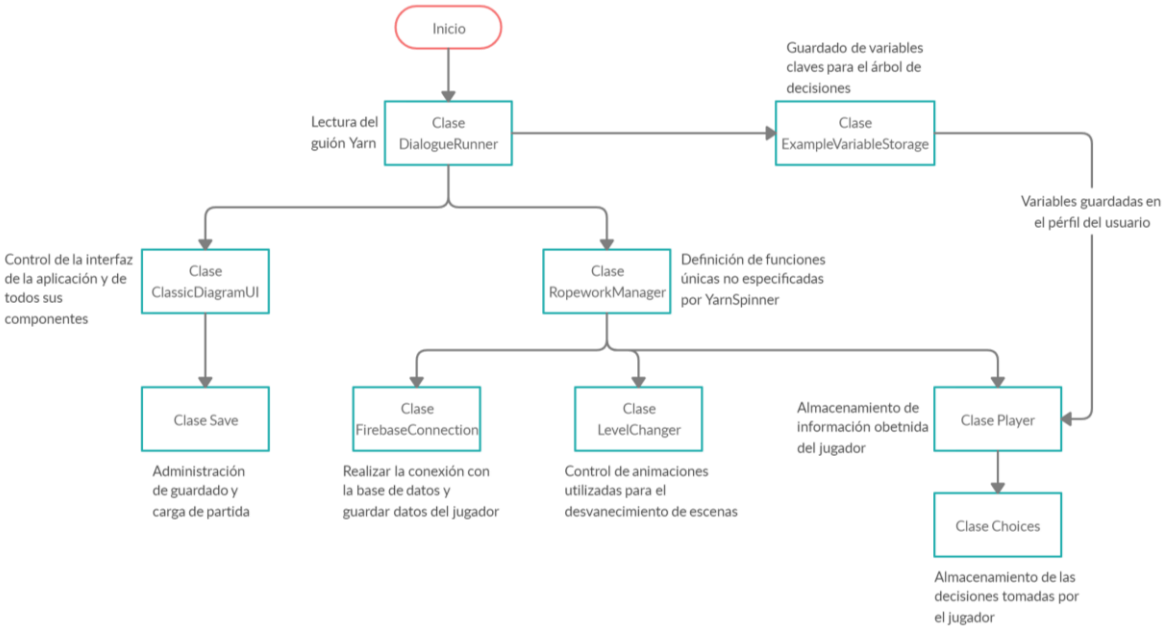

Figura 1: Diagrama de bloques del proyecto

Por otro lado, en el desarrollo del perfil del acosador y las técnicas manipulativas encontradas se han implementado las siguientes: introducir lentamente temas sexuales en las conversaciones, elogiar constantemente a otros usuarios, realizar preguntar personales sobre la situación familiar del jugador y, entre muchos otros, uso de chantajes emocionales al no recibir la respuesta deseada por parte de otros usuarios.

Todas estas características han ayudado al desarrollo de un entorno realista que muestra a los jugadores diversos aspectos que se pueden encontrar en el proceso manipulativo del *Online Grooming*, con el objetivo de educarles sobre esta temática y, a consecuencia de ello, ayudarles a identificar a depredadores sexuales en el mundo de Internet y protegerse de esta manera de sus engaños y manipulaciones.

4. Resultados

En la siguiente imagen (*Figura 2*), se muestra una captura obtenida en la ejecución de la aplicación móvil. Como se puede observar, se ha conseguido implementar los dibujos diseñados previamente en el desarrollo de este proyecto, tanto de personajes como los fondos que establecen el entorno de cada escena. Asimismo, se muestran las líneas de diálogo y narración establecidas en el guion, indicando a su vez el personaje que dice dicha línea.

Figura 2: Captura de pantalla de la aplicación resultante

5. Conclusiones

El mundo de los videojuegos ha sufrido un gran cambio en los últimos años. A medida que las tecnologías van evolucionando, surgen nuevos géneros en este mercado, además de dispositivos de tecnología avanzada que introducen una nueva herramienta a utilizar en esta industria. A pesar de que esta evolución es beneficiosa para la humanidad, ya sea en el mundo del entretenimiento o en nuestras necesidades de la vida cotidiana, junto con estos nuevos descubrimientos también surgen nuevas herramientas que los criminales en Internet pueden utilizar para su propio beneficio. Ante nuevas tecnologías, nuevos peligros acechan, ya sea hacia nosotros mismos o los más vulnerables de nuestra sociedad, los jóvenes.

Entre estas amenazas se encuentra el *Online Grooming*, una técnica manipulativa que utilizan los depredadores sexuales en Internet para ganarse la confianza y amistad de los más jóvenes en las redes sociales, con el objetivo de interaccionar de manera sexual con el menor, ya sea a través de imágenes, mensajes o incluso realizando un encuentro en la vida real. Con este proyecto se desea educar a los jóvenes vulnerables sobre esta gran amenaza de la red y, al mostrarles las técnicas manipulativas que estos usuarios utilizan, protegerse de sus engaños y utilizar Internet de manera segura.

6. Referencias

[1] J. F. C. Lobo, «Juegos Serios: Alternativa Innovadora,» II Congreso en líne en Conocimiento Libre y Educación CLED2011, 2014. [En línea]. Available: http://erevistas.saber.ula.ve/index.php/cled/article/viewFile/4862/4680.

[2] «Save the Children,» Análisis de la violencia contra la infancia y la adolescencia en el entorno digital, 2019 Julio. [En línea]. Available: https://www.savethechildren.es/publicaciones/informe-violencia-viral-y-online-contra-la-infancia-y-la-adolescencia.

[3] P. A. Gonzalez, «Online Grooming,» Abril 2019. [En línea]. Available: https://repositorio.comillas.edu/xmlui/bitstream/handle/11531/30848/TFG-%20Alonso%20Gonzalez%2C%20Patricia.pdf?sequence=1&isAllowed=y.

[4] «Firebase,» [En línea]. Available: https://firebase.google.com/.

[5] «Unity,» [En línea]. Available: https://unity.com/.

[6] Secret Lab y Yarn Spinner Contributors, «YarnSpinner,» 2020. [En línea]. Available: https://yarnspinner.dev/docs/tutorial.

[7] Y. R., «“Roperwork Visual Novel”,» GitHub, [En línea]. Available: https://github.com/radiatoryang/ropework.

CREATION OF AN INTERACTIVE VIDEOGAME WHICH EDUCATES ON THE RESPONSIBLE USE OF SOCIAL MEDIA

Author: Urbistondo Murua, Ana Leticia.
Supervisor: Castro Ponce, Mario.
Supervisor: López López, Gregorio.

ABSTRACT

This project consists of the development of a mobile application which aims to educate and show young Internet users about the dangers and consequences produced by *Online Grooming*, in addition to the manipulative techniques used by these sexual predators on social media.

Keywords: Application, *Online Grooming*, Pedophilia, Internet, Education

1. Introduction

Due to the dangers that lurk on the Internet, in addition to the number of users that fall victim to the deceits and manipulations caused by online sexual predators, the aim of this project is to develop a "*serious games*" [1] that educates players on the manipulative techniques used by these online criminals, thus enable the users to apply these recently acquired knowledge through this application in real life.

2. Project definition

This project consists of the design and development of a visual novel in a mobile application compatible for either Android and IOS, in which develops the story of a teenager who uses their own electronic device for the first time, in this case a computer, and enters the world of online videogames. Throughout the week, the protagonist meets another Internet user and they become friends, increasing their trust towards this person day by day. It is the responsibility of the player to take control of the protagonist's decisions and, consequently, atone to the consequences of their actions. In the game, these decisions lead to different endings depend on the route taken by the player.

Due to the fact that the aim of this project is to educate players on the manipulative techniques used by online sexual predators, a variety of investigative reports covering topics on *Online Grooming* have been used, such as a report called *Violencia Viral* developed by *Save the Children* [2]. Another project written by the student Patricia Alonso from the Faculty of Human and Social Sciences of the University Pontificia Comillas called *Online Grooming* [3] has also become a great source of information in the development of this application.

Finally, this project uses a database created through the *Google Firebase* [4] application, where relevant data related to the players (gender, age, name) is written, as well as the endings obtained within the application and the decisions carried out throughout the story. This functionality enables social and data researches to develop quantitative studies in the future on the knowledge obtained by the users who played this videogame regarding *Online Grooming*, turning this database into a vital source of information for future research projects.

In conclusion, the objectives achieved with this project are the following:

1. Research on the dangers of the Internet: create a profile on the dangers and sexual predator that exist on the Internet and implement them in this project.
2. Programming using Unity and design of a decision tree: develop the story in which the story of the videogame develops using a decision tree.
3. Database design: develop a database where all the data related to the players will be stored.

3. Description of the system

The *Unity* [5] platform has been used as a development environment and video game engine, in addition to *YarnSpinner* [6] as the dialogue controller, where scripts written in *Yarn* language have been used. This dialogue engine enables developers to implement decision trees in the story in a simple way, using throughout the procedure dialogue branches and variable storage. This application uses a variety of classes, also called objects in *Unity*, where the main functionalities of this project are established, as well as its structure.

The main classes that build this application are the following: *DialogueRunner*, in charge of classifying the lines found within the script according to their structure (narration lines, dialog lines, options branch or commands); *ClassicDialogueUI*, controls all components within a scene and answers to the instructions received from the *DilogueRunner*; and finally, the *RopeworkManager* class [7] defines the unique commands elaborated by the developer. These commands are responsible for specifying additional functionalities that want to be implemented on the videogame.

This application, in addition, will interact with other classes, such as *FireBaseConnection*, *Choices* and *Player*, in order to establish the connection with the database, storing the relevant data regarding the player. Additionally, due to the fact that animations in charge of simulating the fading between scenes have been implemented, the class *LevelChanger* have been established in order to administrate and control these animations. Lastly, the save and load game functionality has been added, throughout the class *Save*, enabling a simpler and more efficient gameplay for the user. In the *Figure 1* found in this section the block diagram used for this project is shown. This diagram establishes the relationship of all the previously mentioned objects and their functions.

Throughout the development on the decision tree of the story of this application, the main characteristics found within previously studied research studies regarding *Online Grooming* have been used. In order to design a realistic environment which helps in the education of the player regarding this theme, two different profiles have been developed. These profiles establish the main characteristics and properties that must be fulfilled and implemented in this story: the profile of the victim and the profile of the stalker.

The profile of the victim establishes the following properties: loneliness, lack of communication with the family, a high use of the Internet using their own electronic device, lack of adult surveillance in the use of the Internet and low self-esteem. Following these characteristics, in this story the protagonist is a teenager who, due to the work of his parents,

is constantly moving out. Therefore, they do not have a large number of friends and use the Internet in order to socialize with other users.

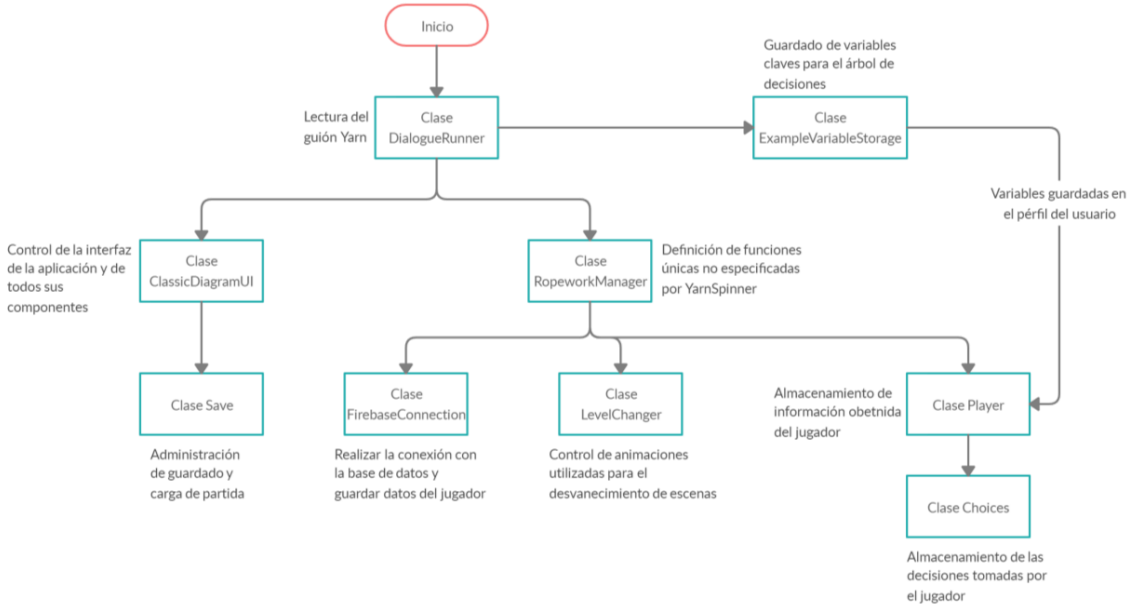

Figure 1: Block diagram of this project

On the other hand, regarding the development of the stalker's profile and the manipulative techniques used by these users, the following characteristics have been implemented: slowly introduce sexual topics in conversations, constantly praise other users, ask personal questions about the player's family situation and, among many others, use emotional blackmail when not achieving the desired answer from other users.

All these characteristics have assisted on the development of a realistic environment where different aspects are shown to the player. These aspects can be found in the manipulative process of *Online Grooming* and they are used in order to educated users regarding this topic and, as a consequence, enable them to identify predators in the Internet world and protect themselves from their deceptions and manipulations.

4. Results

The following image (*Figure 2*) shows a screenshot obtained during the execution of this mobile application. As it is shown in the image, it has been possible to implement the drawings previously designed throughout the development of this project, both for the characters and the backgrounds that establish the environment for each scene. Additionally, it is shown the dialogue and narration lines established in the script, indicating in the process which characters says that line.

Figure 2: Screenshot of the resulting application

5. Conclusions

The videogame world has greatly changed in recent years. As the evolution of technologies continues and develops, new genres emerge in this marked, in addition to advanced technological devices that introduce tools ready to be used in this industry. Despite the benefit for humanity that this evolution provides, whether it is in the entertainment world or in our daily life needs, along with these discovers new tools used by Internet criminals also emerge. With new technologies, new dangers emerge, either targeting ourselves or aiming towards the most vulnerable in our society: young people.

Among these threats *Online Grooming* appears, a manipulative technique used by sexual predators on the Internet in order to gain the trust and friendship of the youngest users found on social media, with the final objective to achieve sexual interaction with the minors, either through exchange of images, messages or even meeting in real live. This project aims to educate these vulnerable young people regarding this online threat and, by showing theme the manipulative techniques used by these users, learning to protect themselves from these deceptions and use the Internet safely.

6. References

[1] J. F. C. Lobo, «Juegos Serios: Alternativa Innovadora,» 2014. Available: http://erevistas.saber.ula.ve/index.php/cled/article/viewFile/4862/4680.

[2] «Save the Children,» Análisis de la violencia contra la infancia y la adolescencia en el entorno digital, 2019 Julio. Available: https://www.savethechildren.es/publicaciones/informe-violencia-viral-y-online-contra-la-infancia-y-la-adolescencia.

[3] P. A. Gonzalez, «Online Grooming,» Abril 2019. Available: https://repositorio.comillas.edu/xmlui/bitstream/handle/11531/30848/TFG-%20Alonso%20Gonzalez%2C%20Patricia.pdf?sequence=1&isAllowed=y.

[4] «Firebase,» [En línea]. Available: https://firebase.google.com/.

[5] «Unity,». Available: https://unity.com/.

[6] Secret Lab, «YarnSpinner,» 2020. Available: https://yarnspinner.dev/docs/tutorial.

[7] Y. R., «“Roperwork Visual Novel”,» GitHub. Available: https://github.com/radiatoryang/ropework.

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

1.1 INTRODUCCIÓN

Actualmente, la tecnología se ha convertido en una de las herramientas más importantes de nuestra sociedad, por no decir imprescindible, y ha evolucionado en los últimos años a una velocidad sorprendente. La transición que han sufrido los dispositivos móviles a lo largo de esta generación es un gran ejemplo donde se muestra el gran impacto que tiene el descubrimiento de nuevas tecnologías en nuestra sociedad. La creación de Internet, sin embargo, es el evento histórico en el mundo digital que más ha marcado a la humanidad.

La primera conexión de ordenadores, convirtiéndose en el primer paso al conjunto de redes interconectadas que conocemos hoy en día, se estableció en 1969 bajo el nombre de Arpanet [1]. Existe la falsa creencia que Internet se creó con fines militares, con el fin de establecer conexiones seguras y de larga distancia durante la guerra fría. Sin embargo, esta conexión se llevó a cabo gracias a la colaboración de tres universidades en California, financiada años después por la agencia del Departamento de Defensa del gobierno de los Estados Unidos (DARPA), y tenía como objetivo obtener una herramienta de comunicación extensa y segura. La finalidad de este proyecto era conseguir una comunicación amplía, libre y disponible para todo el mundo, características que definen el Internet que conocemos hoy en día.

A pesar de que todos los objetivos iniciales planteados por estos desarrolladores han sido cumplidos, incluso sobrepasados dado que se ha llegado a obtener una escala mundial, no sólo son ventajas lo que Internet ha proporcionado. La Word-Wide-Web es más que una fuente de información y acceso a múltiples plataformas; también ha dado paso a nuevas amenazas que crecen cada año y que, además, perjudican a las víctimas en diversos aspectos. Por una parte, nuestra dependencia hacia la red incrementa diariamente, confiando nuestra información personal a páginas web que, sin nuestro conocimiento, utilizan dichos datos para sus propios beneficios o son incluso robados por terceras personas. Por ese motivo,

dicha dependencia hacia Internet, a pesar de ser beneficiosa, nos hace vulnerables hacia múltiples amenazas que se aprovechan de los propios beneficios que Internet aporta. Entre dichos peligros se pueden encontrar: robo de información, robo de identidad, violación de la privacidad, malware en los dispositivos, etc. Afortunadamente, en los últimos años la conciencia hacia estas amenazas ha incrementado considerablemente y, como consecuencia, se han llevado a cabo múltiples proyectos que tienen como iniciativa informar y mostrar a los usuarios diversos métodos de protección contra estos riesgos. Gracias a esta colaboración contra los peligros de Internet, el conocimiento de la sociedad sobre la *ciberseguridad* ha aumentado y los usuarios que han caído víctimas de estas amenazas han disminuido. A pesar de estas precauciones y medidas informativas, se puede encontrar un grupo dentro de nuestra sociedad severamente vulnerable a las amenazas que acechan en la red: los niños y adolescentes.

Debido a los largos periodos de tiempo que los jóvenes utilizan Internet, además de la facilidad con la que la juventud confía en extraños o aplicaciones poco fiables, se convierten en un objetivo perfecto para caer en los engaños que abundan en la red. A pesar del esfuerzo de la sociedad y las instituciones educativas por educar los riesgos asociados a esta tecnología, se necesita la colaboración e iniciativa de los jóvenes para poder protegerles de estos riesgos y alejarles del perfil vulnerable del que son asociados.

Por este motivo, con este proyecto se desea colaborar en la enseñanza de los riesgos de Internet a los preadolescentes pertenecientes al grupo vulnerable de usuarios (jóvenes de edades entorno a los 10-14 años), utilizando un método interactivo con el fin de aconsejar diversos procesos de protección. Dado que es de vital importancia captar el interés y atención de estos jóvenes, se plantea utilizar las mismas herramientas digitales de las que son tan dependientes en su propio aprendizaje. En otras palabras, con este proyecto de trabajo de fin de grado se propone desarrollar un videojuego de aplicación móvil en el que, de manera interactiva, se informe, exponga y, sobre todo, se eduque acerca los diversos riesgos que abundan en la red. En la actualidad existe un género de videojuegos que definen este estilo de aplicaciones y son denominados "*Serious games*", o juegos serios en español.

1.2 *"SERIOUS GAMES"*

Los "*Serious Games*", también denominados "*juegos serios*" en español, son videojuegos con un fin pedagógico, donde sus historias están basadas en escenarios reales y tienen como objetivo educar a los jugadores, sumergiéndoles en el proceso en un contexto específico para conseguir enseñar y entrenar a los usuarios de manera efectiva y exitosa. Dicho escenario les ayuda a mejorar tanto las conductas de los jugadores como sus actitudes, éstas siendo específicas a cada historia y videojuego. Todas ellas, sin embargo, comparten una única característica y es que son necesarias para el eficiente desempeño de una actividad particular en la actualidad [2].

Como se ha mencionado previamente, los "*juegos serios*" pueden alcanzar diversos campos educativos, desde una simulación de batalla utilizado como entrenamiento militar por el ejército, hasta empleado como un método terapéutico para pacientes de diversas enfermedades. En otras palabras, este género abarca un gran abanico de temas y campos, todos ellos con el objetivo de educar y fortalecer a los jugadores, construyendo un contexto realista en un mundo ficticio y digital.

En conclusión, siguiendo la definición de este grupo de videojuegos, se debe considerar este proyecto como un "*juego serio*", dado que tiene como finalidad la de educar a la juventud sobre diversas medidas preventivas y protectoras contra los riesgos de Internet. Para cumplir con este objetivo y recrear de esta manera un entorno realista dentro del mundo ficticio de la aplicación, se han estudiado diversos trabajos investigativos que analizan las amenazas que se encuentran en la red y abarcan diversas medidas de protección contra ellas.

1.3 *PELIGROS DE INTERNET: "ONLINE GROOMING"*

Los peligros de Internet son diversos y los daños causados a los usuarios abarcan múltiples campos, estos pudiendo ser el económico, el social o incluso abarcar problemas relacionados con la salud. Actualmente, y como se ha mencionado previamente en la introducción, los centros educativos han comenzado a tomar diversas iniciativas para enseñar a los jóvenes

sobre estos peligros y disminuir de esta manera el número de usuarios que caen víctimas de estas amenazas, protegiendo a los más vulnerables en el proceso.

Estas medidas educativas suelen abarcar temáticas relacionadas con la *ciberseguridad* como, por ejemplo, sobre la protección de los datos personales, aconsejando a los jóvenes de no dar información privada a ningún otro usuario o mencionarlo en cualquier red social, ya sean datos personales del joven, datos económicos (como cuentas bancarias o tarjetas) como información que puede ayudar a los criminales que acechan en las redes sociales en la elaboración de un crimen.

Otros temas tratados en dichos proyectos educativos aconsejan medidas protectoras contra los virus y "*hackers*" de Internet, recordando a los usuarios de no entrar en páginas sospechosas ni caer en la tentación de anuncios fraudulentos. Estos peligros no sólo tienen la capacidad de dañar los dispositivos electrónicos del usuario, sino que además pueden extraer información privada y utilizarla para robar cuentas y/o realizar diversos chantajes hacia el dueño.

A pesar de que todos estos métodos sean de vital importancia y hayan demostrado ser eficaces en la enseñanza a los jóvenes, el proyecto presentado en esta memoria no comparte la misma temática. Esto se debe a que se desea abarcar con este trabajo una amenaza distinta encontrada en la red; una amenaza que afecta gravemente a una gran cantidad de jóvenes diariamente y, a pesar de todas las medidas preventivas que se han intentado realizar, no disminuye en sus números. Este gran peligro de Internet es *Online Grooming.*

Online grooming hace referencia al embaucamiento realizado por un adulto utilizando Internet como herramienta principal, manipulando y engañando a jóvenes con fines sexuales [3]. En otras palabras, acosadores sexuales utilizan el anonimato de Internet para obtener la confianza de menores de mente vulnerable y, de esta manera, manipularles para hacerles partícipes de actos sexuales, ya sean intercambiando mensajes o fotos sexuales, como engañarles para realizar un encuentro en la vida real. A pesar de que *Online Grooming* sea un tema serio y extremadamente dañino para los jóvenes, a diferencia de las amenazas mencionadas anteriores, no se han podido encontrar proyectos actuales que abarquen este

tema, por lo que se ha decidido enfocar la temática principal de este proyecto en los métodos manipulativos que utilizan los depredadores sexuales online, diseñando de esta manera un "*juego serio*" donde se muestra a los jugadores diversas señales que deberán detectar en la vida real para protegerse del engaño elaborado por estos usuarios.

1.4 MOTIVACIÓN DEL PROYECTO

Como se ha mencionado en la introducción, es de vital importancia educar a los jóvenes pertenecientes al grupo de usuarios vulnerables de Internet sobre los peligros que *Online Grooming* presenta, protegiéndoles de esta manera de las consecuencias que conllevarían un uso irresponsable de esta tecnología.

Actualmente, existen múltiples cursos donde los participantes aprenden distintos métodos de protección informáticos y una utilización segura de Internet. Incluso los propios colegios se están movilizando para realizar charlas y actividades sobre la *ciberseguridad*, implementando también asignaturas sobre informática y seguridad en su programa de estudio. A pesar de estos métodos educativos hacia la juventud, si éstos no muestran interés o no prestan atención en las actividades, ignorando los consejos de los profesionales, entran dentro de este grupo vulnerable de usuarios de Internet y se convierten en potenciales víctimas de acoso y abuso.

Por este motivo, se ha considerado crear un videojuego para poder facilitar esta información a los jóvenes y poder mostrarles de forma interactiva las posibles consecuencias que podría llevar un mal uso de Internet. Muchos preadolescentes de edades de 10 a 14 años utilizan los videojuegos como entretenimiento y ocio, teniendo un especial éxito los juegos móviles, utilizando los dispositivos de sus padres o los suyos propios. Debido a este interés hacia las aplicaciones móviles como modo de entretenimiento, se ha escogido este método de enseñanza para este proyecto. Desarrollando un videojuego llamativo e interesante, que capte la atención de estos jóvenes usuarios, podrán aprender a protegerse en Internet de manera entretenida siguiendo una historia ficticia. Una ventaja de los juegos interactivos basados en una historia es que permite involucrar al jugador en la toma de decisiones (el

curso de la historia) y a empatizar con las situaciones que propone el juego en distintas etapas.

Dado que se quiere mostrar las posibles consecuencias que conlleva el mal uso de Internet y caer en los embaucamientos manipulativos de usuarios desconocidos, se ha planteado diseñar un juego de estilo novela visual con un árbol de decisiones donde se pueden obtener distintos finales. Se desea simular la vida cotidiana del jugador navegando por Internet y, según las decisiones tomadas como respuesta a los métodos manipulativos realizados por los acosadores, obtener distintos finales.

Este proyecto no sólo tiene como objetivo mostrar y enseñar a los jóvenes sobre los peligros en la red, sino que también tiene como iniciativa obtener la información del jugador (edad, género, etc.) junto con las decisiones tomadas en la historia para permitir en un futuro analizar el comportamiento de los jugadores, así como su aprendizaje.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

En este proyecto, dado que se ha trabajado en el diseño tanto artístico como digital del videojuego, se han utilizado diversos programas con funcionalidades distintas, con el fin de obtener el mejor resultado posible para esta aplicación.

2.1 UNITY

Unity es una plataforma de desarrollo en tiempo real para juegos 2D y 3D, que permite a múltiples artistas, diseñadores y desarrolladores poder trabajar de manera unívoca con el fin de crear una gran variedad de videojuegos para cualquier plataforma, desde MAC y PC, hasta IOS y Android [4].

Unity está diseñado para poder crear videojuegos tanto 2D como 3D, dándole la oportunidad de escoger a los desarrolladores al inicio de cada proyecto y de realizar las modificaciones necesarias en los ajustes de *Unity* para obtener la mejor calidad posible tanto en las imágenes de la aplicación, como en sus texturas o también *sprites*. En la página web de *Unity*, además de poder encontrar diversos tutoriales sobre la aplicación, *Unity* también proporciona un manual donde se muestra todo el proceso de descarga de la aplicación además de sus bases fundamentales para poder diseñar un videojuego [5]. En este proyecto, dado que se desea diseñar una novela visual, se ha trabajada en modo 2D y se ha utilizado gráficas planas, también denominados *sprites*, con una cámara sin perspectiva.

Para esta aplicación, se ha trabajado con la última versión de *Unity*, *Unity* 2019.4.0f1 y, a pesar de tener conocimientos sobre diversos lenguajes de programación, se han estudiado múltiples tutoriales online encontrados en la página web de *Unity* [6], para obtener de esta manera un funcionamiento óptimo del videojuego y evitar futuros fallos a la hora de validar sus resultados.

El interfaz de *Unity* está dividido en diversas zonas, cada una de ellas altamente importantes para realizar un uso eficiente y correcto de la aplicación. Este interfaz está dividido en 4 zonas: la ventana del proyecto, la ventana de jerarquía, la vista de escena y, por último, la ventana del Inspector.

2.1.1 VENTANA DEL PROYECTO

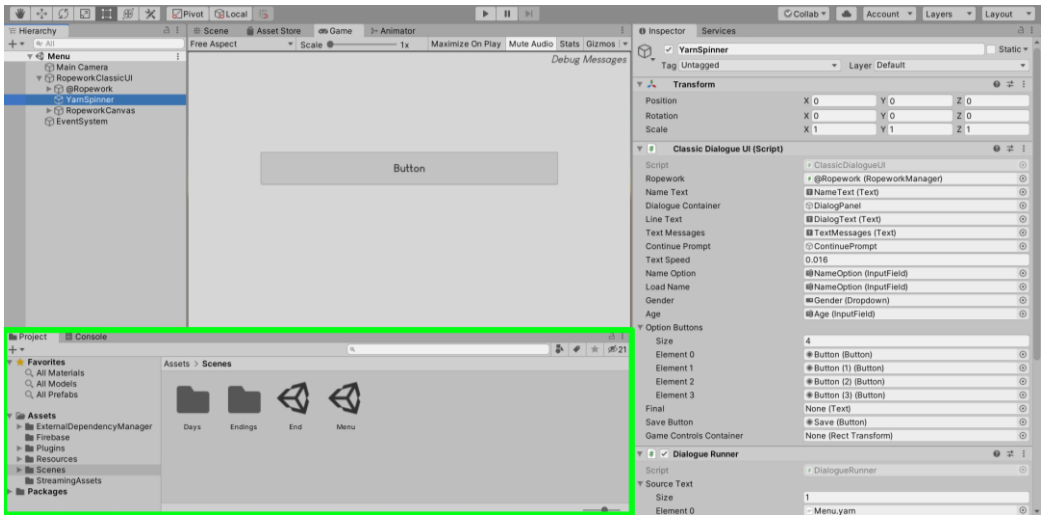

Figura 1. Unity - Ventana del Proyecto

En la ventana del proyecto se puede observar todos los componentes, denominados *Assets* en inglés, de la aplicación. Esta ventana es una vista previa de las carpetas y archivos del proyecto donde el desarrollador puede acceder y gestionar según vea necesario.

En la zona izquierda de la ventana se puede observar la estructura de carpetas del proyecto, siguiendo una jerarquía específica establecida por el desarrollador. En el panel de la derecha se observan los componentes almacenados en la carpeta seleccionada, pudiendo de esta manera ser modificados por el usuario en cualquier momento del desarrollo del videojuego. Además, se puede observar un buscador donde el jugador puede realizar la búsqueda de un componente específico que se encuentra dentro de las librerías o, incluso, comprobar si el componente seleccionado se encuentra en la escena actual del videojuego.

En *Unity* las escenas son componentes del videojuego donde se almacenan todos los objetos y clases necesarios para su funcionamiento, al igual que los ajustes de su cámara y las

animaciones usadas en la aplicación. En la *Figura 1* que se ha mostrado anteriormente, se puede observar que la escena *Menu* se encuentra activa y en la ventana de jerarquía se pueden observar todos sus objetos y componentes.

En este proyecto, la historia se desarrolla a lo largo de 5 días, utilizando de esta manera una escena para cada uno de ellos, además de añadir dos escenas adicionales: uno para el menú de inicio y otro para mostrar el final obtenido por el jugador. Los componentes y las escenas del proyecto se han separado en distintos directorios, los primeros (imágenes, escenas, código, etc.) se encuentran en la carpeta denominada "*Resources*" (Recursos en español) y los últimos en la carpeta llamada "*Scenes*" (Escenas en español), para obtener de esta manera una estructura clara y sencilla de todos los componentes utilizados en el proyecto y facilitando de esta manera su gestión y edición.

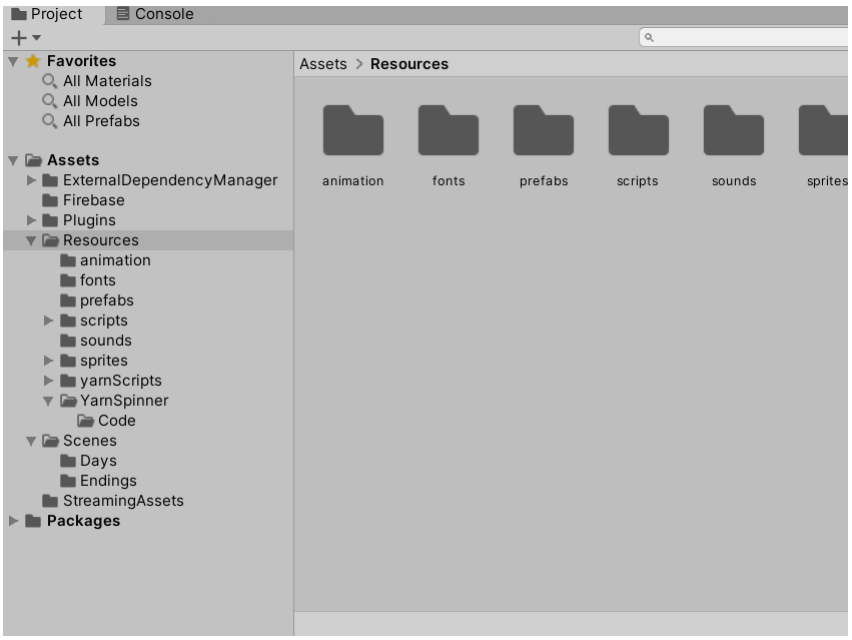

Figura 2. Proyecto final – Ventana del proyecto

2.1.2 VENTANA DE JERARQUÍA

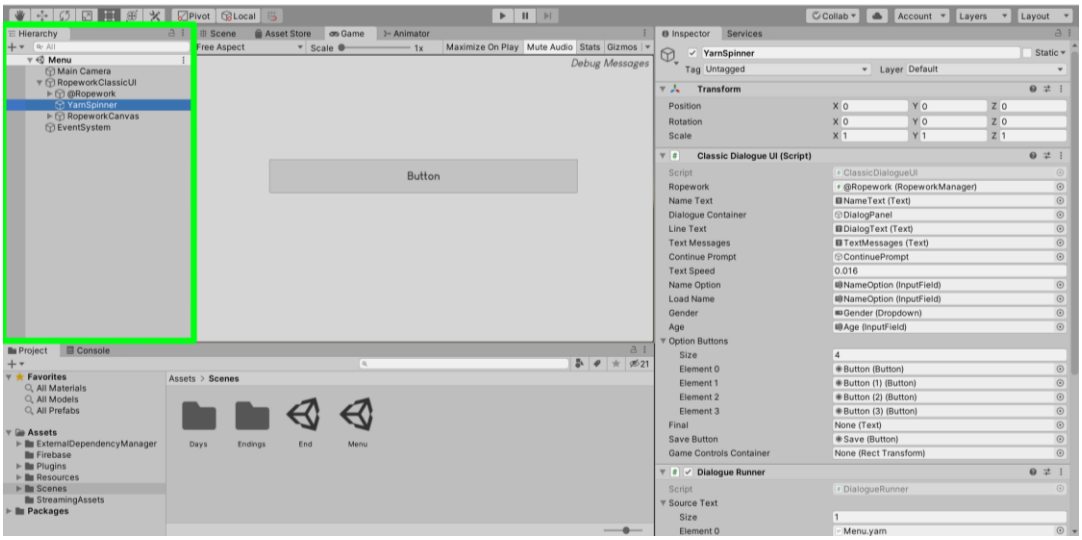

Figura 3. Unity – Ventana de Jerarquía

La ventana de jerarquía es una representación jerárquica de cada objeto (denominados *GameObject* en inglés) en la escena activa. Dicha ventana muestra la estructura que sigue los objetos que componen una escena, cómo están agrupados y la relación de parentesco (*Parenting*) que siguen. El concepto de *Parenting* en *Unity* se refiere a los componentes y funcionalidades que un "*Padre*" hereda a su "*Hijo*", permitiendo de esta manera un uso más eficiente y sencillo del código utilizando herencias.

En este proyecto, se ha utilizado una jerarquía de objetos perteneciente al código obtenido de un usuario de *GitHub* denominado *Ropework*, respetando la herencia de objetos obtenida de la estructura *YarnSpinner* que utiliza este videojuego. Ambos componentes se explicarán más detalladamente en apartados posteriores de esta memoria.

En dicha ventana, se pueden observar dos secciones fundamentales que se compone cada escena del juego: "*OptionsPanel*" y "*DialogPanel*". En el primero se encuentran todos los botones de decisión que se implementan en la historia, dando la posibilidad de colocar hasta 4 botones totales. En el caso de la escena menú, se incluirán en este panel las opciones utilizadas por el jugador para introducir su información. El panel denominado "*OptionsPanel*" crea una zona dentro de la escena donde se pueden colocar estas opciones,

permitiendo de esta manera poder colocar los objetos de elección del usuario respetando el tamaño de ventana del dispositivo. En otras palabras, el panel controla las posiciones y tamaños de los objetos que se encuentran dentro de él, sus hijos, pudiendo respetar la estructura del juego. Lo mismo ocurre dentro del panel denominado "*DialogPanel*", sólo que éste controla el diálogo de texto de la aplicación, además de los botones de guardado y carga del juego, junto con el panel que contiene el nombre del personaje.

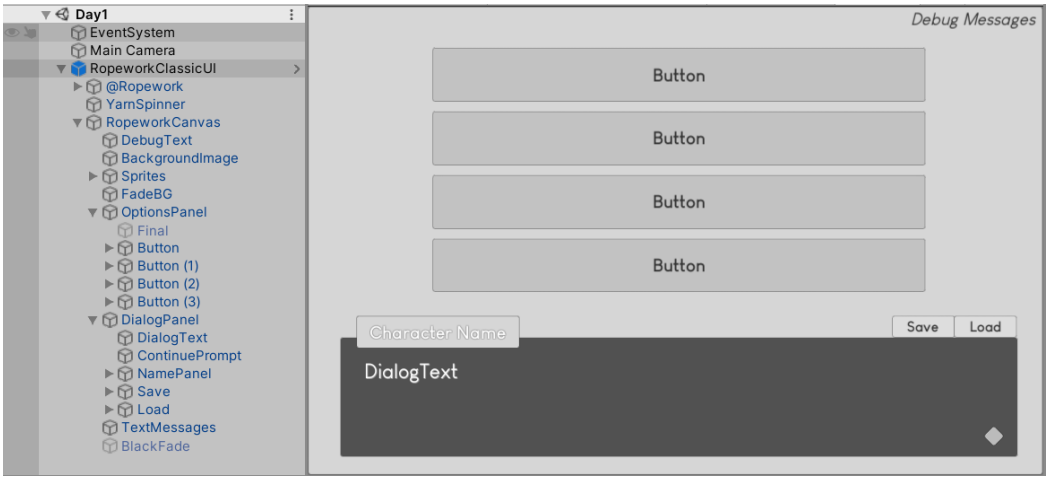

Figura 4. Proyecto final - Ventana de jerarquía

2.1.3 VISTA DE ESCENA

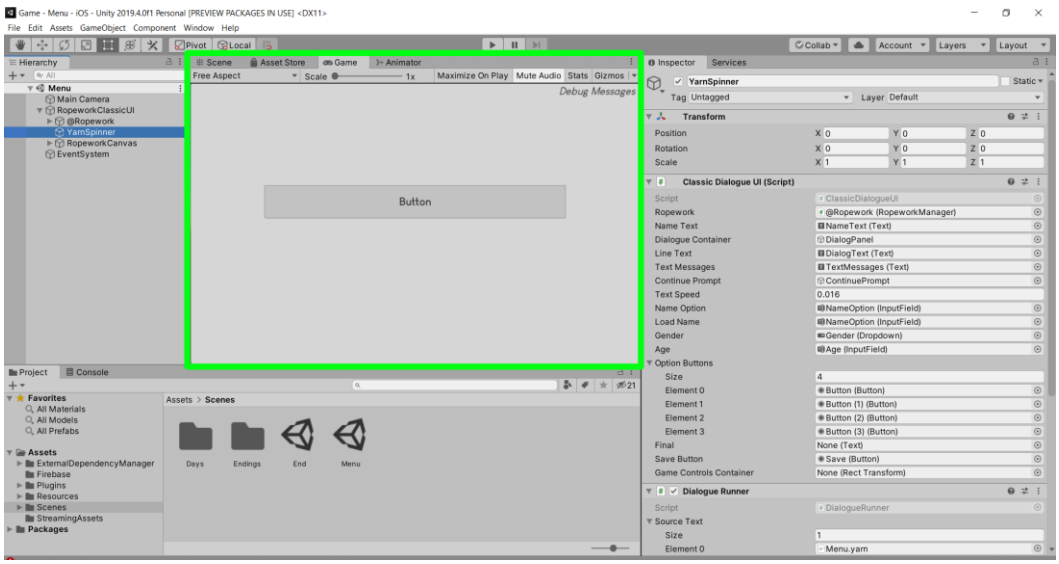

Figura 5. Unity - Vista de escena

La vista de escena es un componente de *Unity* que permite al desarrollador una navegación visual del proyecto desarrollado y poder editar su escena para realizar las correcciones necesarias que se observen en la ventana.

En esta zona existen dos subapartados: la vista de escena y la vista del juego. La primera, como se ha mencionado antes, permite al desarrollador poder editar el videojuego de manera visual, modificando a su vez sus componentes y cámaras. La vista del juego, sin embargo, muestra el resultado final que se observaría al ejecutar el videojuego, permitiendo de esta manera al desarrollador poder ver sus resultados en esta ventana y realizar las correcciones necesarias en la vista de escena.

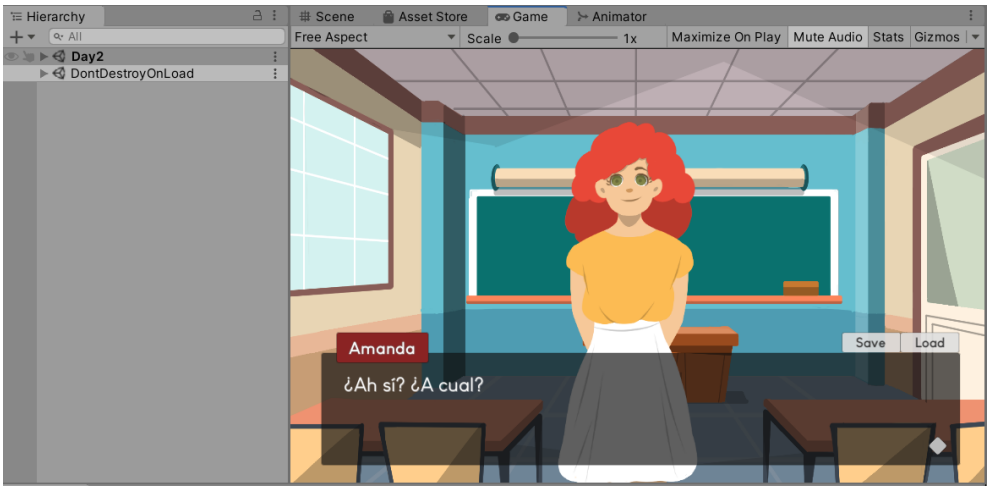

Figura 6. Proyecto final - Vista del juego

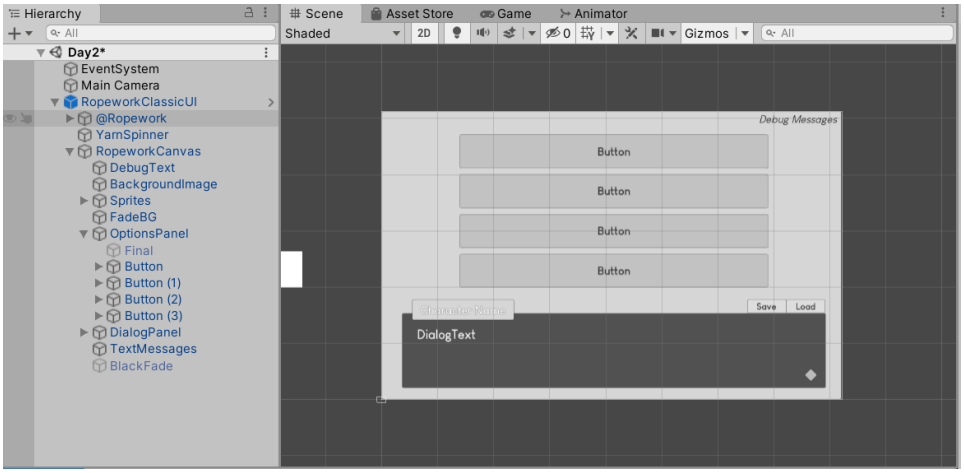

Figura 7. Proyecto final - Vista de escena

2.1.4 VENTANA DEL INSPECTOR

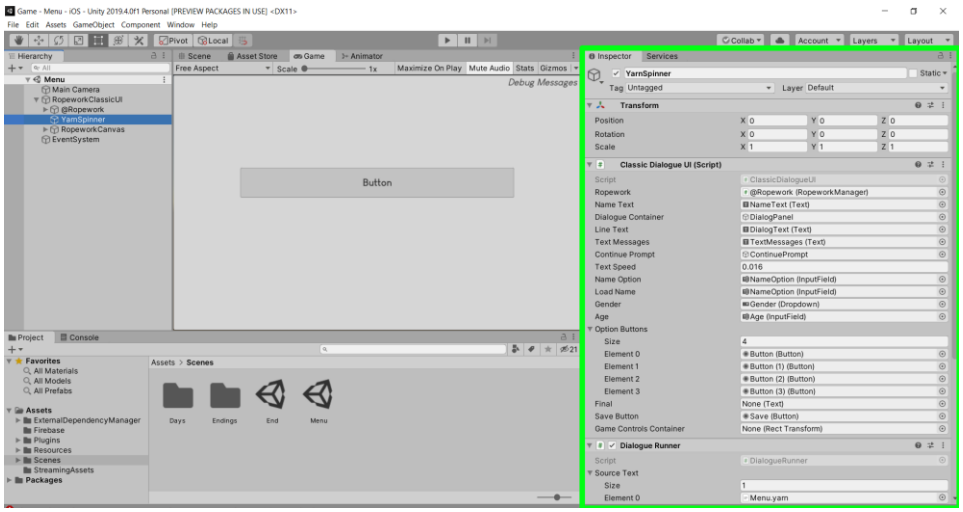

Figura 8. Ventana del Inspector

La ventana del Inspector permite al desarrollador visualizar y editar todas las propiedades del objeto de escena. El contenido de la ventana del Inspector varía, ya que todos los objetos encontrados en escena contienen diferentes propiedades y componentes, por lo que el *layout* (diseño) no será el mismo para todos los objetos.

Existen objetos provenientes del propio *Unity* como, por ejemplo, los botones o paneles, donde simplemente se pueden editar el formato de texto y la visualización del botón. Sin embargo, el propio desarrollador puede crear sus propios objetos, seleccionado la opción de *Unity* de "*create Empty*", donde se crea un objeto vacío. El desarrollador debe proporcionar a la aplicación un código en C#, un archivo ".cs", donde se especifica las funcionalidades y componentes del nuevo objeto creado. Una vez creado dicho archivo, en la ventana del Inspector se muestra la opción "*Add Component*" (Añadir componente en español), donde se le debe asociar el archivo de código diseñado al objeto de *Unity* vacío. De esta manera, los desarrolladores de videojuegos pueden crear sus propios objetos con funcionalidades específicas y únicas, obteniendo los resultados deseados y necesarios para el diseño de su videojuego.

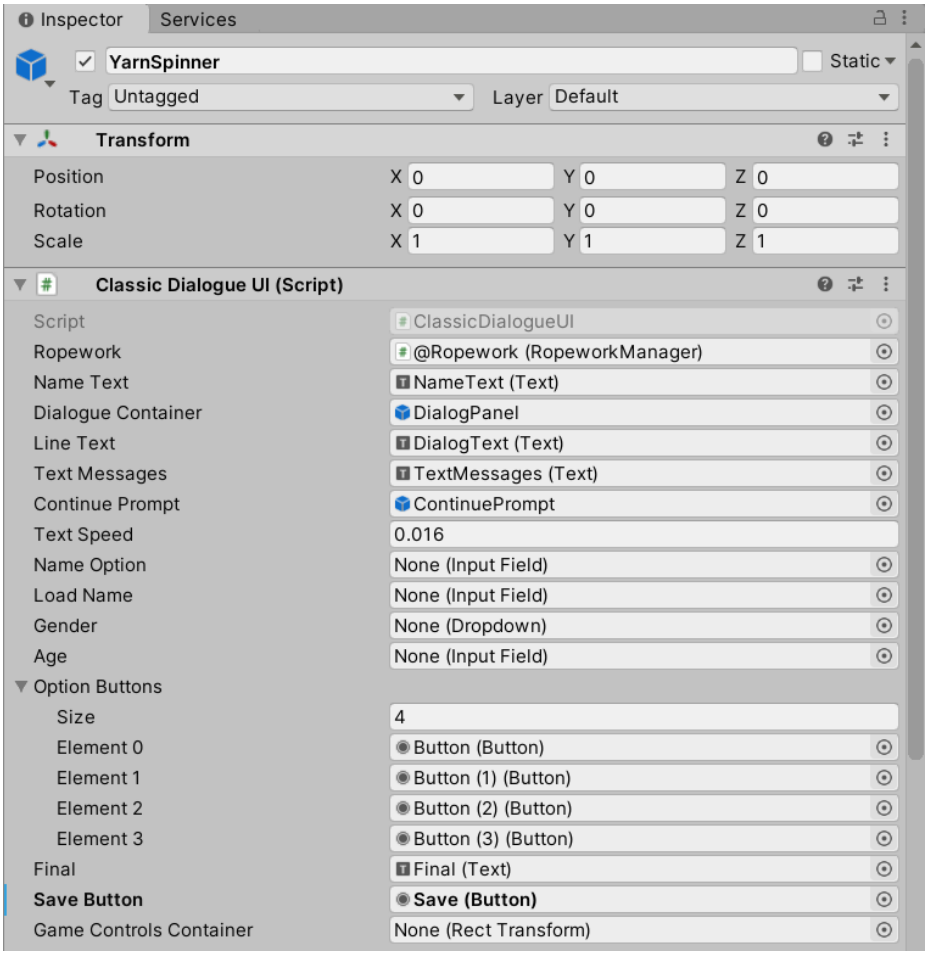

Figura 9. Clase única YarnSpinner

Esta aplicación ha utilizado la clase *YarnSpinner*, donde se utiliza tanto los componentes "*Classic Dialogue UI*", "*Dialogue Runner*" y "*Variable Storage*", que controlan la interfaz del juego, mostrando los mensajes de diálogo de los personajes, sus imágenes y los botones de opción que el jugador debe seleccionar. Además, se explicará más adelante la funcionalidad de los otros dos componentes, donde se controla la lectura del guion de la historia y se almacenan las variables necesarias para la creación del árbol de decisiones de la historia.

2.2 *YARNSPINNER*

En este proyecto se ha utilizado "*YarnSpinner*" [7], un implementador del lenguaje *Yarn* que permite diseñar de manera sencilla diálogos de texto y tomas de decisiones, facilitando el diseño del árbol de decisiones que se desea implementar en este proyecto.

Yarn es un formato de texto simple, diseñado para poder crear diálogos de texto y ramas de conversaciones dependientes de las decisiones tomadas anteriormente por el jugador. *YarnSpinner* tiene como funcionalidad analizar y ejecutar los archivos de texto que contienen el lenguaje *Yarn*, guardados en formato "*archivo.yarn.txt*", obteniendo de esta manera un diálogo donde las decisiones del jugador tienen sus variaciones y consecuencias en la historia.

YarnSpinner no solo permite guardar variables, crear ramas de decisiones y expresiones condicionales de tipo "*if*" para modificar los diálogos, sino que además permite al desarrollador crear sus propias funciones, pudiendo implementarlas dentro de su archivo de tipo *Yarn* y realizar diferentes ejecuciones dependiendo del estado en el que se encuentre en la historia.

A continuación, se mostrará un ejemplo de un archivo con lenguaje *Yarn* y se demostrará el resultado obtenido dentro del juego, mostrando la rama de decisiones y la implementación de funciones dentro del archivo de texto:

```
Mamá: ¿Estás jugando a estas horas?
Mamá: [name], me alegro de que te guste el regalo, pero ya es tarde y mañana
tienes colegio.
Mamá: Anda, apágalo y vete a dormir.

User: Sí, mamá.

<<Hide @ Mamá>>

-> Seguir jugando
      <<set $jugarTarde to true>>
      Cuando mi madre cerró la puerta, me puse el pijama y me tumbé en la cama
con el portátil
      Para simular que estaba durmiendo, apagué las luces y seguí jugando a
oscuras con los auriculares
```

```
      No fue hasta la madrugada cuando me empezaron a doler los ojos y me fui a
dormir.
-> Ir a dormir
      <<set $jugarTarde to false>>
      Mi madre tenía razón, mañana iba a estar muy cansada
      Guardé la partida y cerré el ordenador.
      Afortunadamente, terminé mis deberes a lo largo del fin de semana, por lo
que no me tenía que preocupar por las entregas del día siguiente.
      Me puse el pijama y me fui a dormir
```

En este código se puede observar diversas implementaciones del lenguaje *Yarn* en el proyecto. Lo primero de todo, se debe diferenciar las líneas de diálogo, las líneas de narración, las tomas de decisiones y las funciones.

Las líneas de diálogo se refieren a las frases que un actor o personaje dice dentro de la conversación. Estas líneas se pueden diferencias gracias a la implementación del símbolo ":". Cuando el formato de la línea es el siguiente "*Mamá: ¿estás jugando a estas horas?"* se puede observar que el personaje con identificador "*Mamá*" está diciendo la frase "*¿estás jugando a estas horas?"*". Cuando el código obtiene esta frase, *YarnSpinner* se encargará de mostrar el identificador del personaje en el panel del nombre, mostrando en este caso el nombre "*Mamá*", y se encargará de visualizar dentro del panel de diálogo la frase "*¿estás jugando a estas horas?"*. Además de exponer el nombre del personaje y su diálogo dentro de la escena, *YarnSpinner* también se encargará de resaltar la imagen del personaje, pudiendo diferenciar de esta manera quién está hablando dentro del diálogo.

Figura 10. Ejemplo de línea de diálogo

Las líneas de narración no pertenecen a ningún personaje, son simplemente líneas de texto que se muestran dentro del archivo. Como las líneas de narración son descriptivas para el jugador, no se resalta ninguna imagen del personaje y sólo se muestra la línea de texto. En el ejemplo mostrado en el código anterior, la frase "*Mi madre tenía razón, mañana iba a estar muy cansad@*" pertenece a este tipo de líneas y se puede observar su resultado en la *Figura 11:*

Figura 11. Ejemplo de línea de narración

Otra implementación que tiene el lenguaje *Yarn* es la rama de opciones. Para poder representar una opción dentro de la historia e indicar a *YarnSpiner* que debe de mostrar los botones de decisiones al usuario, *Yarn* utiliza el símbolo "->" para indicar una toma de decisión. En el juego puede haber hasta un total de 4 botones de opción, por lo que se pueden colocar hasta 4 símbolos "->" en los archivos de lenguaje *Yarn*. Cuando el usuario haya tomado una decisión, el texto a mostrar será aquel que se encuentre debajo de la toma de decisión. Es decir, en el código mostrado anteriormente se muestran dos opciones: "*Seguir jugando*" e "*Irse a dormir*". Si el jugador escoge la decisión "*Seguir jugando*" el texto que se mostrará a continuación son las 6 líneas tabuladas que se encuentran justo debajo del texto "-> *Seguir jugando*".

Para poder diferenciar entre el diálogo compartido, es decir el diálogo que se muestra sin importar qué decisión se haya tomado, y el diálogo único a cada decisión, las líneas posteriores a línea de decisión deben de estar tabuladas y sin ninguna línea vacía, ya que *YarnSpinner* lo interpretará como que el diálogo es común a las decisiones. En este ejemplo, las líneas de narración son únicas a cada opción, sin embargo, la función *<<Go @ Day2>>* se ejecutará sin importar qué decisión se ha tomado. Se puede observar el resultado final del código en la *Figura 12:*

Figura 12. Ejemplo toma de decisiones

Por último, se encuentran la ejecución de funciones dentro de un archivo de texto de tipo *Yarn*. *YarnSpinner* detecta una función cuando ésta se encuentra entre los símbolos "<<" y ">>". Es decir, la línea en el código anterior "*<<Go @ Day2>>*" es una función denominada "*Go*" y tiene como parámetros el nombre "*Day2*". Esta es una función diseñada para este proyecto y tiene como funcionalidad cargar la escena correspondiente al parámetro establecido en la función, en este caso, "*Day2*", ejecutando la animación que difumina las escenas y guardando las variables establecidas en la clase compartida del jugador.

Como se ha mencionado anteriormente, *YarnSpinner* permite diseñar funciones únicas, como la mencionada previamente. Sin embargo, *YarnSpinner* implementa sus propias funciones que se deben tener en cuenta en la utilización de este lenguaje. Una de ellas se ha mostrado en el código anterior y tiene como estructura: "***<<set $**nombreVariable* **to***

valor>>". Esta función lo que permite es guardar una variable, que debe de tener el símbolo "*$*" para que *YarnSpinner* lo identifique como tal, y permite realizar el guardado en cualquier formato. Este puede ser un numérico, ya sea un *float*, un *integer*, un *string* o también un simple *boolean*. Para este proyecto, se han guardado variables en formato *boolean*, pudiendo obtener de esta manera una diversificación de diálogos, cada uno de ellos diferentes dependiendo de la decisión tomada por el jugador. En este ejemplo, si el jugador decide seguir jugando e ignorar el consejo de su madre, el código guardará la variable "*$jugarTarde*" como "*true".* En la siguiente escena, dado que el diálogo se encuentra en un entorno distinto donde se ha guardado la variable, se utilizará el comando "<<*if*>>" para comprobar el estado de la variable y, si este es equivalente a "*true*", se mostrará el diálogo acorde a la decisión tomada. En caso contrario, se mostrará una conversación distinta. La función "<<*if*>>" tiene la misma funcionalidad que el comando condicional de programación, siendo éste simplemente implementado a lenguaje *Yarn.*

```
<<Scene @ habitacion>>

<<Act @ User, user,left,center,grey>>

// AUDIO DESPERTADOR

El sonido del despertador irrumpió en mi sueño
<<if $jugarTarde is true>>
      Después de estar jugando hasta la madrugada, no tenía suficiente energía
para levantarme de la cama.
      …
<<else>>
      Me levanté de la cama.
      …
<<endif>>

<<SceneChange @ clase>>

Ya había sonado la campana, pero el profesor no había llegado todavía a clase.
```

En este ejemplo, si la variable "*$jugarTarde*" equivale a "*true*", se mostrará las líneas de diálogo dentro de esa condición. En caso contrario, el comando "<<*else*>>" indica las líneas que se deben mostrar. La condición debe terminar siempre con un "<<*endif*>>" para indicar a *YarnSpinner* cuándo termina ésta y continuar de esta manera con el diálogo normal. En

este caso, la última línea del código se encuentra fuera de la condición, por lo que se mostrará sin importar el valor de la variable "*$jugarTarde*".

2.3 *CLIP STUDIO PAINT*

Clip Studio Paint es un programa de software centrado en el diseño gráfico, habilitando múltiples herramientas de dibujo a sus usuarios que podrán utilizar para realizar sus diseños artísticos [8]. Se ha utilizado esta herramienta para poder realizar el diseño de personajes y fondos necesarios para la creación de la historia de este videojuego.

En este proyecto, hay un total de 3 escenarios; el hogar del protagonista, su habitación y el aula del colegio, añadiendo además un fondo para el menú de inicio y una la televisión perteneciente a la escena final de la historia. A su vez, se han diseñado un total de 4 personajes: lo padres del jugador, su amiga del colegio y el avatar que representa el depredador sexual que intenta manipular al protagonista. Además de esta herramienta de dibujo, se ha utilizado una tableta gráfica junto a este software para poder realizar los diseños mencionados anteriormente.

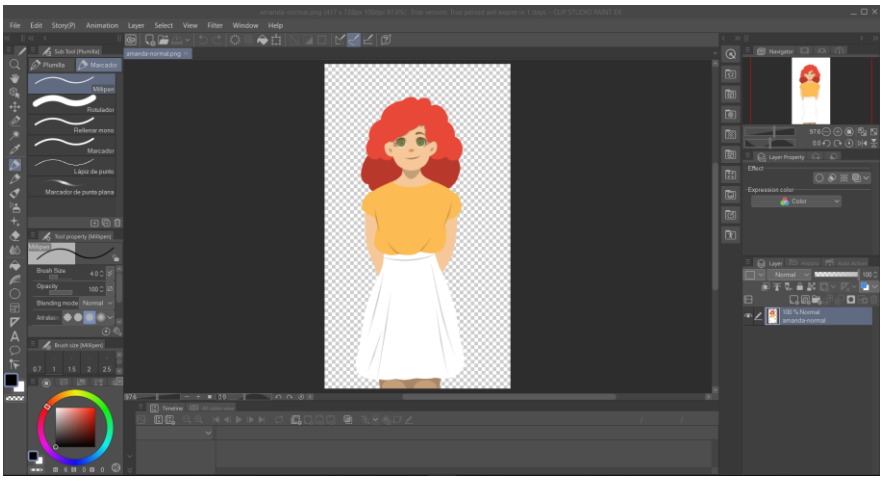

Figura 13. Interfaz de Clip Studio Paint junto con imagen del personaje "Amanda"

2.4 FIREBASE

Firebase es una plataforma orientada a móviles de *Google* que permite desarrollar bases de datos guardadas en la nube y poder implementarlas a aplicaciones móviles diseñadas, donde también se encuentran entre ellas videojuegos para móviles desarrollados por *Unity* [9].

Para este proyecto, se desea almacenar la información relevante del jugador (ésta siendo rasgos característicos como su edad, género y nombre) además de las decisiones tomadas por éste y, a consecuencia de ello, su final obtenido en la historia, para poder habilitar en un futuro un estudio sobre las decisiones que han tomado los usuarios que han participado en este videojuego. En otras palabras, esta implementación de una base de datos proporciona la posibilidad de realizar un estudio futuro sobre la conciencia que tienen los jugadores sobre *Online Grooming*, representando su conocimiento sobre el tema y sus capacidades para reconocer técnicas manipulativas. Además, para poder utilizar *Firebase* en *Unity*, *Firebase Google* proporciona un tutorial sencillo donde se muestran los pasos que se deben realizar para implementar la conexión de la aplicación con la base de datos [10]. Finalmente, se ha diseñado una clase denominada *FirebaseConnection* en el desarrollo de este proyecto que se encarga de establecer la conexión con la base de datos y de realizar la escritura de la clase *"Player"*, donde se han guardado todos los datos del jugador. En futuros apartados se explicará más detalladamente el proceso de guardado a la base de datos y la estructura de la clase "*Player*" mencionada.

Se puede observar la estructura resultante de la base de datos en la *Figura 14:*

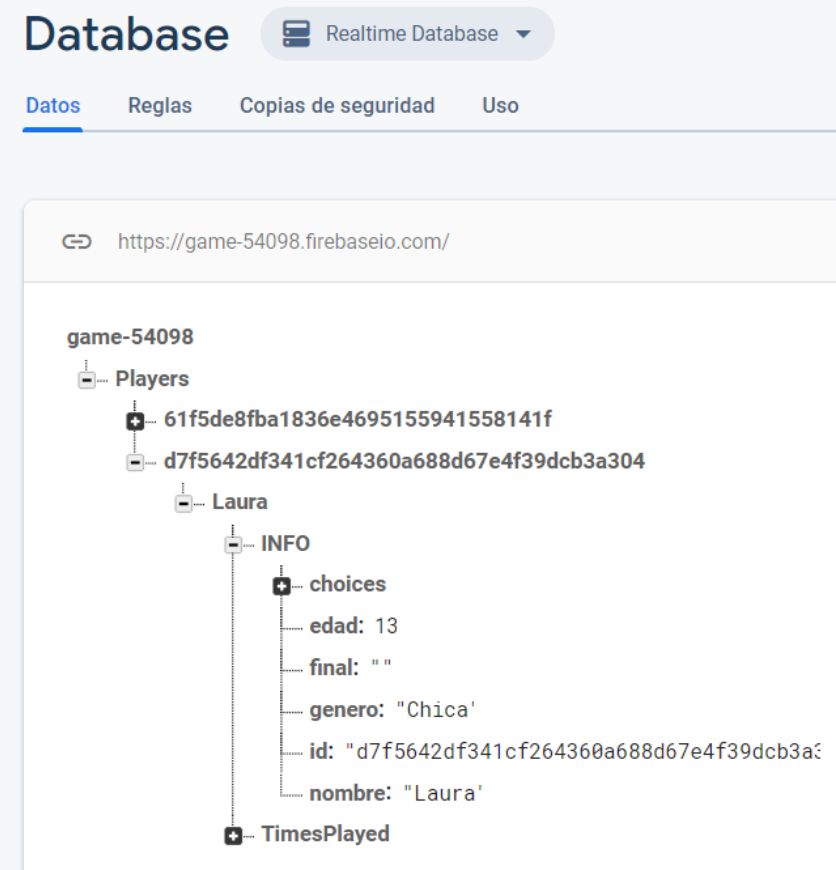

Figura 14. Estructura base de datos en Firebase

Capítulo 3. ESTADO DE LA CUESTIÓN

Este trabajo de fin de grado tiene como objetivo diseñar un videojuego perteneciente a la categoría de "*Juegos serios*", utilizando un motor de videojuego multiplataforma implementado en múltiples ocasiones en la industria de los videojuegos. Además, se desea utilizar el lenguaje *Yarn* para facilitar la implementación de toma de decisiones y la diferenciación de diálogo en la aplicación. En este capítulo, se mostrarán trabajos pertenecientes al mismo género que el proyecto a diseñar, junto con aplicaciones actuales en el mercado que hayan utilizado el lenguaje de diálogo mencionado previamente.

3.1 ESTADO DE LA CUESTIÓN

La industria de los videojuegos ha ido creciendo en estos últimos años en paralelo con la evolución de las nuevas tecnologías, obteniendo un público cada vez mayor y abarcando nuevos géneros y temáticas. Una de estas temáticas es la ciberseguridad, tema relacionado con este proyecto, por lo que podemos obtener múltiples juegos y referencias que nos pueden ayudar para este trabajo. [11]

3.1.1 "NIGHT IN THE WOODS"

Figura 15. Night in the Woods

"Night in the woods" es un videojuego multiplataforma diseñado por *Infinite Fall*. Este videojuego ha utilizado como motor *Unity*, misma plataforma de desarrollo implementada en este trabajo, junto con el lenguaje *Yarn* como motor de diálogo [12].

Este videojuego es una aventura donde el protagonista "*Mae*" regresa a casa después de abandonar sus estudios universitarios. Sin embargo, al alojarse nuevamente en casa de sus padres, descubre un oscuro secreto que le lleva a un bosque cercano, en búsqueda de los secretos de la ciudad y de la desaparición de su viejo amigo *Casey*. A pesar de que la temática de este juego es completamente distinta al proyecto en cuestión, es un gran ejemplo de cómo *YarnSpinner* facilita la implementación de diálogo en juego móvil y es altamente útil para el proyecto en cuestión.

3.1.2 CYBERSCOUTS

Figura 16. Juego Cyberscouts

Una de las aplicaciones pertenecientes al género "*juegos serios*" que se ha podido encontrar se llama "*Juego de Cyberscouts*" [13](*Figura 16*), desarrollado por IS4K (*Internet Segura for Kids*. Este videojuego es una aplicación para ordenadores y tabletas donde se va poniendo a prueba a los jugadores sobre el conocimiento que poseen acerca de ciberseguridad. Además, tiene como finalidad mostrar y enseñar nueva información sobre el tema, educando de esta manera a los usuarios sobre un so seguro de Internet. Tanto padres como hijos (mayores de 6 años), pueden jugar a este juego, ya que esta aplicación proporciona dificultades distintas (fácil, medio y difícil) y dos interfaces separadas, uno para los niños y

otro para los adultos. La aplicación abarca distintos temas relacionados con Internet y seguridad como, por ejemplo: correo malicioso, virus en páginas, piratas de identidad, etc.

A pesar de que este videojuego forma parte del mismo género que el proyecto en cuestión y abarca temas relacionados sobre los peligros de Internet, los minijuegos que utiliza no profundizan tanto como se desearía sobre las consecuencias que estas amenazas pueden crear. Algunos de estos niveles son: una sopa de letras (*Figura 17*), juego de parejas (*Figura 18*), abordaje de piratas, etc. Es cierto que estos juegos están pensados para un público de edad menor, como bien muestra la edad límite de 6 años, y se requieren juegos sencillos y visuales. En este trabajo de fin de grado, sin embargo, está centrado para un público mayor, de 10 a 14 años, por lo que se debe utilizar minijuegos más complejos y realistas que enseñen las graves consecuencias que un mal uso de Internet puede conllevar.

Figura 17. Juego sopa de letras de Cyberscout

Figura 18. Juego sopa de parejas de Cyberscouts

3.1.3 CONECTADOS

Otro juego que se acerca más a la estructura que este proyecto desea abarcar, es un videojuego para ordenador llamado "*Conectado*" [14] desarrollado por e-UCM. Este videojuego es una novela visual, mismo concepto que se desea desarrollar en este trabajo, donde el protagonista se cambia de colegio y debe sobrepasar por todos los aspectos que conlleva este cambio de vida, en ellos incluido el posible acoso o, en inglés, *bullying*. El jugador tiene 5 días donde se muestran las durezas que el protagonista debe sobrepasar, entre ellos el constante *bullying* tanto en persona por parte de sus nuevos compañeros como en las redes sociales.

En múltiples ocasiones, se le presentará al jugador opciones que deberá tomar sobre la vida del protagonista y que establecerán tanto su personalidad como las consecuencias que esas decisiones conllevan. En otras palabras, el jugador tiene el control sobre la vida del protagonista y su relación con el resto de los personajes. Este juego no sólo muestra aspectos sociales y relacionados con el *bullying* y maltrato, sino que también aparecen situaciones relacionadas con Internet y ciberseguridad como, por ejemplo: chantaje, robo de contraseñas y robo de identidad en las redes sociales; conceptos que se desean introducir en este proyecto.

Figura 19. Juego Conectado

Figura 20. Toma de decisiones en Conectado

"*Conectado*" es un gran ejemplo que muestra el resultado al que se desea llegar con este trabajo de fin grado. Asimismo, en este proyecto se utilizará una base de datos donde se

almacenarán los datos de los jugadores (género, edad, etc.) junto con las decisiones tomadas en el juego y los finales obtenidos.

3.1.4 OTROS VIDEOJUEGOS RELACIONADOS

Existen muchos otros videojuegos relacionados con la *ciberseguridad*. Algunos de ellos son más técnicos como por ejemplo *Cyber Awareness Challenge* [15], donde el jugador debe de tomar las decisiones correctas para proteger su información y no ser *hackeado* por usuarios maliciosos. Otro ejemplo también puede ser *Zero Threat*, donde se muestra una red que está en constante ataque y que el jugador deberá utilizar diferentes recursos informáticos para protegerla.

A pesar de que estos videojuegos muestren los peligros de Internet y aumenten la conciencia de los usuarios sobre estas amenazas, mostrando diversas medidas de protección ante ellas, apenas se han podido encontrar aplicaciones relacionadas sobre *Online Grooming*. Podemos encontrar sobre *ciberseguridad*, *ciberbullying*, malware, etc.; pero ninguno sobre los depredadores sexuales que se esconden tras el anonimato de Internet. Con este proyecto, sin embargo, podemos proporcionar un *"juego serio"* donde se introduce un nuevo concepto que muchos jóvenes desconocen y es imprescindible aprender a defenderse de él.

3.2 TRABAJO DE INVESTIGACIÓN UTILIZADOS

Dado que este proyecto entra dentro de la categoría "*juego serio*" y se desea recrear un contexto realista, se han estudiado diversos trabajos psicológicos que investigan tanto el comportamiento de los jóvenes en Internet, como los perfiles característicos tanto de los depredadores sexuales como las víctimas que caen bajo su manipulación.

3.2.1 II ESTUDIO ACOSO ESCOLAR Y CIBERBULLYING – FUNDACIÓN ANAR

El "*II Estudio sobre Acoso Escolar y Ciberbullying según los afectados*" es una investigación realizada por la Fundación ANAR junto con la colaboración de Fundación Mutua Madrileña [16]. Este segundo estudio sobre el acoso escolar tiene como objetivo conocer y estudiar la evolución del acoso escolar y del *ciberbullying* entre las víctimas,

analizando características similares que éstos comparten y los entornos en los que se encuentra.

A pesar de que el tema de este estudio no coincida con el problema que se quiere tratar con este proyecto, éste siendo *Online Grooming*, los resultados obtenidos en el estudio han ayudado a realizar un perfil genérico de la víctima del acoso, pudiendo aplicar dicho perfil al protagonista de esta historia.

Una de las gráficas pertenecientes a este estudio y que se ha utilizado en el diseño de esta historia es "*Problemas asociados en la víctima por acoso escolar*" que se representa en la gráfica 90 del documento, en este caso en la *Figura 21* que se muestra a continuación. En dicha gráfica, las principales características que sufre el joven a ser víctima de un acoso escolar son: la tristeza, ansiedad, miedo, soledad, aislamiento, dificultad con los compañeros y baja autoestima. Estas características se han utilizado para describir las emociones que siente el protagonista en caso de que haya compartido fotos sexuales con el agresor y éste las haya subido online, comenzando de esta manera el *ciberbullying* por parte de los compañeros al ver dichas imágenes. En ese entorno, se ha intentado representar las emociones por las que pasa una víctima de acoso, intentando mostrar de esta manera al jugador la situación en la que se podría encontrar si repitiese los mismos pasos que ha realizado el protagonista en la vida real.

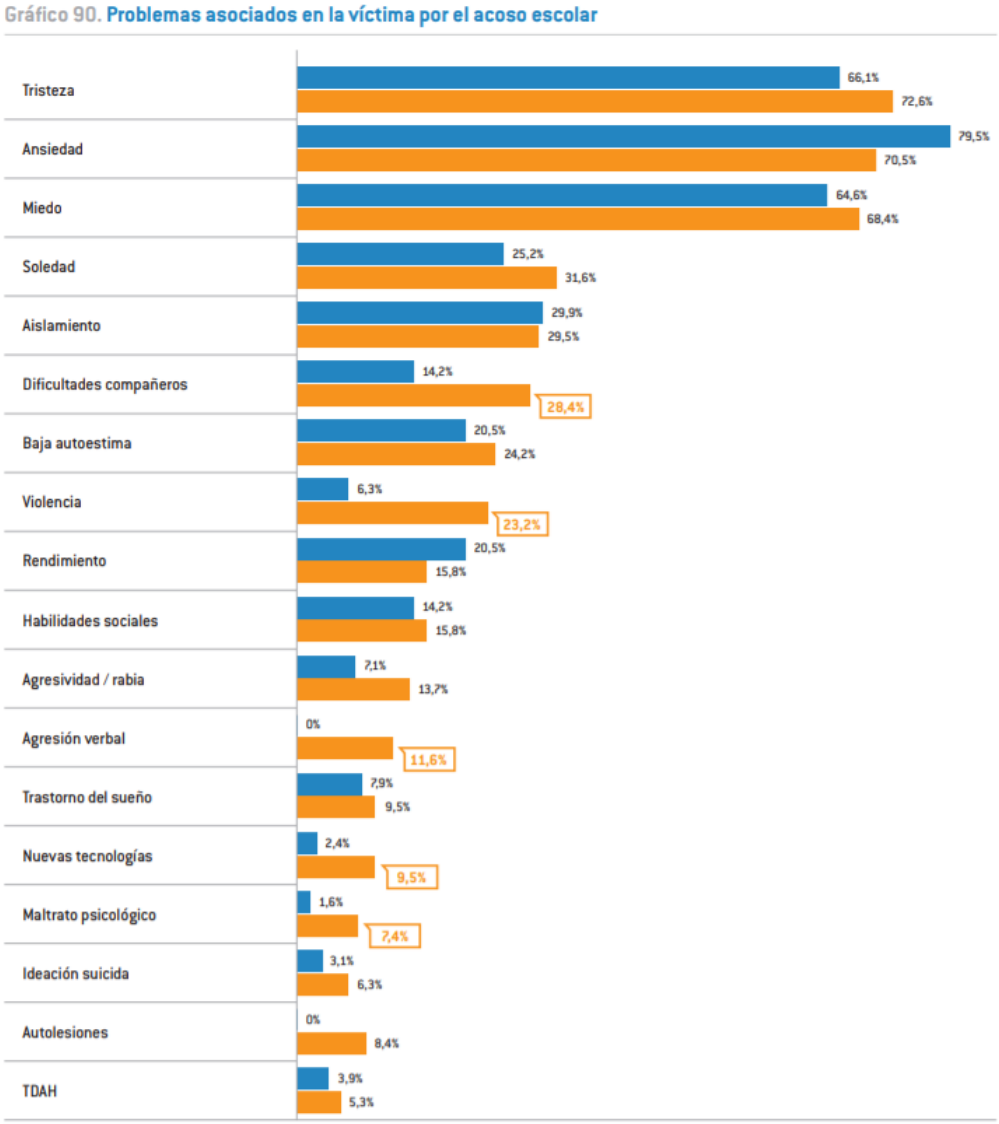

Figura 21. Problemas asociados a víctimas de acosos escolar – II Estudio Acoso Escolar y Ciberbullying, Teléfono ACNAR

Otra gráfica informativa que se ha utilizado en el diseño de esta historia es la denominada "*Tipos de* ciberbullying", que se muestra a continuación en la *Figura 22.* En dicha gráfica se pueden observar los tipos acosos se suelen encontrar en Internet, éstos siendo: insultos o palabras ofensivas, amenazas, fotos y vídeos comprometidos y difusión de información personal. Estos tipos de acoso, a pesar de que no se encuentren en un entorno escolar sino realizados por un depredador sexual, se han utilizado como métodos dañinos que el acosador ha utilizado sobre la víctima en la historia de la aplicación, especialmente el uso de

amenazas, fotos y vídeos comprometidos y difusión de información personal. En caso de que el protagonista haya sido manipulado por el agresor y haya confiado en él, enviando información personal y fotos comprometidas, el acosador utilizará dicha información para amenazar al protagonista, terminando por publicar todo el contenido en Internet. Sus compañeros, al ver dichas imágenes, comenzarán a insultar al protagonista en la página web, comenzando con el *ciberbullyin* escolar y trasladando ese acoso a la vida real, marginando al protagonista y aislándole del resto de la clase.

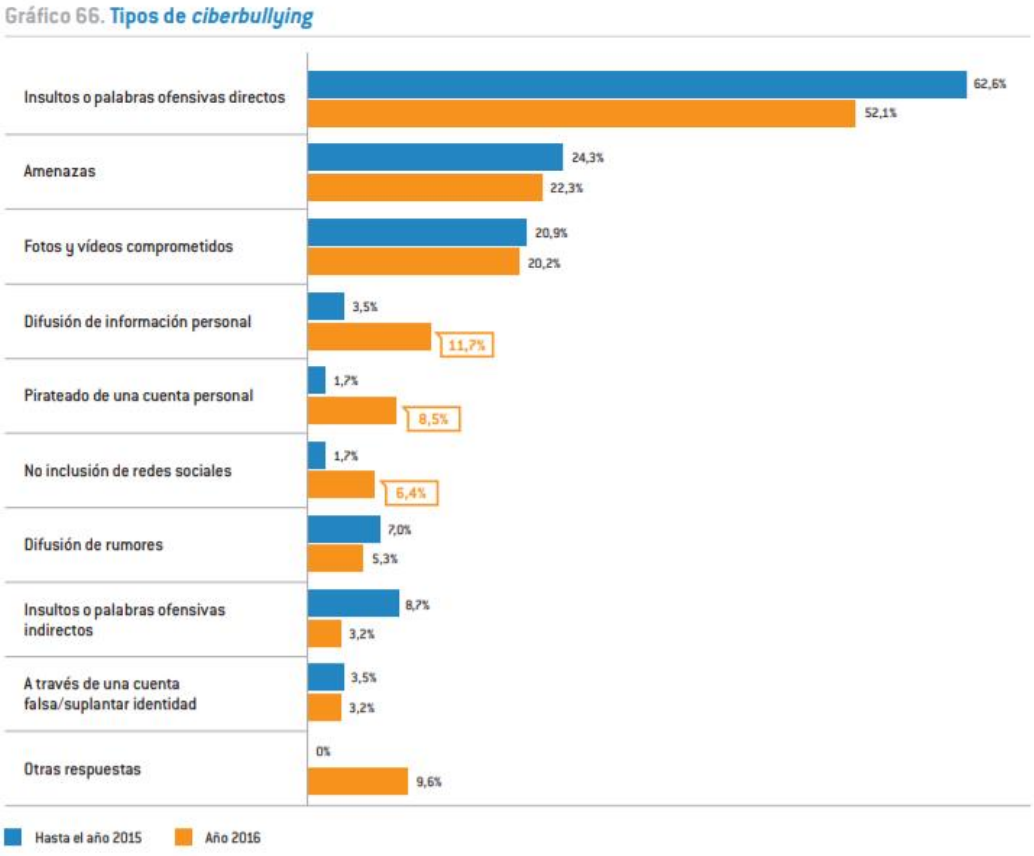

Figura 22. Tipos de ciberbullying – II Estudio Acoso Escolar y Ciberbullying, Teléfono ACNAR

Este estudio, a pesar de tener una temática distinta a la que se discute en este proyecto, ha sido altamente útil para estudiar y analizar los entornos en los que se encuentran diversas víctimas de acosos, pudiendo comprender además las emociones que siente la víctima del acoso, trasladándolas a la historia que se quiere diseñar para este proyecto.

3.2.2 VIOLENCIA VIRAL – SAVE THE CHILDREN

El informe de "*Violencia Viral*" desarrollado por la fundación "*Save the Children*" es un análisis sobre la violencia contra la infancia y la adolescencia en el entorno digital, donde se muestran los diversos tipos de violencia *online* a los que se puede enfrentar un menor ante el constante uso de las tecnologías y dispositivos electrónicos [17].

En este trabajo se definen múltiples tipos de violencia viral, junto con sus descripciones y ejemplos actuales del acoso. En este trabajo, además, se muestra una relación entre todas las violencias explicadas y una secuencia de acciones que pueden ocurrir en el proceso del acoso. Hay un esquema en dicho documento donde se muestra una cadena de acontecimientos por las que pasa un usuario y los distintos tipos de violencia online que sufre. Para este trabajo, la historia diseñada ha seguido fielmente uno de estos procesos (*Figura 23*), mostrando diferentes ejemplos de acoso que puede conllevar *Online Grooming.*

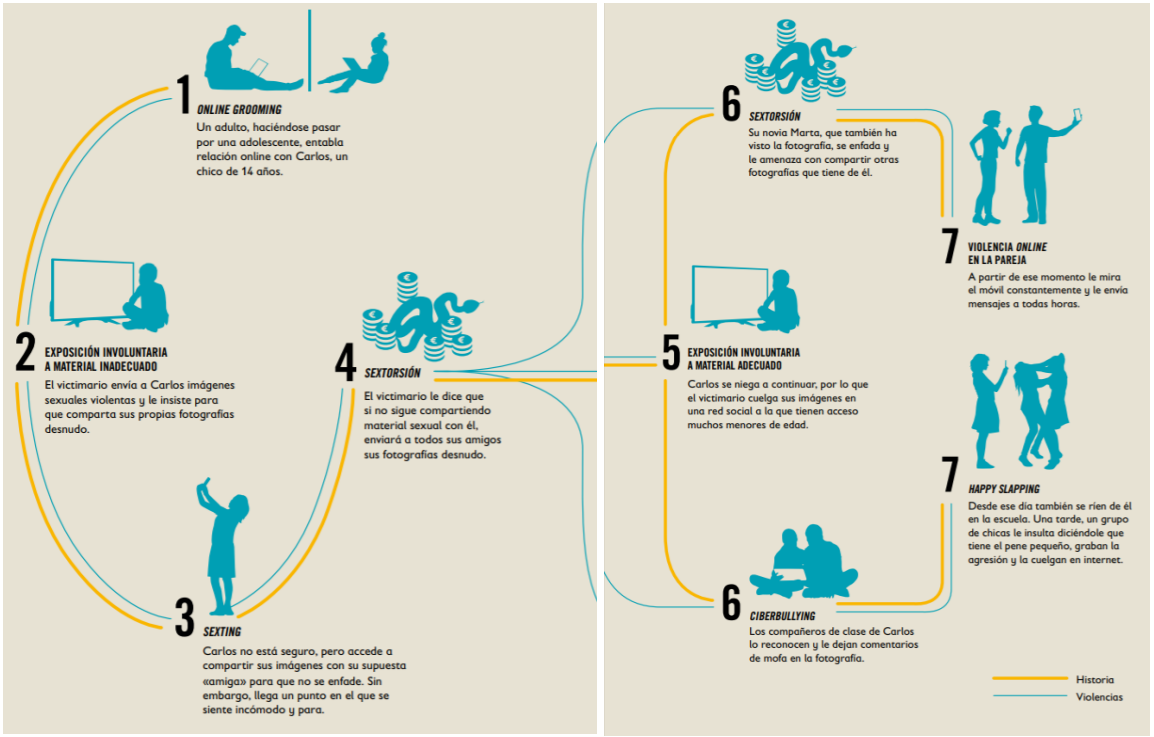

Figura 23. Relación entre las distintas violencias – Violencia Viral, Save the Children

Para la historia de este proyecto, las violencias utilizadas, y que se han podido observar en la figura anterior, son las siguientes:

- *Online Grooming*: acoso y abuso sexual *online*, donde una persona adulta contacta electrónicamente con un menor, manipulando lentamente al usuario con el propósito de involucrarse en una actividad sexual.
- *Exposición involuntaria a material inadecuado:* material de contenido sexual y/o violento es expuesto al usuario sin el consentimiento de éste. En otras palabras, una persona desconocida y/o familiar envía dicho contenido al usuario sin su permiso.
- *Sexting sin consentimiento:* intercambio de mensajes o material con contenido sexual. El *sexting* sin consentimiento ocurre cuando dichos mensajes sexuales son enviados a terceras personas sin el consentimiento del usuario propietario de dicho contenido.
- *Sextorsión:* combinación de la palabra sexo y extorsión que indica el chantaje/amenaza de publicar en redes sociales y/o páginas web material o información privada de contenido sexual del usuario
- *Ciberbullying:* hostigamiento hacia la víctima a través de mensajes, imágenes o vídeos, con intención de dañar y humillar a la víctima.

Además, otro material que se ha utilizado en el diseño de la historia de este proyecto, en este informe sobre *Violencia Viral* se proporciona un perfil característico de las víctimas que han sufrido acosos sexuales en Internet. Este perfil ha sido crucial en el desarrollo del entorno del protagonista y la personalidad que éste tiene. Dicho desarrollo se explicará más detalladamente en futuros apartados, pero cabe comentar que el análisis proporcionado por este informe ha ayudado en gran medida al desarrollo principal de la historia.

Las características principales que se pueden observar en la víctima según este trabajo, representados en las *Figura 24* y *Figura 25*, y que se han utilizado en el desarrollo de la historia son:

- Edad de la víctima: entre 10 y 14 años
- Rasgos psicológicos: Soledad, baja autoestima

- Comportamientos de riesgo: tener un dispositivo propio, pasar mucho tiempo en línea, chatear con personas desconocidas, usar Internet en el dormitorio, enviar mensajes despectivos en las redes.
- Problemas en su vida cotidiana: relaciones familiares complicadas y conflictos en el hogar
- Falta de comunicación en la familia sobre seguridad digital: Mantienen en secreto la violencia sufrida.

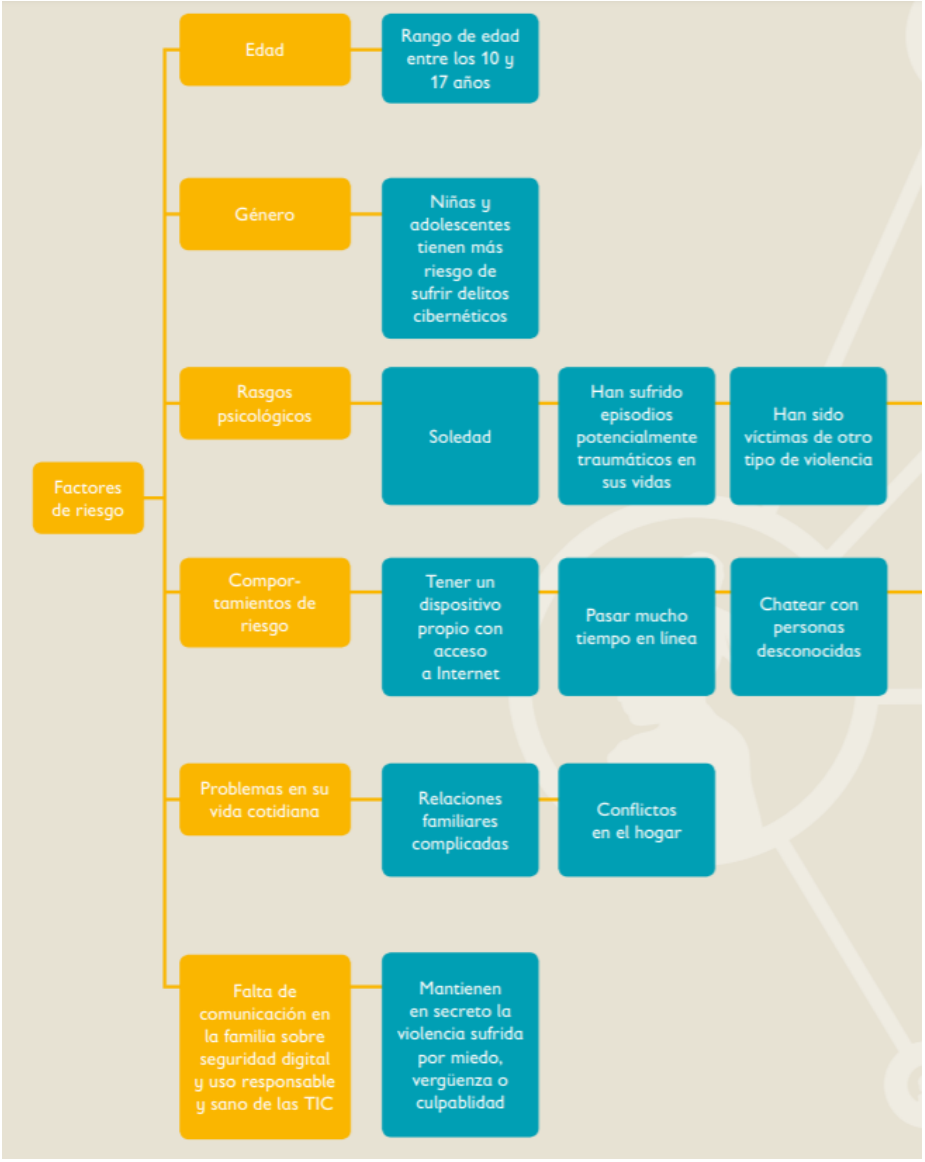

Figura 24. Características principales de las víctimas [1] – Violencia Viral, Save the Children

Figura 25. Características principales de las víctimas [2] – Violencia Viral, Save the Children

3.2.3 TRABAJO FIN DE GRADO SOBRE ONLINE GROOMING DE PATRICIA ALONSO

Otro trabajo relacionado con *Online Grooming* que se ha utilizado para este proyecto es el trabajo de fin de grado realizado por la estudiante de comillas en la facultad Ciencias Humanas y Sociales Patricia Alonso, titulado "*Online Grooming*" [18].

Este trabajo de fin de grado estudia la problemática actual sobre *Online Grooming*, investigando sobre los métodos que utiliza el agresor con sus víctimas, los pasos que suele utilizar en la manipulación, además de los perfiles tanto de la víctima como los del agresor. A pesar de que estos perfiles hayan aportado en el desarrollo del agresor y de la víctima, este trabajo ha contribuido en entender el proceso que suele seguir el agresor en la manipulación y engaño hacia el usuario. A pesar de que dicho proceso no sea exacto para todos los agresores, ha ayudado a poder realizar la estructura de la historia y establecer el entorno dañino que el jugador debe de reconocer para aplicar dichos conocimientos en la vida real.

Los resultados del estudio realizado en dicho trabajo de fin de grado muestran el siguiente *Modus Operandi* del agresor:

1. Formación de la Amistad: conocer al menor, intercambio de fotografías para verificar la edad del usuario y que éste se encuentre dentro de su prototipo.

2. Formación de la Relación: aumenta la relación con el menor para conseguir una mayor intimidad. Pregunta acerca la vida personal del usuario, aumentando la confianza entre ambos.
3. Estado de Evaluación del Riesgo: intenta controlar aspectos personales de la vida del menor que puedan exponer su relación con éste.
4. Estado de Exclusividad: hace que el menor se sienta especial y va introduciendo de forma gradual temas sexuales, con la finalidad de aumentar más su confianza y demostrando al usuario que pueden hablar de cualquier cosa.
5. Estado Sexual: la conversación contiene elementos sexuales que pueden variar desde simples solicitudes hasta insinuaciones directas. Puede justificar sus intenciones con fines educativos.
6. Etapa Final: finalmente, el agresor persigue a la víctima para reestablecer la relación y minimizar el riesgo de divulgación o exposición a los familiares.

En el desarrollo de la historia de esta aplicación, estos estados de manipulación al menor representan un día específico dentro del juego, comenzando con la formación de la amistad, aumentando la relación introduciendo lentamente temas sexuales, comprobar la exclusividad del menor realizando preguntas sobre su entorno y estado familiar hasta, finalmente, realizar la solicitud sexual al usuario.

Como se ha podido observar, estos trabajos investigativos han establecido las bases que forman la historia de este proyecto, pudiendo de esta manera crear un entorno realista que el jugador pueda utilizar en la vida real para identificar procesos manipulativos de otros usuarios online.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

A pesar de que la conciencia hacia los peligros de Internet haya aumentado en los jóvenes adolescentes gracias a las múltiples iniciativas realizadas por centros educativos, se deben encontrar nuevos métodos, o realizar un mayor uso de otros existentes, para no solo enseñar sobre dichas amenazas, sino también para captar su interés y conseguir que ellos mismos tomen la iniciativa de educarse y protegerse.

4.1.1 INCREMENTO EN EL USO DE INTERNET EN LOS JÓVENES

A medida que van evolucionando las tecnologías, la dependencia que la humanidad tiene hacia ella también incrementa con el tiempo. Ya que los más vulnerables hacia las amenazas y manipulaciones de Internet son los jóvenes y menores, hay que prestar cierta atención a la cantidad de tiempo que pasan en Internet. El informe de "*Violencia Viral*" elaborado por "*Save the Children*" [17], documento mencionado en apartados previos, proporciona una gráfica ilustrando perfectamente la cantidad de horas que los jóvenes pasan en Internet (*Figura 26*).

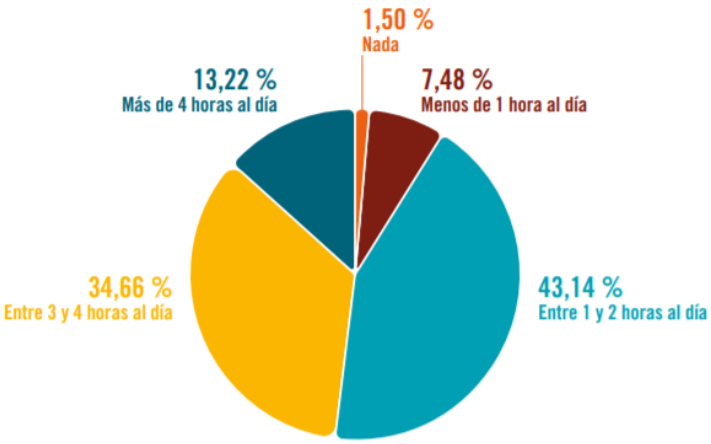

Figura 26. Frecuencia en el uso de Internet – Violencia Viral, Save the Children

En esta gráfica, se puede observar que el 47.88% de los encuestados por *Save the Children* pasan más de 3 horas diarias en Internet, con otro 43.14% entre 1-2 horas. En otras palabras, el 90% de los encuestados en esta investigación usan diariamente Internet, con un uso variado de horas, pero de todas formas una cifra extremadamente alta. Ese uso elevado de la tecnología les convierte en un claro objetivo de los peligros de Internet, blanco que se vuelve cada vez más vulnerable cuanto más se incremente dicho uso.

En el trabajo de fin de grado desarrollado por Patricia Alonso sobre *Online Grooming* [18], proporciona más datos sobre el elevado uso de Internet por los jóvenes. En este trabajo se ilustra en una gráfica (*Figura 27*) las horas que adolescentes de edades entorno los 15 años pasan en Internet. En dicha gráfica, se muestra que la gran mayoría se conectan entre 2 o 4 horas diarias, datos que coinciden con la gráfica mostrada anteriormente perteneciente al informe sobre *Violencia Viral*, mostrando una vez más la cantidad de horas que pasan los jóvenes conectados a la red.

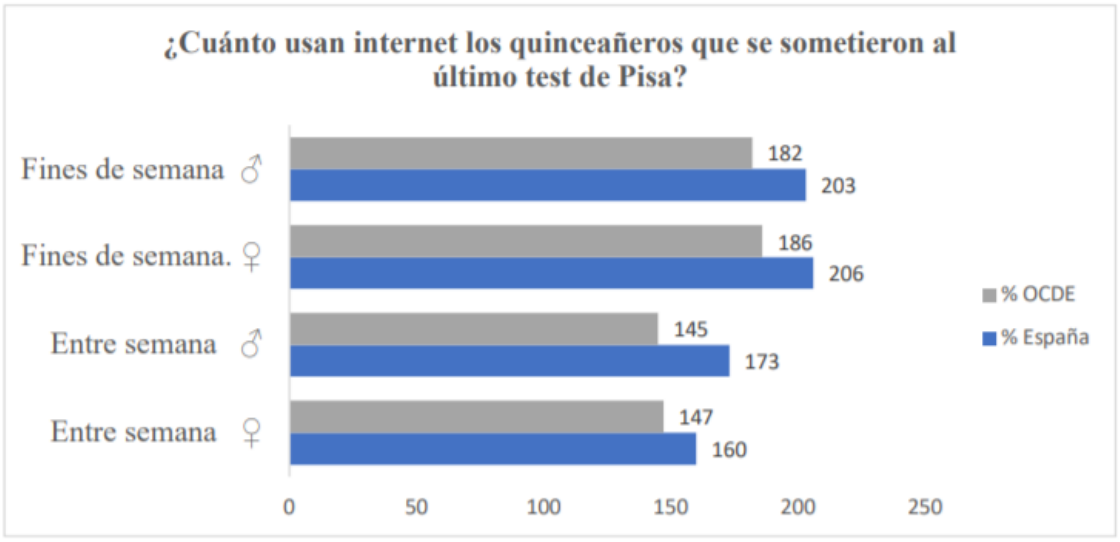

Figura 27. Uso (en horas) de Internet en adolescentes – TFG sobre Online Grooming, Patricia Alonso

Previamente en el informe de *Violencia Viral*, se indicó que una de las características vulnerables que sufren las víctimas de *Online Grooming* es el elevado uso de Internet junto con la disponibilidad de un dispositivo electrónico propio, dando la oportunidad a los agresores online de manipular a los menores de manera pausada y sin el conocimiento de los

adultos responsables. Estos datos obtenidos de trabajos previos muestran la necesidad de aumentar la conciencia a los jóvenes sobre los acosadores sexuales que se esconden en el anonimato de la red y, dado el elevado uso de Internet, mostrar la necesidad de mantenerse siempre protegidos en la red.

4.1.2 USO DE APLICACIONES MÓVILES POR LOS JÓVENES

Previamente se ha mostrado el incremento de dependencia que sufren los adolescentes españoles a Internet. En este apartado, sin embargo, se mostrará cuánta de dicha dependencia se debe a los móviles, justificando de esta manera la creación de una aplicación móvil para este proyecto en comparación con cualquier otro dispositivo electrónico.

En 2019, casi un 90% de los adolescentes son dueños de un "*smartphone*", según *Epdata*, una página web donde se almacenan múltiples datos mundiales y se generan diversas gráficas informativas [19]. En dicha página, además, se muestra que, en un estudio elaborado por Google, FAD y BBVA, el 89.9% de los adolescentes encuestados disponen de un teléfono móvil, mientras que un 76% tienen a su vez y ordenador portátil. En la gráfica mostrada en la *Figura 28*, se puede observar claramente la diferencia de los dispositivos utilizados por los adolescentes y jóvenes:

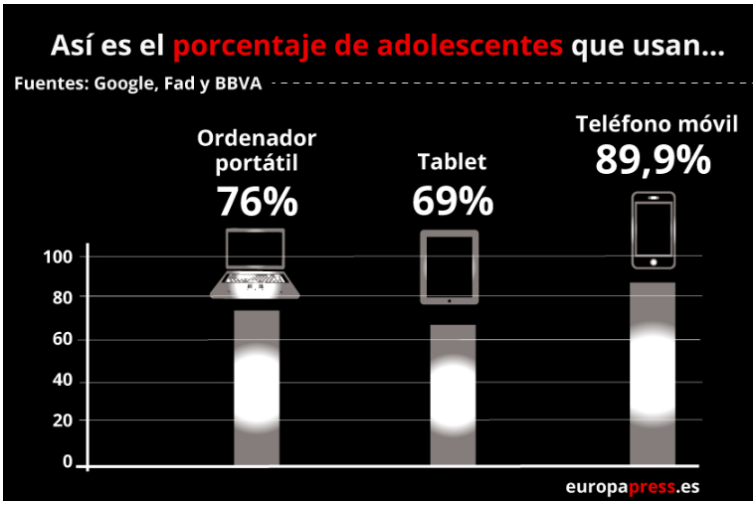

Figura 28. Porcentaje de adolescentes que disponen de ciertos dispositivos tecnológicos - Epdata

Esta información demuestra que los adolescentes utilizan en gran medida el teléfono móvil. Utilizando esta dependencia hacia este dispositivo tecnológico, se puede captar el interés de dicha población para poder crear una aplicación móvil que les eduque sobre los peligros de *Online Grooming*. En otras palabras, desarrollando una aplicación móvil en vez de otro programa compatible con otro dispositivo, se podrá captar más fácilmente el interés de los adolescentes, obteniendo su atención para mostrar y educar, en vez de utilizar dicho dispositivo simplemente por entretenimiento.

4.1.3 ESCASEZ DE JUEGOS SOBRE ONLINE GROOMING

Previamente, en el capítulo 3 de este trabajo, indicado el Estado de Cuestión de este proyecto, se han mostrado diversos videojuegos que exploran los peligros de Internet y tienen una iniciativa educativa hacia los jugadores. Entre los mencionados se encuentra *Conectado* y *Cyberscouts*. El primero habla sobre *bullying* escolar y las dificultades que pasa el protagonista de la historia a tener que enfrentarse al acoso de sus compañeros. El último, sin embargo, muestra de manera interactiva a través de minijuegos los diversos peligros que se pueden encontrar en Internet, entre ellos la posibilidad de encuentro con *hackers* o virus en páginas web poco fiables. Estos dos juegos entran dentro del género de "*juegos serios*" con el objetivo de educar a los jóvenes a través del entretenimiento.

Se pueden encontrar otros múltiples juegos que enseñan sobre ciberseguridad y diferentes tácticas al enfrentarse a las amenazas de Internet como, por ejemplo, *Cyber Awareness Challenge*, juego desarrollado por el ejército donde ocurren múltiples incidentes relacionados con la ciberseguridad donde el jugador debe resolver, o también *Cybersecurity Lab*, donde el jugador representa un jefe de una compañía de Internet que se debe enfrentar a diversos problemas en su compañía y debe resolverlos [15]. Todos estos juegos se centran en una temática de ciberseguridad, enseñando tácticas de protección ante *hackers* o virus. A pesar de que dicho tema tenga equivalente importancia a *Online Grooming*, apenas se pueden encontrar juegos educativos que discutan este tema.

Debido a la carencia de un *juego serio* que eduque sobe las tácticas manipulativas que utilizan los agresores online y el incremento de uso de Internet que se observa en la

población, aumentando su vulnerabilidad hacia estos depredadores sexuales, este proyecto abarcará este problema y se convertirá en uno de los primeros que eduque sobre esta amenaza de Internet.

4.2 OBJETIVOS

Los objetivos principales que se persiguen con este proyecto son:

1. Realizar un estudio sobre las amenazas de Internet: realizar un perfil de amenazas y depredadores que existen en Internet e implementarlas en el proyecto. De esta manera, no solo se creará un videojuego desarrollado en un entorno realista, sino que además se proporcionar información fiable al jugador de cómo defenderse en este tipo de situaciones y aplicar el nuevo conocimiento adquirido en la vida real.
2. Programación utilizando Unity y diseño de un árbol de decisiones: diseñar la historia en la que se desarrollará el videojuego utilizando un árbol de decisiones. Dicho árbol deberá tener en cuenta el perfil de amenazas realizado anteriormente y utilizar la herramienta *Unity* como entorno de desarrollo.
3. Diseño de una base de datos: diseñar una base de datos donde se almacenarán los datos de los jugadores.

Por otro lado, los objetivos secundarios de este proyecto son:

4. Desarrollar el diseño artístico: una vez finalizado la programación y la historia del videojuego, se procederá con el diseño de personajes y escenarios utilizados en la historia.
5. Validación funcional del prototipo: al finalizar con el prototipo final del videojuego, se realizará un análisis de dicho producto para poder verificar tanto su diseño, en busca de posibles errores en el juego, como su historia, comprobando que el árbol de decisiones se ha implementado correctamente y que sus decisiones tengan un verdadero impacto en el jugador.

4.3 METODOLOGÍA

Para realizar la programación del proyecto y el diseño de la interfaz, se utilizará la herramienta *Unity*, aplicación mencionada apartados previos, como entorno de desarrollo. El código de la aplicación se desarrollará en *Microsoft Visual Studio*, usando *C#* como lenguaje de programación.

Para poder facilitar la programación del videojuego, se utilizarán códigos de otras aplicaciones existentes y se modificarán de tal manera que cumplan con los requisitos de este proyecto. En *GitHub* se ha podido encontrar un código viable y sencillo que permitirá programar el árbol de decisiones del proyecto más rápidamente, utilizando *YarnSpinner* como motor de diálogo. Dicho código se llama *Ropework* [20] y utiliza textos denominados *Yarn*, líneas de texto que definen el entorno de la escena, para crear fácilmente escenarios con diálogo, personajes, música y tomas de decisiones. El código *Ropework* utiliza *Unity* para traducir las líneas los textos *Yarn* al entorno de desarrollo de *Unity* y crea los objetos necesarios para establecer la escena del juego. El desarrollador de este código estableció una licencia MIT de libre uso y modificación, por lo que se podrá utilizar para este proyecto.

Figura 29. Novela visual Ropework - Ropework

Algunas modificaciones que se han realizado en este proyecto son las siguientes: la conexión de la base de datos y la carga de escenas según las decisiones tomadas para crear el árbol de decisiones.

4.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

4.4.1 PLANIFICACIÓN

El cronograma que se ha utilizado en el desarrollo de esta aplicación se muestra en la *Figura 30* encontrada a continuación:

Figura 30. Cronograma proyecto de fin de grado

Como este proyecto se comenzó en septiembre de este curso, la búsqueda de código y la programación del videojuego se encontraban prácticamente finalizadas a comienzos de este año 2020. Sin embargo, se utilizaron los dos primeros meses, enero y febrero, para realizar ciertas modificaciones que contribuían a una mejora en la jugabilidad de la aplicación. Entre ellas se encontraban: diseño del menú de inicio, implementación del guardado de partida y la conexión con la base de datos.

Para poder realizar un videojuego realista que utilizase métodos útiles para la protección de los peligros de Internet, en este caso de *Online Grooming*, durante el mes de marzo se realizó un estudio exhaustivo sobre las características principales que este acoso online presenta. De esta manera, se encontraron diversos trabajos investigativos que proporcionaron información vital para el desarrollo de la historia de esta aplicación. A finales del mes de marzo, al finalizar con el estudio de este peligro online, se procedió escribir la historia del videojuego, aplicando los nuevos conocimientos adquiridos de estos informes psicológicos. Para finales

del mes de abril, se pudo desarrollar un árbol de decisiones que establecería las bases fundamentales de esta aplicación.

Al finalizar con el desarrollo de la historia principal, se procedió a realizar el diseño artístico de este videojuego. Este proceso se ha dividido en dos secciones: los personajes y los fondos. Los primeros se completaron en dos semanas, proporcionando un diseño de los 3 personajes de la aplicación. Por otro lado, durante el resto del mes de mayo, se procedió a terminar con los escenarios de la aplicación, realizando un total de 5 fondos (habitación, salón, aula del colegio, televisión y ordenador) donde se desarrolla los diversos escenarios de la historia.

Finalmente, al terminar con el diseño de la historia y el diseño artístico, se procedió a realizar la implementación de éstos en el código previamente desarrollado. Durante las primeras semanas de junio, se realizó un periodo de validación donde se comprobó que todas las funcionalidades de la aplicación se aplicaban correctamente, pudiendo realizar a su vez una mejora dentro del código previamente diseñado, obteniendo de esta manera un programa óptimo como resultado final. Además, durante este periodo de pruebas, se realizaron diversas conexiones a la base de datos utilizando diferentes dispositivos para comprobar que la escritura a la base de datos se realizaba correctamente.

4.4.2 ESTIMACIÓN DEL COSTE DE DESARROLLO

Durante este proyecto, se han trabajado en 3 campos diferentes que forman la base de la industria de los videojuegos: la rama artística, con diseño de personajes y entornos del videojuego; la rama creativa, donde se desarrolla el diseño de la historia y la creación de personajes; y, finalmente, la zona de programación, donde se desarrolla la base y comportamientos del videojuego.

En el campo de programación, debido a que se ha utilizado la versión de estudiante de *Unity*, donde se proporciona las herramientas básicas en el desarrollo de un videojuego, no existe un coste adicional al realizar un uso de dicha aplicación. Sin embargo, en caso de que se necesite una mayor cantidad de recursos con resultados de calidad superior, existen una variedad de subscripciones de *Unity* que proporcionan dichos componentes. El precio por

esas mejoras varía entre los 40$ al mes (*Unity Plus*, para compañías con una cantidad de ingresos menos de 200 mil dólares al mes en el último año), 150$ al mes (*Unity Pro*, compañías con ingresos mayores a los 200 mil dólares al mes en el último año) y 200$ al mes (*Unity Enterprise*, misma condición que en *Unity Pro*, pero con mayores beneficios que en la suscripción anterior) [21]. Para este proyecto se ha utilizado la versión de estudiante que no es necesario realizar ningún método de pago. Sin embargo, si se desea mejorar este proyecto utilizando herramientas más avanzadas o se trabaja con una compañía con una cierta cantidad de ingresos anuales, se deberá tener en cuenta los costes de estas suscripciones.

El mismo caso ocurre en el diseño artístico de esta aplicación. Como se ha mencionado previamente, se ha utilizado la aplicación *Clip Studio Paint* como entorno de desarrollo artístico. Esta aplicación proporciona un periodo de prueba de 30 días, tiempo suficiente para realizar el diseño de las imágenes utilizadas en esta aplicación. Sin embargo, si futuros desarrolladores necesitan un periodo de mayor longitud o un acceso a herramientas más avanzas de la aplicación, se deberá realizar una suscripción a *Clip Studio Paint*. Esta aplicación proporciona dos subscripciones diferentes, con la posibilidad de comprar la aplicación entera o realizar un pago mensual: la primera es *Clip Studio Paint Pro* con un total de 50$ (o 0.99$ al mes). La segunda, en cambio, es *Clip Studio Paint Ex*, donde esta vez se deberá pagan 219$ por la aplicación entera (o 2.49$ al mes) [22]. La diferencia entre estas dos versiones de la misma aplicación proporciona herramientas distintas, la versión de *Ex* proporcionando una mayor cantidad y de mejor calidad, como por ejemplo animaciones y posibilidad de manejar múltiples dibujos al mismo tiempo.

A pesar de que este proyecto no ha supuesto ningún coste económico, se debe de tener en cuenta el periodo de tiempo en el que se ha dedicado exclusivamente a su desarrollo. Si esta aplicación hubiera sido desarrollada por una empresa, se añadiría a este presupuesto el coste de los sueldos de sus trabajadores. Este TFG comenzó a inicios de este curso 2019-2020, aunque los primeros meses fueron dedicados exclusivamente al aprendizaje de la plataforma *Unity* y a la búsqueda de plantillas existentes que pudieran ser recicladas en esta aplicación. Por este motivo, sólo se tendrán en cuenta los 6 primeros meses de este año 2020 donde se

ha realizado la mayor parte del trabajo al realizar la adaptación del código obtenido y el diseño tanto la historia como las imágenes de esta aplicación. Si aplicásemos una jornada laboral de 6 horas diarias sin tener en cuenta los días festivos y asumiendo que el sueldo a pagar gira entorno los 20 EUR/h, se puede calcular el coste total que este TFG ha supuesto en estos últimos meses: 14.400 euros.

En resumen, exceptuando el sueldo laboral estimado perteneciente a un trabajador de una empresa, este proyecto no ha proporcionado ningún coste económico en su desarrollo. Sin embargo, en caso de que se desee obtener una variación de herramientas que ayudarán a conseguir una calidad mayor para esta aplicación, se deberá tener en cuenta las subscripciones que ofrecen las tecnologías utilizadas en el desarrollo de esta aplicación.

Capítulo 5. DISEÑO Y DESARROLLO DEL VIDEOJUEGO

El diseño y desarrollo del videojuego de este proyecto está dividido en dos grandes ramas. Por una parte, se encuentra el diseño del código donde se han utilizado las tecnologías *Unity* y *YarnSpinner* como entorno de desarrollo y control de diálogo. Gran parte de esta estructura ha sido reciclada de un programa ya existente denominado *Ropework* creado y subido a *GitHub* por el usuario *Robert Yang* [20]. Por otro lado, se ha desarrollado el diseño de la historia, donde se ha trabajado con los informes investigativos mencionados en el *Capítulo 3* de esta memoria, para poder desarrollar una historia realista donde se puedan identificar las diversas señales manipulativas que utilizan los agresores sexuales en Internet.

5.1 ANÁLISIS DEL SISTEMA - UNITY Y YARNSPINNER

En el *Capítulo 2* de esta memoria, se ha descrito previamente las funcionalidades básicas que proporcionan *Unity* y *YarnSpinner*, a su vez de una breve introducción sobre el lenguaje *Yarn*. Para este trabajo, no sólo se ha utilizado la creación de clases únicas de *Unity* y el lenguaje *Yarn* mencionado para crear una novela visual con toma de decisiones, sino que además se ha diseñado funciones únicas utilizadas a lo largo del proyecto para poder obtener distintas funcionalidades establecidas en los objetivos de esta memoria, que son: conexión con la base de datos de *Firebase* y sistema de guardado en el videojuego.

Antes de explicar detalladamente las funciones añadidas en el proyecto y su funcionalidad, se procederá a analizar el código reciclado de una novela visual ya existente proporcionada por un usuario de *Github*: *Ropework*.

5.1.1 ROPEWORK

Ropework es una plantilla que proporciona un marco genérico para el diseño de una novela visual. Está construida sobre el lenguaje *YarnSpinner* y utiliza a su vez *Unity* y C# para el diseño de clases y lectura de códigos. Esta plantilla proporciona la oportunidad de utilizar guiones de *Yarn* para controlar el diseño de escena de una novela visual, controlando el

fondo de las escenas, los *sprites* de personajes y, a su vez, administrar los audios utilizados en la escena. En la *Figura 31* se puede observar la plantilla proporcionada por dicho código, dividiendo la escena en dos zonas: panel de opciones, donde se encuentran los botones de decisiones, y el panel de diálogo, donde se muestras los diálogos y los nombres de los personajes.

Figura 31. Plantilla proporcionada por Ropework

Ropework funciona de la siguiente manera: existen dos clases principales en la escena que controlan todo el flujo de funcionamiento del videojuego: la clase *RopeworkManager* y la clase *YarnSpinner*. Existe a su vez un *Canvas* denominado *RopeworkCanvas* que muestra todos los componentes que componen la escena (botones, paneles, texto, etc.). Sin embargo, los dos primeros objetos mencionados son los que controlan su funcionamiento y visibilidad dentro de la escena. Todos los componentes se encuentran dentro de una clase genérica denominada "*RopeworkDialogueUI*", como se puede observar en la ventana de jerarquía de la *Figura 32:*

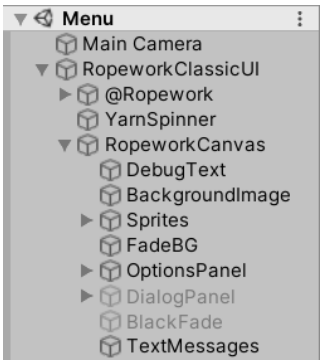

Figura 32. Venta de jerarquía del proyecto final

5.1.1.1 Objeto YarnSpinner

El objeto *YarnSpinner* tiene como componentes diversos códigos, cada uno con diversas funcionalidades.

5.1.1.1.1 Clase DialogueRunner

El primero de ellos es *DialogueRunner*, proporcionado por *YarnSpinner*, donde se realiza la lectura del archivo de guion "*.yarn.txt*". Este archivo se encuentra guardado en la variable *SourceText* del este objeto.

DialogueRunner tiene como funcionalidad clasificar e identificar las diversas nomenclaturas mencionadas en el *Capítulo 2* de esta memoria (comandos, ramas de opciones, líneas de diálogo y líneas de narración). Según las nomenclaturas encontradas, se comunicará con la clase *ClassicDialogueUI* para administrar los diálogos de texto, nombres y botones dentro de la escena.

Además, es responsable de realizar la búsqueda de funciones únicas no pertenecientes a *Yarn* que han sido específicamente diseñadas para cumplir con los objetivos establecidos en este proyecto. Todas estas funciones se definen dentro del objeto denominado *RopeworkManager*, y *DialogueRunner* accederá a ella al encontrar un comando que no reconoce.

En resumen, *DialogueRunner* es responsable de leer los archivos de lenguaje *Yarn* y de comunicarse con la clase encargada de controlar la interfaz de aplicación, *ClassicDialogueUI*, para realizar el desarrollo de la escena acorde con el guion de tipo "*.yarn.txt"*. En la *Figura 33* se puede observar en la ventana del Inspector con los parámetros que componen la clase *DialogueRunner:*

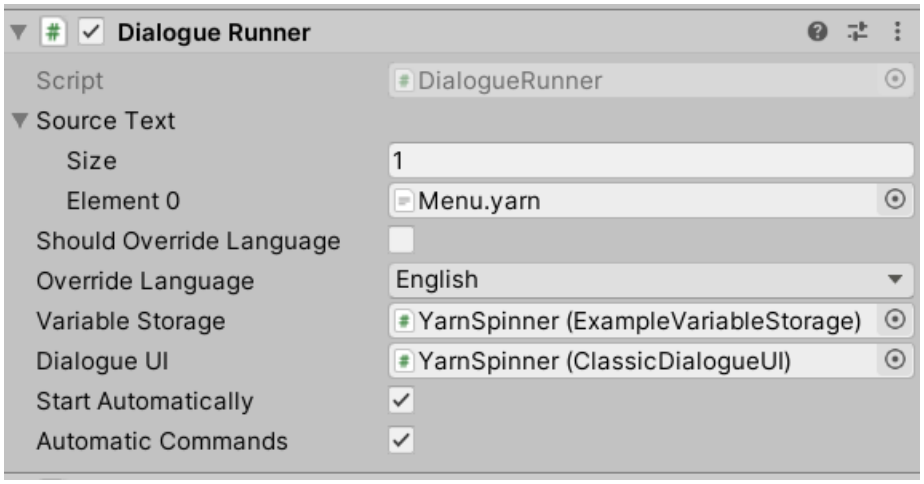

Figura 33. Ventana de Inspector con los parámetros de Dialogue Runner dentro del objeto YarnSpinner

Esta clase permite además guardar más de un archivo de tipo "*.yarn*", pudiendo de esta manera crear diferentes ramas en una misma escena utilizando la funcionalidad de nodos. Esta funcionalidad se ha utilizado en el diseño de los diversos finales de la historia, creando de esta manera diversas ramas de diálogo. En la *Figura 34* se muestra posibilidad de utilizar diversas fuentes de texto para una misma escena:

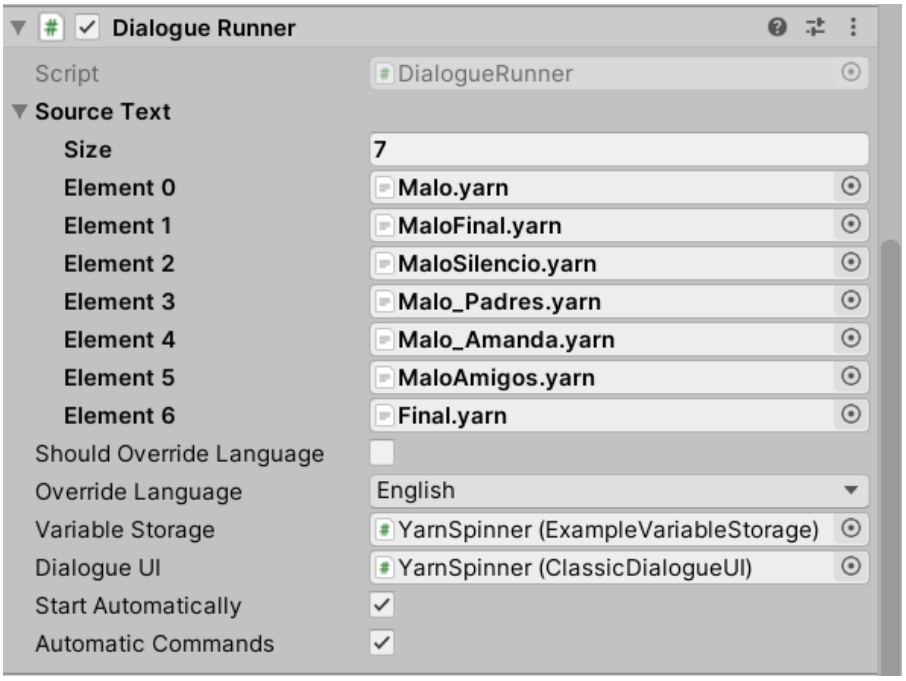

Figura 34. Ventana de Inspector - Escena Malo con diversas fuentes de código Yarn

5.1.1.1.2 Clase ClassicDialogueUI

El segundo componente perteneciente al objeto *YarnSpinner* es el denominado *ClassicDialogueUI.* Éste controla los componentes de la escena según las indicaciones obtenidas de la clase *DialogueRunner*. Además, si el código indica que se trata de una rama de decisiones, *ClassicDialogueUI* se encarga de mostrar los botones de opciones en la escena y queda a la espera a que el jugador escoja una opción. En la *Figura 35*, se puede observar en la ventana del Inspector las variables que este código controla en la escena:

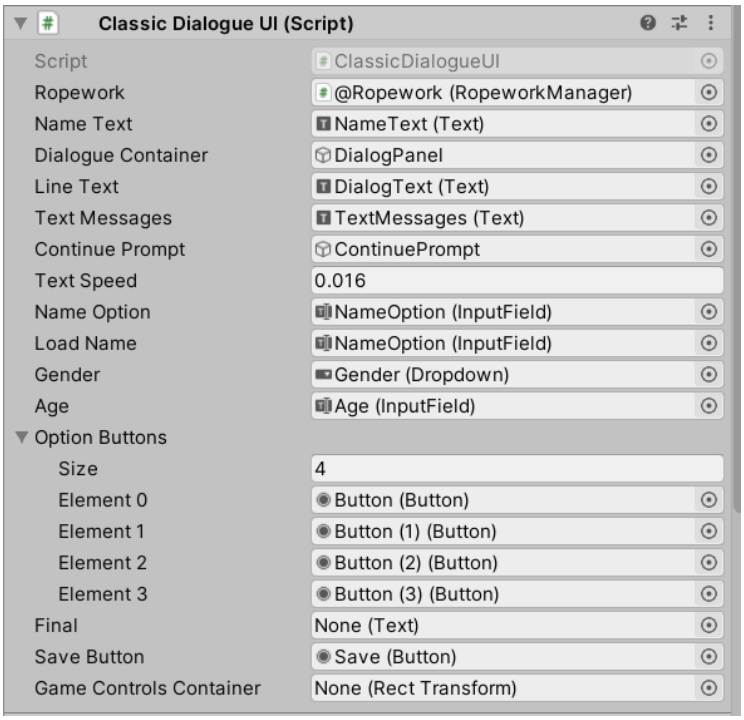

Figura 35. Ventana de Inspector perteneciente a la clase ClassicDialogueUI

Como se puede observar, éste se encarga de controlar todas las variables pertenecientes a la presentación de escena: botones, nombres de personajes, panel de diálogo, texto, velocidad de texto, entre muchos otros. Los dos botones adicionales que se pueden observar en el código, *Save Button* (botón de guardado) y *Load Button* (botón de carga), son dos funcionalidades que se han añadido en este proyecto. Dichos botones se explicarán más adelante en esta sección del código.

5.1.1.1.3 Clase ExampleVariableStorage

Por último, la clase *ExampleVariableStorage* se encarga del almacenamiento y guardado de las variables establecidas dentro del archivo "*.yarn*" y son de gran importancia para establecer las diversas ramas dentro de la historia del videojuego. Una vez más, el código *DialogueRunner* se encarga de controlar esta clase cuando detecte una línea de comando relacionada con el almacenamiento de variables. En el *Capítulo 2* de esta memoria, al realizar una breve descripción del lenguaje *Yarn*, se mostró cómo se puede realizar el guardado de variables dentro del código para poder diseñar de esta manera ramas de diálogo y poder diferenciar diferentes rutas y cambios dentro de la historia según las decisiones tomadas por el jugador.

5.1.1.2 Objeto RopeworkManager

Por un lado, hemos observado que el objeto *YarnSpinner* se encarga de la lectura del guion *Yarn* y del control de las variables del código junto con la administración de los diálogos y opciones de la historia. Sin embargo, por otro lado, el objeto *RopeworkManager* es responsable de especificar todas las funciones diseñadas en el desarrollo de este código para administrar tanto la creación de personajes, como el guardado de la historia y el control del uso de animaciones dentro del videojuego. De esta manera, cuando la clase *DialogueRunner* identifique una función no perteneciente a *YarnSpinner* se encargará de buscar dentro de la clase *RopeworkManager* para ejecutar dicha función.

A continuación, se mostrarán las funciones desarrolladas por *Ropework* que han sido recicladas y utilizadas en este proyecto, junto con una breve explicación donde se mostrará tanto su funcionalidad como la estructura que esta función debe seguir dentro del guion *Yarn*.

5.1.1.2.1 **Comando *Scene***

La funcionalidad de este comando es establecer una imagen de fondo en la escena actual. En lenguaje *Yarn* tiene el siguiente formato: <<***Scene @*** *nombreImagen*>>. Su definición dentro de la clase *RopeworkManager* es la siguiente:

```
// sets background image
[YarnCommand("Scene")]
public void DoSceneChange(string spriteName) {
```

```
        bgImage.sprite = FetchAsset<Sprite>(spriteName);
}
```

El comando "[*YarnCommand("Scene")*]" indica a *YarnSpinner* que se está definiendo un nuevo comando de lenguaje *Yarn* para que, de esta manera cuando el *DialogueRunner* obtenga como comando "*Scene*", podrá acceder a esta función y establecer la imagen de fondo para la escena. Es decir, esta línea sirve como "etiqueta" que *DialogueRunner* utilizará para localizar la nueva función.

Un ejemplo en la escena "*Day2*" dentro del archivo "*Day2.yarn.txt*" que muestra la funcionalidad de este comando es el siguiente:

```
<<Scene @ habitacion>>

<<Act @ User, user,left,center,grey>>

El sonido del despertador irrumpió en mi sueño
```

La primera línea del código muestra el uso del comando *Scene*, estableciendo la imagen "*habitación*" como fondo de la escena. El resultado final se observa a continuación en la *Figura 36*:

Figura 36. Escena Día 2 - Ejemplo función Scene

5.1.1.2.2 Comando *Act*

En el ejemplo anterior se ha podido observar el uso de otro comando único: *Act*. Este comando diseñado por *Ropework* tiene como finalidad "crear" un personaje. En otras

palabras, esta función asocia el nombre de un personaje, en este caso *User*, a una imagen, o también denominado *Sprite*, que le representa (la imagen "*user*" en este ejemplo). Además, se define la posición en la que se encuentra la imagen dentro de la escena, primero estableciendo la posición horizontal, el parámetro "*left*" marcando que el usuario se encuentra a la izquierda de la escena, y luego la vertical, colocando la imagen del personaje en el centro del eje vertical gracias al parámetro "center". Por último, se le asocia un color al personaje creado, modificando de esta manera el color del panel del nombre que se encuentra dentro del diálogo.

En resumen, la función *Act* tiene la siguiente estructura: *<<**Act @** NombreActor, ImagenNombre, posiciónX, posiciónY, colorNombre>>* y se encarga de asociar el nombre de un personaje a una imagen y color predeterminados.

Otro ejemplo donde se puede observar el resultado obtenido al utilziar este comando se en encuentra en la *Figura 37* al ejecutar el código a continuación.

```
<<Act @ Papá,papa-normal,left,center,blue>>
<<Act @ Mamá,mama-normal,right,center,green>>

Mamá: Rápido, rápido, abre ahora tu regalo.
Papá: Tu madre y yo hemos preparado algo un poco especial.
```

Figura 37. Escena Día 1 - Ejemplo de función Act

Como se puede observar en la imagen, *RopeworkManager* ha "creado" el personaje "*Mamá*", asociando la imagen "mama-normal" y color verde a dicho personaje y colocando su imagen en la parte derecha de la escena.

5.1.1.2.3 Comando *Hide*

Esta función tiene una funcionalidad contraria al comando *Act* explicado en el apartado anterior: se encarga de "destruir" un personaje, consiguiendo de esta manera que su imagen deje de aparecer en la escena e indicando al jugador que no se encuentra en el entorno actual de la historia. Esta función tiene la estructura: <<***Hide @*** *nombreActor*>> y *RopeworkManager* se encarga de destruir las imágenes guardas al nombre del actor dado, junto con su posición y color.

En la misma escena mostrada en el apartado anterior, "*Day1*", se puede tomar un ejemplo de esta función. El personaje se va a su cuarto, por lo que no sólo cambia el fondo de imagen, sino que además los *sprites* de los personajes "*Papá*" y "*Mamá*" deben desaparecer, ya que no se encuentran con el personaje principal en la nueva escena. El código utilizado es el siguiente:

```
<<Hide @ Mamá>>
<<Hide @ Papá>>

<<SceneChange @ habitacion>>
User: Ahhh....
La tarta estaba deliciosa.
```

Y en la *Figura 38* se puede observar cómo han desaparecido los dos personajes nombrados:

Figura 38. Escena Día 2 - Ejemplo de la función Hide

5.1.2 CÓDIGO PROPIO DISEÑADO

5.1.2.1 Clases

Para este proyecto, se han diseñado un cierto número de clases adicionales no sólo para poder cumplir con los objetivos planteados de este proyecto, sino además para mejorar el diseño de la aplicación y la jugabilidad del jugador.

5.1.2.1.1 Clase *FirebaseConnection*

Esta clase se encarga de realizar la conexión con la base de datos de *Firebase*, proporcionando a su vez la funcionalidad de escribir toda la información relacionada con el jugador y las decisiones que éste ha tomado a lo largo de la historia. Para poder implementar la conexión a través una aplicación de *Unity*, la propia página web de *Firebase* proporciona un tutorial donde se explican todos los pasos a seguir, además de todos los archivos y documentos necesarios en el proceso. Entre ellos se encuentra el paquete *SDK* donde están almacenadas todas las librerías de *Firebase* y los archivos de configuración correspondientes para cada dispositivo: *GoogleService-Info.plist* para IOS y *Google-servies.json* para Android [10]. Para este proyecto, se ha tenido que importar dos librerías de *Firebase: FirebaseAnalytics* y *FirebaseDatabase*.

La estructura de la clase *FirebaseConnection* es la siguiente:

```
public class FirebaseConnection : MonoBehaviour
{
    DatabaseReference root;
    public Ropework.RopeworkManager ropework;
    string nameScene;


    // Start is called before the first frame update
    void Start()
    {

        // Set up the Editor before calling into the realtime database.
        FirebaseApp.DefaultInstance.SetEditorDatabaseUrl("https://game-
54098.firebaseio.com/");
        // Get the root reference location of the database.
        root = FirebaseDatabase.DefaultInstance.RootReference;
```

```
    }
```

En este apartado del código se muestra la conexión que realiza *Unity* con la base de datos. Para ello, se utiliza el método *SetEditorDatabaseUrl*, donde se debe especificar en el parámetro de la función la dirección de la *URL* perteneciente a la base de datos creada en la página web de *Firebsase*. El parámetro *root* representa la localización de la base de datos y se utilizará en futuras funciones para realizar las escrituras necesarias en ella. Dado que la clase *RopeworkManager* utiliza *Firebase* para realizar sus conexiones, será ésta la que realizará la inicialización de esta clase y, por lo tanto, creará la conexión con la base de datos.

```
// Function that writes the information of the player in the database
    public void WriteNewUser(Player player)
    {
        if (player!=null)
        {
            string json = JsonUtility.ToJson(player);
            //string stringId = player.getId().ToString();


            DatabaseReference rootUser =
root.Child("Players").Child(player.getId()).Child(player.getName());

            // Save player info in database
            rootUser.Child("INFO").SetRawJsonValueAsync(json);

            Choices choices = player.getChoices(); //choices made by the player
in this route
            string timesPlayed = choices.getTimesPlayed().ToString();


            if(timesPlayed.Equals("0") == false)
            {
                json = JsonUtility.ToJson(choices);

rootUser.Child("TimesPlayed").Child(timesPlayed).SetRawJsonValueAsync(json);

            }

        }
```

La función *WriteNewUser(Player player)* se encarga de realizar la escritura de nueva información en la base de datos. Para ello, convertirá primero la clase *Player* (que representa los datos del jugador) en estilo *json* y procederá a guardar el resultado en la base de datos. Esta nueva información se almacenará dentro de la base de datos, bajo el nodo "*Players*",

donde se encontrará la información de todos los usuarios partícipes de esta aplicación. Además, se creará un nodo adicional denominado "*TimesPlayed*" donde se almacenarán todas las rutas realizadas por el jugador. Para evitar que la información sea eliminada por otro usuario, se ha utilizado un identificador único como padre de estas dos ramas, diferenciando de esta manera distintos dispositivos y separando los datos de cada jugador. El resultado final del guardado de la base de datos se muestra en la *Figura 39:*

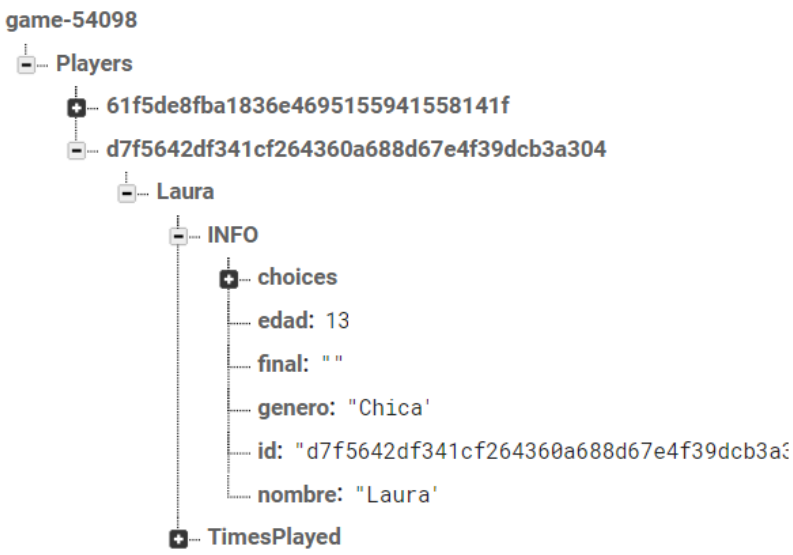

Figura 39. Resultado escritura del jugador "Laura" en la base de datos

En caso de que más de un jugador haya jugado en la aplicación en un mismo dispositivo, toda la información se guardará en un nodo nuevo correspondiente al nombre del nuevo jugador, como se puede observar en la imagen anterior.

Asimismo, si el mismo jugador ha completado más de una ruta, para no eliminar información antigua con las nuevas decisiones, bajo la rama "*TimesPlayed*" se indica el número de veces que el jugador ha completado el videojuego, además de las decisiones que ha tomado en cada una de las rutas finalizadas. Para evitar que exista una sobreescritura en la información del juego, ésta se ha almacenado en la rama "*INFO*", donde se encuentra todos los atributos almacenados en la clase "*Player*". En la *Figura 40*, mostrada a continuación, se puede observar un ejemplo del jugador "*Laura*" obteniendo 3 finales totales:

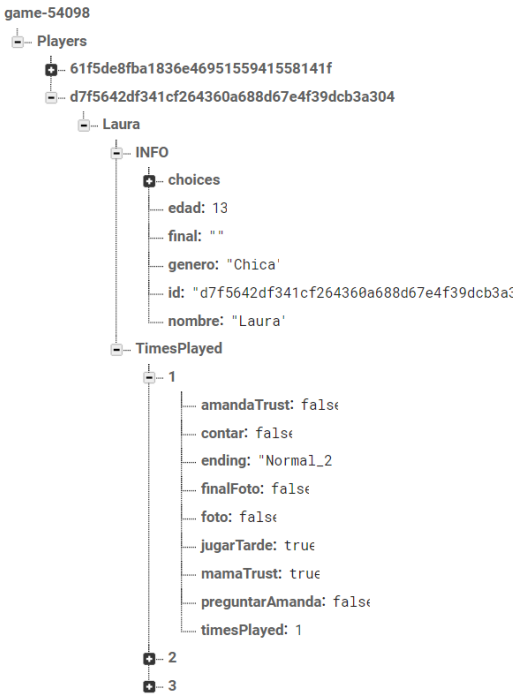

Figura 40 - Escritura rama TimesPlayed en base de datos

5.1.2.1.2 Clase *Player*

Esta clase representa toda la información relacionada con el jugador. Estos datos son introducidos por el propio jugador en el menú de inicio del juego al comenzar una nueva partida. Se puede observar dicha ventana en la *Figura 41*:

Figura 41. Sección de opciones dentro del menú de inicio del proyecto

Una vez que el jugador introduzca sus dados y seleccione el botón comenzar, se creará una nueva clase *Player* donde se almacenarán todos los datos obtenidos. Sin embargo, no sólo se guardará el nombre, género y edad del usuario, sino que, además, una vez haya finalizado la historia, se almacenará el final obtenido por el jugador. La estructura de esta clase se puede observar en el código a continuación:

```
public class Player
{
    [SerializeField] private string id;
    [SerializeField] private string nombre;
    [SerializeField] private int edad;
    [SerializeField] private string genero;
    [SerializeField] private string final;
    [SerializeField] private Dictionary<string, string> decisiones;
    [SerializeField] private Choices choices;

    public Player()
    {
        //parámetros predeterminados en caso de no estar vacíos
        decisiones = new Dictionary<string, string>();
        setName("Laura");
        setAge(13);
        setGender("Chica");
        this.choices = new Choices();
    }


    // Constructor of the class, cannot be created without all the information,
except the ending
    public Player(string name, int age, string gender)
    {
        //this.id = (int)Random.Range(00000, 99999);
        decisiones = new Dictionary<string, string>();
        setName(name);
        setAge(age);
        setGender(gender);
        this.choices = new Choices();

    }
```

Existe otro parámetro dentro de esta clase denominado "*choices*". Se trata de un diccionario donde se almacena todas las opciones tomadas por el usuario. Esta variable es necesaria ya que, en caso de que la clase *ExampleVariableStorage* borre por accidente alguna variable que represente la opción tomada de un usuario, se podrá comprobar con la clase *Player* las decisiones tomadas por el jugador y, en caso de no tener una variable que la clase *Player* sí tiene a su disposición, poder seguir manteniéndose fiel a la historia.

La clase *Player* genera un identificador, que se obtiene llamando al método *SystemInfo.deviceUniqueIdentifier*. Esta función proporcionada por *Unity* permite obtener un identificador único a cada dispositivo. En caso de ser un IOS, devuelve un *hash* de la dirección MAC del dispositivo. Para Android, éste devuelve el *md5* del ANDROID_ID. Utilizando este parámetro, se podrá almacenar con identificadores únicos en distintos nodos y evitar de esta manera sobreescrituras de usuarios dentro de la base de datos.

Las funciones y constructores dentro de esta clase son utilizadas para modificar y obtener los datos del jugador en cualquier momento de la aplicación. En otras palabras, representan "*setters*" y "*getters*", por lo que no se describirá detalladamente el código de éste.

5.1.2.1.3 Clase *Save*

La clase *Save* se encarga de realizar el guardado y carga de la partida. Éste es llamado tanto por la clase *RopeworkManager* (que realiza un autoguardado antes de pasar a la siguiente escena) como en la clase *ClassicDialogueUI*¸ donde representa la funcionalidad de los botones *Save* y *Load.*

La estructura de esta clase es la siguiente:

```
[System.Serializable]
public class Save
{
    public Player user;
    public string file;
    public string line;

    public Save()
    {

    }

    public Save(Player player, string line, string file)
    {
        this.user = player;
        this.file = file;
        this.line = line;
    }
```

Esta clase, como se ha podido observar en el código, se encargará de guardar la información del usuario, la línea en la que se encuentra en la historia y el nombre de la escena activa al

realizar el guardado de ésta. Se debe tener en consideración que todos estos parámetros son serializables (System.Serializable), es decir, permite convertir la clase actual junto con sus parámetros en formato de *bytes*, recurso necesario para realizar el guardado de éste en un archivo dentro del dispositivo.

Esta clase tiene dos funciones: una para el guardado de partida y otra para la carga de ésta. La primera se representa en el siguiente código:

```
public void SaveGame()
    {
        //Save save = CreateSaveGameObject();

        BinaryFormatter bf = new BinaryFormatter();
        string path = Application.persistentDataPath + "/SaveData/";
        Directory.CreateDirectory(path);
        FileStream file = File.Create(path + "/gamesave.save");
        bf.Serialize(file, this);
        file.Close();

        Debug.Log("GameSaved");

    }
```

Para realizar el guardado de partida, se genera un archivo denominado "*gamesave.save*" donde se almacena la clase entera de *Save* en dicho archivo. Este archivo queda guardado en un directorio proporcionado por el método *Application.persistentDataPath.* Esta función, como su nombre indica, devuelve un directorio del dispositivo donde se almacenan todos sus datos. Los archivos guardados en ese directorio no pueden ser modificados ni borrados por otras aplicaciones o entre partidas, recurso necesario para proteger los datos de guardado del jugador.

Por otro lado, la función de cargar partida es la siguiente:

```
public void LoadGame(Ropework.RopeworkManager rp)
    {
        if (File.Exists(Application.persistentDataPath +
"/SaveData/gamesave.save"))
        {
            BinaryFormatter bf = new BinaryFormatter();
            Debug.Log(Application.persistentDataPath +
"/SaveData/gamesave.save");
```

```
            FileStream file = File.Open(Application.persistentDataPath +
"/SaveData/gamesave.save", FileMode.Open);
            Save save = (Save)bf.Deserialize(file);
            file.Close();

            if (save !=null)
            {
                rp.setPlayer(save.user);
                rp.cd.setLineProgress(save.line);

                //rp.saveUser();

                rp.loadSceneName(save.file);

                Debug.Log(save.file);
                Debug.Log(save.user.getId());
                Debug.Log(save.line);
            }

        } else
        {
            rp.cd.ShowMessage("NO GAME DATA");
            rp.loadSceneName("Menu");
            Debug.Log("No game Data");
        }
```

Esta función accede al directorio establecido en el método anterior del guardado de partida y comprueba si éste es nulo o no. En el primer caso, recupera todos los datos guardados previamente en el archivo: información del usuario, línea donde había guardado la partida y escena activa cuando había realizado el guardado. Una vez obtenido la información, modifica los datos de la clase *RopeworkManager* y, una vez finalizado este proceso, carga la escena en la que se encontraba anteriormente, llamando a la función de *RopeworkManager*: *loadSeceneName(string nombre).*

Sin embargo, en caso de que el archivo esté vacío, la aplicación lanza un mensaje informado de que no existe partida guardada y vuelve al menú de inicio, utilizando la misma función de cambio de escena que en el apartado anterior.

5.1.2.1.4 Clase *LevelChanger*

Esta clase se utiliza para controlar las animaciones establecidas en la aplicación. Al realizar un cambio de escena o un cambio de imagen en el videojuego, para mejorar la fluidez de éste y evitar transiciones bruscas, se ha optado por crear una animación que simule un

difuminado de la escena. Existen dos animaciones que ayudan a este proceso, denominadas *Fade_In* y *Fade_Out*. Además, se ha utilizado una imagen denominada *BlackFade* que simplemente representa una imagen de color negro en nuestra aplicación y que permite ocultar todos los componentes en la escena. Este objeto se representa en la *Figura 42:*

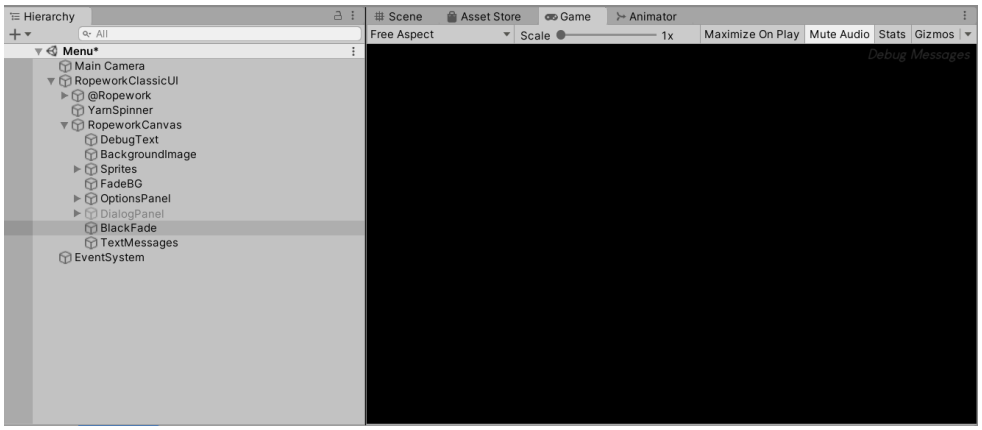

Figura 42. Representación objeto BlackFade en el proyecto

Para el diseño de las animaciones, se ha utilizado la ventana de animación de *Unity*. Para acceder a ella, se selecciona el objeto *RopeworkDialogueUI* que podemos encontrar en la ventana de jerarquía y, manteniendo a éste seleccionado, se procede a pulsar la combinación de teclas: CTRL + 6. Este comando permitirá al desarrollador acceder a la ventana de animación de *Unity*, como se puede observar en la *Figura 43:*

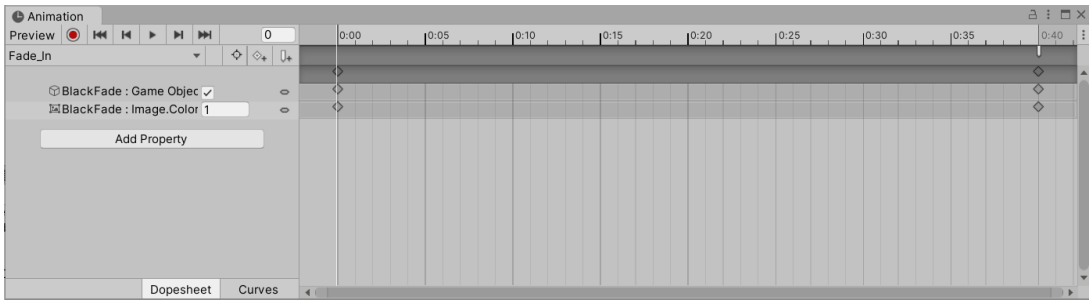

Figura 43. Ventana de animación de Unity

Para realizar el diseño de la animación de *Fade_Out* se han establecido dos estados: el primero es la aplicación en su entorno normal, mostrando sus diálogos e imágenes siguiendo el guion de *Yarn*. En el segundo, sin embargo, se activa el objeto *BlackFade*, cubriendo toda la pantalla del videojuego en una imagen negra. La animación se encargará de pasar del

primer estado al segundo en un determinado periodo de tiempo, en este caso menos de un segundo, sumergiendo de manera sutil y lenta la imagen de la aplicación en negro.

La animación de *Fade_In*, en cambio, se encargará de realizar el proceso inverso a la animación anterior: se realiza la transición con objeto *BlackFade* cubriendo toda la pantalla a obtener la imagen normal del videojuego. Esta combinación de animaciones es altamente útil para realizar el cambio de escenas, aprovechando el momento en el que el videojuego se encuentra cubierto en negro para cambiar la imagen y proceder a la siguiente escena con un nuevo fondo. Gracias a esta animación, el jugador no es consciente del cambio realizado de escenas.

Para poder controlar el cambio de imágenes y de escenas, se utilizarán dos *triggers* o disparadores para activar las animaciones y, mediante la clase *LevelChanger*, se controlará las activaciones de estos parámetros.

Existe una ventana en *Unity* denominada *Animator*, que se encuentra junto con la vista del juego y la vista de escena, que permite controlar todas las animaciones y las transiciones entre ellas, estableciendo a su vez las funcionalidades de los disparadores *FadeIn* y *FadeOut*. En la *Figura 44* se puede observar el esquema de animaciones utilizado en este proyecto:

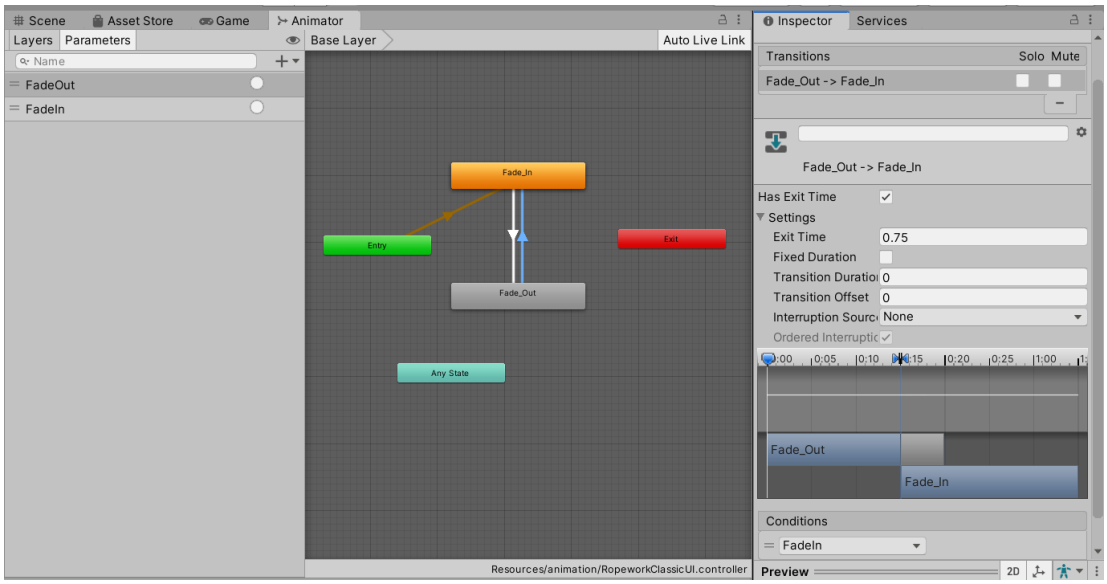

Figura 44. Esquema de flujo de animaciones del proyecto

En este flujo de pueden observar dos parámetros o *triggers* a la izquierda: *FadeOut* y *FadeIn*. Estos parámetros se encargan de controlar las transiciones entre las dos animaciones mencionadas y poder modificar de esta manera la escena de manera sutil.

En el diagrama se muestra que, nada más inicializar la escena, se activa la animación *Fade_In*. En otras palabras, la escena pasa de estar cubierta por una imagen negra a poder observar el panel del videojuego de manera normal nada más activar la escena. Sin embargo, la animación *Fade_Out* no se activará, es decir, la escena se quedará en el estado *Fade_In* hasta que el parámetro *FadeOut* sea activado. Esta función la realiza el *LevelChanger*, que controla constantemente los valores de los dos disparadores. Una vez que el disparador hacia la animación *Fade_Out* se encuentre activado, comenzará la transición de *Fade_In* a *Fade_Out*, activando la animación de este último. En este estado ocurre lo mismo que en el caso anterior; hasta que el parámetro *FadeIn* no se ha activado, no comenzará la transición desde *Fade_Out* a *Fade_In*, por lo que la clase *LevelChanger* se encargará de controlar el valor de dicho disparador para realizar la transición entre estados.

El método de la clase *LevelChanger* que controla estos parámetros de transición se denomina *FadeToLevel(string nombre, bool scena)* y tiene la siguiente estructura:

```
public void FadeToLevel(string name, bool scene)
{

    levelName = name;
    this.scene = scene;
    animator.SetTrigger("FadeOut");

    if (scene)
    {
        sprites.gameObject.SetActive(false);

        cd.dialogueContainer.gameObject.SetActive(false);
    }
}
```

Esta función, al ser llamada por la clase *RopeworkManager*, activa el parámetro *FadeOut* y, siguiendo el flujo mostrado en la figura anterior, activa la animación *Fade_Out*. Sin embargo, se debe añadir un evento dentro de la animación que realice una acción al terminar

de ejecutarse. La ventana de animación permite esta funcionalidad, obteniendo como resultado una animación mostrada en la *Figura 45:*

Figura 45. Evento FadeInComplete() en la animación Fade_In

Como se puede observar en la figura, al finalizar la animación *Fade_In*, *Unity* llama a la función *FadeInComplenete*. Este método también se encuentra definido dentro de la clase *LevelChanger:*

```
    public void FadeInComplete()
    {
        if (scene)
        {
            this.scene = false;
            levelName = "";

            cd.dialogueContainer.gameObject.SetActive(true);
        }
    }
```

En este caso, dado que la transición es hacia una nueva escena, en este método simplemente se activa el diálogo del videojuego para continuar con las líneas del guion.

Existe otro evento para *Fade_Out* y este es el que se encarga de modificar la escena una vez que la imagen se encuentre en negro. El método al que llama es *OnFadeComplete()* y tiene la siguiente estructura:

```
    public void OneFadeComplete()
    {
        if (scene) {

            rp.DoSceneChange(levelName);
            sprites.gameObject.SetActive(true);


            animator.SetTrigger("FadeIn");
        }
```

```
        else if (levelName.Equals("")) {

            Application.Quit();
        } else
        {
            SceneManager.LoadScene(levelName);
            levelName = "";
        }
    }
```

En este caso, si se trata de un cambio de escena o de imagen, *LevelChanger* se encargará de llamar a la clase *RopeworkManager* para realizar dichas modificaciones y, una vez finalizadas, activará el parámetro *FadeIn* para terminar el cambio de escena, comenzando la transición de una pantalla en negro a la escena modificada. Sin embargo, en caso de que se trate del final del juego o del cierre de este, *LevelChanger* se encargará de finalizar la aplicación en completo una vez finalizado la animación.

5.1.2.2 Funciones

En este apartado se especificarán los comandos diseñados en este proyecto que han sido implementados en los guiones de *Yarn* para mejorar la creación de una novela visual.

5.1.2.2.1 Comando *Go*

Este comando se encarga de realizar los cambios de escena, inicializando la transición entre ellas. La estructura de este método en *Yarn* es ***<<Go @ nombreEscena>>*** y el código utilizado es el siguiente:

```
        [YarnCommand("Go")]
        public void loadSceneName(string name)
        {

            if (SceneManager.GetActiveScene().name.Equals("Menu") ||
SceneManager.GetActiveScene().name.Equals("End"))
            {
                fc.WriteNewUser(player);
            }
            lc.FadeToLevel(name, false);
        }
```

Como se puede observar, esta función simplemente llama a la clase *LevelChanger* para realizar el cambio de escenas. Además, en caso de que la escena activa pertenezca a la escena *Menú* se inicializará una nueva clase *Player*, obteniendo su información de los parámetros

establecidos en el menú de opciones. Si pertenece a esta escena o a la escena final denominada "*End*", se realizará la escritura a la base de datos con la información del usuario, llamando a la clase *FirebaseConnection*.

Una vez terminado el guardado del jugador y de la partida, se procederá a llamar a la función *FadeToLevel(string nombre, boolean escena)* de *LevelChanger*, para comenzar de esta manera la transición entre escenas, utilizando las animaciones en el proceso.

5.1.2.2.2 Comando *SceneChange*

Este método, estructura similar a la función anterior *Go*, tiene como finalidad cambiar las imágenes de fondo. *RopeworkManager* proporciona una función denominada *Scene* que cumple con esta finalidad. Sin embargo, para poder utilizar las animaciones y crear una transición fluida entre modificaciones, se ha procedido a desarrollar este método donde se utilizan las funciones proporcionadas por la clase *LevelChanger*. Como se puede observar en el código, la función simplemente consiste en llamar al método *FadeToLevel (string nombre, boolean scene)* de *LevelChanger* para realizar el cambio de imágenes:

```
[YarnCommand("SceneChange")]
public void ChangeScene(string name)
{
    lc.FadeToLevel(name, true);
    cd.lineText.gameObject.SetActive(true);
}
```

5.1.2.2.3 Comando *Quit*

Este comando se utiliza en ciertos momentos de la historia para indicar que se debe cerrar la aplicación. Esto puede ocurrir al terminar la historia, donde se puede comenzar desde el principio y simplemente salir del videojuego. Su estructura es <<***Quit @*** >> y *Unity* simplemente activa la animación que difumina la escena y procede a cerrar la aplicación, utilizando una vez más la clase *LevelChanger* en el proceso.

```
[YarnCommand("Quit")]
public void Quit()
{
```

```
            lc.FadeToLevel("", false);
            //Application.Quit();
        }
```

5.1.2.2.4 Comando *Option*

En el menú de inicio, a diferencia de otras escenas del videojuego, existe un apartado que consiste en solicitar al jugador sus datos personales. Por lo tanto, una vez llamado está función en el guion de *Yarn*, *RopeworkManager* procederá a llamar a *ClassicDialogueUI* para indicarle que debe mostrar el panel de opciones con datos a introducir al jugador. Este comando no necesita ningún parámetro, por lo que su estructura es: **<<*Option* @ >>.**

```
        [YarnCommand("Option")]
        public void optionMenu()
        {
            Image img = GameObject.Find("OptionsPanel").GetComponent<Image>();
            Color color = new Color(19f/255f, 19f / 255f, 19f / 255f);
            color.a = 0.65f;
            img.color = color;

            cd.SetOptionMenu();
        }
```

Además, para poder diferenciar el panel con el fondo de la escena, se modifica el color del objeto *OptionPanel*, panel que controla la representación de opciones al jugador, asignándole un tono más oscuro con respecto el fondo de la pantalla. El panel resultante en el menú de inicio se muestra en la *Figura 46:*

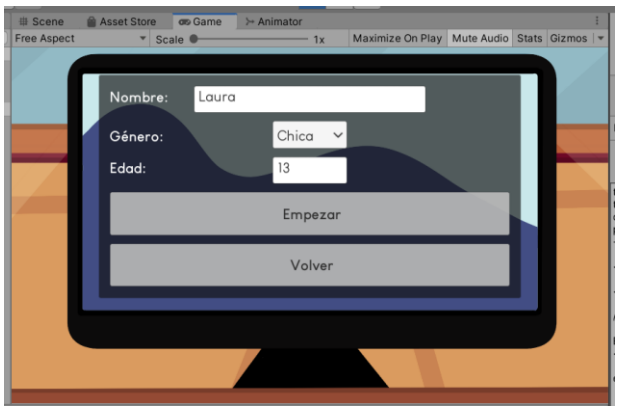

Figura 46. Panel de opciones establecido en el menú de inicio mediante el método Option

5.1.2.2.5 Comando *HideDialog*

Este comando también es utilizado en el menú de inicio y tiene la finalidad ocultar el panel de diálogo de la escena, pudiendo mostrar en cambio sólo el cuadro de opciones del jugador. El resultado se puede observar en la figura anterior, *Figura 46*, ya que en la pantalla sólo se puede observar el cuadro de opciones, mientras que el diálogo se mantiene en oculto. El código utilizado para este método es el siguiente:

```
[YarnCommand("HideDialog")]
public void hide()
{
    GameObject.Find("DialogPanel").SetActive(false);
}
```

Al igual que el método anterior, *HideDialog* no necesita ningún parámetro por lo que su estructura es: <<***HideDialog @*** >>

5.1.2.2.6 Comando *HideOption*

Esta función sirve para ocultar el panel de opciones establecido previamente con el comando *Option*. Este método simplemente es una forma de asegurar que el panel establecido sea eliminado completamente y no aparezca en futuras escenas, por lo que es una medida de prevención contra posibles fallos. El código utilizado es el siguiente:

```
[YarnCommand("HideOption")]
public void hideOption()
{
    GameObject.Find("OptionsPanel").SetActive(false);
}
```

Como se puede observar, este método no requiere ningún parámetro, por lo que el método de llamada utilizado en *Yarn* es el siguiente: <<***HideOption @*** >>

5.1.2.2.7 Comando *SetEnding*

Este método se utiliza para establecer el final obtenido por el usuario y guardarlo en la clase *Player*. Dado que este final depende de la rama en la que se encuentre el jugador, a consecuencia de las decisiones tomadas, se necesita un parámetro de tipo *string* donde se introduzca el nombre del final concluyente. Por lo tanto, la estructura de este comando será

el siguiente: <<***SetEnding @*** *nombreFinal*>>**.** El código desarrollado para esta función es el siguiente:

```
[YarnCommand("SetEnding")]
        public void SetLoadScene(string ending)
        {
            try
            {
                player.setEnding(ending);
                fc.WriteNewUser(player);
            }
            catch (NullReferenceException e)
            {
                Debug.Log("Player is empty");
                Debug.Log(player.getId() + ", " + player.getName() + ", " +
player.getAge() + ", " + player.getGender());

            }
        }
```

Como se puede observar en el código, *RopeworkManager* llama a la clase *Player* para modificar el parámetro correspondiente con el final. Una vez que este cambio ha sido finalizado, *RopeworkManager* procede a conectarse a la base de datos y realizar la escritura del jugador en ella, pudiendo observar ahora que el campo "ending" no se encuentra vacío y que se han almacenado además las decisiones tomadas por el jugador en la rama "*TimesPlayed*".

5.1.2.2.8 Comando *ShowFinal*

Este comando es utilizado al final del videojuego, cuando el jugador ha terminado la historia y ha llegado a un final determinado. En la escena final denominada *End*, se le presentará dos opciones al jugador: Comenzar la historia desde cero o cerrar partida y terminar con la aplicación. Encima de estas dos opciones se puede observar un cuadro de texto, indicando el final obtenido por el jugado. Este texto es llamado por esta función, *ShowFinal*, donde se obtiene el final guardado dentro del jugador, paso realizado previamente utilizando el método *SetEnding*, y se muestra en pantalla. Dado que, una vez más, este método no requiere ningún parámetro, la estructura tomada en *Yarn* es la siguiente: <<***ShowFinal @*** >>. Como se puede observar en el código, esta función llama a *ClassicDialogueUI*, quien controla el cuadro de texto donde se muestra el final:

```
[YarnCommand("ShowFinal")]
public void ShowFinal()
{
    cd.showFinal(player.getEnding());
}
```

Se puede observar el resultado final en la *Figura 47:*

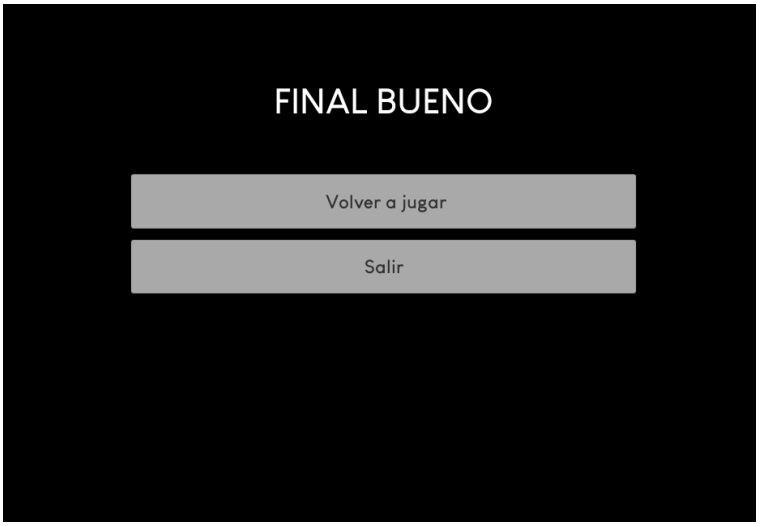

Figura 47. Escena final, mostrando ejemplo del método ShowFinal y el final obtenido

5.1.2.3 Modificaciones del código Ropework

En apartados anteriores se ha explicado el código utilizado de *YarnSpinner* y *Ropework*, además de las funciones adicionales diseñadas para cumplir con los objetivos establecidos de este proyecto, junto con otros elementos que mejoran la jugabilidad de la aplicación. En este apartado, sin embargo, se mostrarán las modificaciones que se han realizado dentro del código reciclando, justificando sus cambios y mostrando los resultados obtenidos.

5.1.2.3.1 Menú de inicio

Como se ha podido observar en apartados anteriores, se ha diseñado una función de *Yarn* que tiene como finalidad mostrar un cuadro de opciones que el jugador debe rellenar para introducir su información. La clase *ClassicDialogueUI* debe de establecer, sin embargo, unas funciones adicionales que modifiquen los parámetros del jugador cuando detecté que ha habido una modificación dentro de los campos de opciones.

Si tomamos como ejemplo el campo de la edad, que se trata de un campo de texto que sólo permite introducir números, éste está establecido de tal manera que en el momento que se haya terminado de modificar el campo (cuando el usuario pulsa *Enter* o pulsa cualquier zona fuera del campo de edad), llamará a la función *Age_Changed(string newAge)* definido esta clase, introduciendo en el parámetro *newAge* el nuevo valor y guardando éste dentro de la clase *Payer*. Si observamos en la Ventana del Inspector los parámetros del objeto *Age* (correspondiente con el campo de texto donde se introduce el nombre), se puede observar la opción de introducir un método dentro del cuadro *On End Edit (String)*. En otras palabras, esta opción llama al método *AgeChanged(string newAge)* cuando detecta que el usuario ha terminado de modificar el campo.

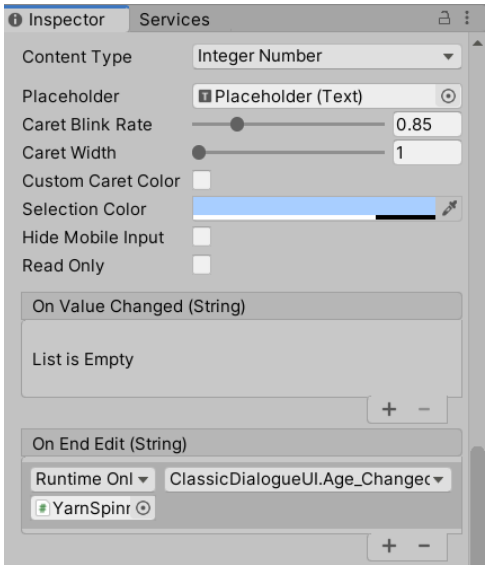

Figura 48. Opción del campo de texto Age donde se llama al método Age_Changed(string)

El código de dicho método es el siguiente:

```
public void Age_Changed(string newAge)
{
    int userAge;

    if (newAge == null || newAge.Equals(""))
    {
        userAge = 13;
    } else
    {
        userAge = int.Parse(newAge);
    }
```

```
            Debug.Log(newAge);

            ropework.getPlayer().setAge(userAge);
        }
```

En el caso de introducir el nombre, utilizando el campo de texto *NameOption*, el proceso es el mismo: se modifica la opción *On End Edit(string)*, introduciendo el método de *ClassicDialogueUI New_Name(string)* para indicar a *Unity* que debe llamar a dicho método cuando detecte que el usuario ha terminado de modificar el campo. El código de este método consiste en guardar el nuevo valor del nombre del jugador en la clase *Player* y se muestra a continuación:

```
        public void New_Name(string newText)
        {
            string userName;

            if (newText == null ||newText.Equals(""))
            {
                userName = "Laura";
            } else
            {
                userName = newText;
            }

            Debug.Log(userName);

            ropework.getPlayer().setName(userName);
        }
```

Por último, el parámetro perteneciente al género del jugador se trata de un *Dropdown* (despliegue en español), donde sólo se presentan dos únicas opciones: Chico o Chica. En este caso, las opciones escogidas son un *integer* de valores 0 o 1, según su posición de la lista. Cuando el campo es modificado, *Unity* llamará a la función *New_Gender(int)* de *ClassicDialogueUI* donde convertirá el valor numérico en el género del jugador, éste siendo Chico o chica. Se puede observar el código explicado a continuación:

```
public void New_Gender(int newGender)
        {
            string gender;

             if (newGender == 1)
            {
                gender = "Chico" ;
            } else
            {
```

```
                gender = "Chica";

            }

            Debug.Log(gender);

            ropework.getPlayer().setGender(gender);
        }
```

Estas funciones determinan la funcionalidad establecida dentro de los campos. Sin embargo, es necesario establecer dos funciones adicionales que permitan mostrar estos parámetros en pantalla; *SetOptionMenu()* y *DeleteOptionMenu()*. Estos dos métodos simplemente activan y desactivan los campos de opciones y son llamados por las funciones de *Yarn* establecidas en *RopeworkManager: Option* y *HideOption*. El código de ambas funciones es el siguiente:

```
        // set as active all the parameter's found in the Option's panel .
        // Shows: NameOption Input field, Age Input Field and Gender dropwdown
        public void SetOptionMenu()
        {
            nameOption.gameObject.SetActive(true);
            age.gameObject.SetActive(true);
            gender.gameObject.SetActive(true);
        }

        // Hides all the options inside the Option Menu Panel
        // Hides: NameOption Input field, Age Input Field and Gender dropwdown
        public void DeleteOptionMenu()
        {
            nameOption.gameObject.SetActive(false);
            age.gameObject.SetActive(false);
            gender.gameObject.SetActive(false);
        }
```

5.1.2.3.2 Funciones Guardar/Cargar partida

En esta aplicación, se han añadido dos botones adicionales que se pueden observar en todas las escenas del juego: *Guardar* y *Cargar* partida. Estos dos botones, dado que la clase *ClassicDialogueUI* controla todos los elementos de una escena, necesitan unos métodos a los que llamar cuando éstos son pulsados, por lo que se ha diseñado dos funciones adiciones que interaccionan con la clase *Save* para guardar y cargar la información del usuario, la línea en la que se encuentra en la historia y en la escena en la que se encontraba al guardar partida.

La función *SaveGame()* obtiene los parámetros a enviar a la clase *Save* gracias a los objetos *DialogueRunner* y a *RopeworkManager*. Para obtener el nombre de escena, simplemente utiliza una función de *Unity* para obtener el nombre de la escena que se encuentra activa en estos momentos: *GetActiveScene()*. Sin embargo, debe recurrir a las clases *DialogueRunner* para obtener la línea en la que se encuentra en el guion *Yarn* al guardar la escena y la clase *RopeworkManager* para obtener la información del jugador.

La función de guardar partida es la siguiente:

```
        public void SaveGame()
        {
            Debug.Log("Saving...");
            this.ShowMessage("Saving...");
            Save save = new Save(this.ropework.getPlayer(), lineProgress,
SceneManager.GetActiveScene().name.ToString());

            save.SaveGame();

            this.ShowMessage("Game Saved");
        }
```

En caso de cargar partida, *ClassicDialogueUI* simplemente llama a la función *LoadGame()* perteneciente a la clase *Save* para cargar la escena. Se puede observar su código a continuación:

```
        public void LoadGame()
        {
            Save save = new Save();
            save.LoadGame(ropework);

            //this.ShowMessage("Game Loaded");

        }
```

5.1.2.3.3 Implementación nombre/género del jugador

Esta modificación del código *ClassicDialogueUI* se ha utilizado para mantener una linealidad dentro de la historia del videojuego. Cuando el jugador establece su nombre dentro de la aplicación, cada vez que el protagonista hable, el panel de nombre debe mostrar el nombre del jugador. Además, los personajes secundarios de la historia se referirán al

protagonista en algún momento y, para mantener la familiaridad de la conversación, se referirán al protagonista por su nombre.

Por ese motivo, se ha modificado la función existente *RunLine (Yarn.Line)* de *ClassicDialogueUI*. Esta función es utilizada por *DialogueRunner* y define el proceso que el código sigue para mostrar tanto el nombre del personaje que está hablando como la línea de diálogo a la que se está refiriendo. El código está diseñado de tal manera que, en el momento que *YarnSpinner* detecte el símbolo ":" dentro de una línea, significa que un personaje está hablando. La zona previa a los dos puntos corresponde al nombre del personaje mientras que la zona secundaria es la frase de diálogo que está diciendo. Una pequeña modificación realizada en esta parte del código es que, cuando se detecté que el nombre del personaje es "*User*", se modificará dicho parámetro por el nombre guardado en la clase *Player*. De esta manera, *YarnSpinner* en vez de mostrar el campo "*User*" dentro del panel de nombre, mostrará el nombre del jugador.

```
        public override IEnumerator RunLine (Yarn.Line line)
        {
            lineProgress = line.text;
            // Show the text
            lineText.gameObject.SetActive (true);

            // ROPEWORK SPECIFIC STUFF
            string speakerName = "";
            string lineTextDisplay = line.text;
            if ( line.text.Contains(":") ) { // if there's a ":" separator, then
identify the first part as a speaker
                var splitLine = line.text.Split( new char[] {':'}, 2); // but
only split once
                speakerName = splitLine[0].Trim();
                if(speakerName.Equals("User"))
                {
                    speakerName = ropework.getPlayer().getName();
                }

                lineTextDisplay = splitLine[1].Trim();
                // …
                 // …
```

Se puede observar a continuación la comparación en lo que indica el guion de *Yarn* con el resultado obtenido en la *Figura 49:*

```
User: ¡Mi nuevo cuarto es enorme!
```

Figura 49. Ejemplo implementación nombre de usuario en panel de nombre

Otra modificación realizada en este clase se encuentra una vez más en esta misma función *RunLine(Yarn.Line)* de *ClassicDialogueUI*. En este caso, el código Comprueba si la línea de diálogo contiene el *string* "[name]". En caso de que esta afirmación sea cierta, la función se encargará de sustituir dicha cadena de texto por el nombre del usuario. De esta manera, cada vez que un personaje terciario desee llamar al protagonista, se introducirá "[name]" en el guión *Yarn* y *ClassicDialogueUI* se encargará de modificarlo.

```
                // …
                // …
                if (lineTextDisplay.Contains("[name]"))
                {
                    lineTextDisplay = lineTextDisplay.Replace("[name]",
ropework.getPlayer().getName());
                } else if (lineTextDisplay.Contains("[NAME]"))
                {
                    lineTextDisplay = lineTextDisplay.Replace("[NAME]",
ropework.getPlayer().getName().ToUpper());
                }
            }
```

Un ejemplo mostrando la funcionalidad de esta modificación, se muestra a continuación en la *Figura 50*, mostrando primero la línea de *Yarn* que establece dicho diálogo:

```
Mamá: Me alegro mucho, [name].
```

Figura 50. Ejemplo implementación [name] dentro del videojuego

Finalmente, esta última modificación se debe a la utilización del género en ciertos adjetivos y sustantivos utilizados en el español. Como en este videojuego se desea implementar el nombre del jugador y su género, ciertas descripciones deben ser modificadas según el género establecido. Por ese motivo, siguiendo un proceso similar a la modificación anterior, se ha utilizado el símbolo "@" en sustitución a la letra "a" u "o" al describir el género del protagonista en ciertos adjetivos. El código, una vez más dentro de la función *RunLine(Yarn.Line)* de *ClassicDialogueUI*, comprueba si la línea de diálogo o de narración contiene el símbolo "@". En caso afirmativo, procederá a obtener el género del jugador guardado dentro de la clase *Player*. En caso de que éste sea masculino, se cambiará el símbolo "@" por la letra "o". En caso de que sea femenino, esta modificación se realizará con la letra "a". La funcionalidad explicada ase muestra en el código a continuación:

```
            if (lineTextDisplay.Contains("@"))
            {
                //lineTextDisplay.Split(new char[] { '@'});
                string genero = ropework.getPlayer().getGender();
                //Debug.Log(genero);
                //Debug.Log(genero);

                switch (genero)
                {
                    case "Chico":
                        lineTextDisplay = lineTextDisplay.Replace('@', 'o');
                        break;
                    default:
                        lineTextDisplay = lineTextDisplay.Replace('@', 'a');
```

```
                        //Debug.Log(genero);
                        break;
                    }
                }
                 //CONTINUACIÓN CÓDIGO ROPEWORK
                 //…
        }
```

Se mostrará a continuación un ejemplo donde se implemente esta modificación en el proyecto final, representando el resultado final en la *Figura 51*, mostrando primero la línea correspondiente en el guion de *Yarn:*

```
No sabía por qué de repente Hally empezó a hablar sobre este tema, pero las
preguntas se estaban volviendo un poco personales y no estaba segur@ de si me
sentía del todo cómod@...
```

Figura 51. Ejemplo implementación "@" en el proyecto

5.1.3 DIAGRAMA DE FLUJO

Se puede observar la interacción entre clases y objetos creados en *Unity* que definen el funcionamiento del videojuego diseñado en el siguiente diagrama de flujo mostrado en la *Figura 52*.

En este diagrama se puede observar el flujo que sigue la aplicación al realizar la lectura del guion *Yarn* perteneciente a la escena activa. En resumen, la clase *DialogueRunner* se encarga de analizar las líneas encontradas en el archivo *Yarn*, identificando y clasificando cada estilo de línea. En el proceso, se comunica con la clase *ClassicDialogueUI* para modificar los componentes pertenecientes a la escena, mostrando las líneas de diálogos, botones e

imágenes según las instrucciones recibidas de la clase *DialogueRunner*. Sin embargo, en caso de que se trate de una función única no perteneciente a *YarnSpinner*, la clase *DialogRunner* se comunicará esta vez con el objeto *RopeworkManager*, donde se encuentra almacenadas todas las funciones diseñadas para esta aplicación.

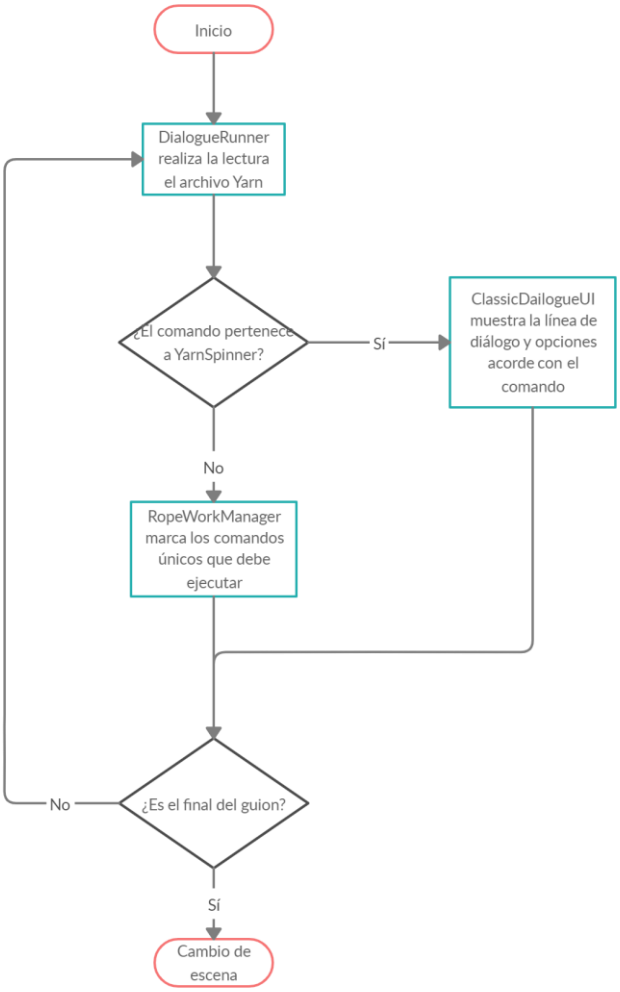

Figura 52. Diagrama de flujo utilizado en este proyecto

5.1.4 DIAGRAMA DE BLOQUES

Una vez visto el flujo que sigue esta aplicación, se mostrará a continuación en la *Figura 53*, el diagrama de bloques de todos los objetos encontrados en este proyecto. A pesar de que cada una de sus funcionalidades se han explicado a lo largo de este capítulo, este diagrama

ayudará a visualizar más fácilmente la relación que existe entre ellas y las funcionalidades básicas que establecen en cada una de ellas.

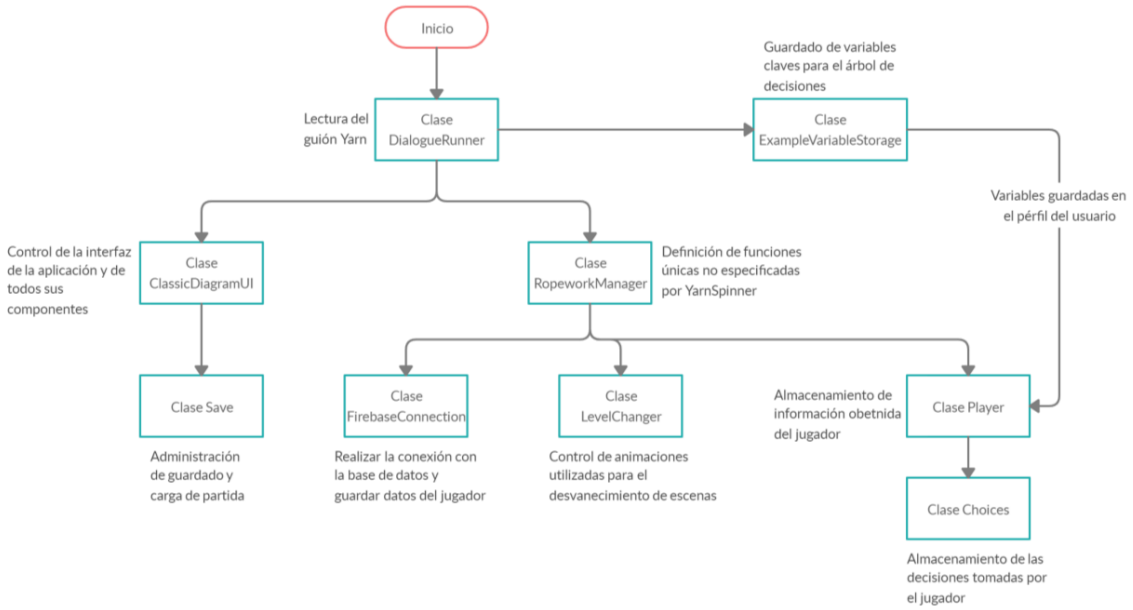

Figura 53. Diagrama de bloques

5.2 DISEÑO DE LA HISTORIA - ÁRBOL DE DECISIONES

La historia de este videojuego se ha diseñado estableciendo como base los informes investigativos de *Online Grooming*, todos ellos mencionados en el *Capítulo 2* de esta memoria. Se han utilizado elementos característicos obtenidos de dichos informes tanto en el diseño del perfil de la víctima como la del propio acosador, con el objetivo de crear un entorno realista donde se muestren señales claras de manipulación. A lo largo de la historia, el jugador deberá identificar dichas manipulaciones y, de esta manera, aplicar el mismo comportamiento en la vida real para protegerse de depredadores sexuales.

La historia gira en torno a un adolescente que pasa mucho tiempo sólo en casa. Por su cumpleaños, sus padres le regalan un portátil nuevo y comienza a jugar videojuegos online por primera vez. En uno de ellos conoce a “Hally”, otro jugador muy sociable y extrovertido que le ayuda tanto dentro del juego como en la vida real, animándole cuando se siente sólo y ayudándole a tener más confianza en sí mismo. Sin embargo, a medida que van pasando

los días, la actitud de "Hally" va cambiando, poco a poco introduciendo temas sexuales en las conversaciones que incomodan al protagonista. Además, mientras su amistad con "Hally" aumenta junto con las preguntas sexuales que éste realiza al protagonista, se le presenta la posibilidad al jugador de mantener la relación de cercanía con sus padres, que apenas están en casa, y con su compañera Amanda, la única persona que debe considerar como amiga en toda su clase.

La historia de este videojuego se desarrolla a lo largo de 5 días, donde cada uno de ellos representa una fase distinta en el proceso de manipulación de un acosador online, procedimiento obtenido del informe de *Online Grooming* de Patricia Alonso [18]. Las decisiones que tome el jugador afectarán su vida personal en distintos aspectos, ya sea su relación con sus padres, su amistad con Amanda y el resto de clase, o la misma confianza que el protagonista tiene hacia "Hally".

5.2.1 DÍA 1 – INTRODUCCIÓN

El primer día de la historia sirve como un proceso introductorio tanto del protagonista como del entorno que le rodea. La historia comienza con el cumpleaños del jugador, donde sus padres se disculpan por apenas estar en casa y la dan un portátil como regalo de cumpleaños.

A lo largo del día, mientras el protagonista va probando su nuevo ordenador y se va descargando múltiples aplicaciones, se presenta la situación en la que se encuentra el jugador: los padres de éste siempre están viajando por temas de trabajo y, por ese motivo, se tiene que mudar múltiples veces de casa a lo largo del año. A consecuencia de esto, el protagonista está constantemente trasladándose de colegio, repitiendo una y otra vez el proceso de comenzar un grupo de amigos. Esta vez, en cambio, prefiere no mantener relación alguna con nadie de clase, evitando de esta manera tener que despedirse de alguien cercano.

La soledad que siente el protagonista en la escuela al no pertenecer a ningún grupo ni tener a nadie en confianza en clase aumenta al permanecer constantemente sólo en cada, dado que sus padres están constantemente ocupados debido a sus respectivos trabajos.

Se ha diseñado este entorno con el objetivo de implementar características encontradas en las víctimas de *Online Grooming*, información obtenida de uno de los trabajos investigativos previamente estudiados. El informe de *Violencia Viral* de *Save the Children* [17] establece que la soledad, la baja autoestima y falta de comunicación con los familiares son puntos vulnerables y característicos que pueden marcar una víctima de *Online Grooming*. Entre ellos también se encuentra un uso intensivo de Internet y tener un dispositivo electrónico propio. Todos estos aspectos han sido fundamentales en el proceso de desarrollo de esta historia.

En esta parte de la historia, a pesar de que en ella se presenten dos opciones de diálogo, éstos no serán mencionados. Esto se debe a que estas opciones simplemente sirven como método introductorio al jugador de las mecánicas que se utilizan en esta aplicación, con el objetivo de que se familiarice con ellas. No tienen ninguna repercusión en la historia y simplemente modifican ligeramente el diálogo de ésta.

5.2.2 DÍA 2 – AMISTAD

El segundo día de la historia comienza con el protagonista yendo al colegio. Allí, se presenta la única amiga que el jugador tiene en clase llamada Amanda. La relación entre estos dos personajes existe gracias a la insistencia de Amanda, ésta queriendo ser amiga del protagonista y no queriendo que esté sólo en clase. Sin embargo, debido a que el jugador se ha cambiado múltiples veces de colegio y no quiere tener relación con nadie por miedo de rechazo o por el dolor de despedida al mudarse, el protagonista no hace ningún amago de mantener su amistad, por lo que Amanda siempre está proponiendo planes al protagonista que éste, casi siempre, rechaza. Amanda, intentando una vez más conectar con el protagonista, propone quedar en el juego que el protagonista probó el día anterior y echar juntos una partida online. Esta vez, el protagonista acepta, bastante entusiasmado de quedar de esa forma con su compañera.

Al terminar las clases, el protagonista vuelve a su casa y se encuentra que ésta está vacía una vez más, con sus padres fuera trabajando. Sintiéndose sólo, el protagonista procede a jugar con su ordenador. Al rato se da cuenta de que se acerca la hora con la que ha quedado con

Amanda, por lo que le escribe un mensaje. Al rato, Amada ve el mensaje y lo ignora. El protagonista comienza a sentirse inseguro y su estado de ánimo va empeorando. Intentando distraerse, continúa jugando y en el juego se encuentra con otro usuario llamado "Hally" y termina jugando toda la tarde con él. Gracias a este usuario, el protagonista deja de lado su soledad y olvida completamente el mensaje ignorado por Amanda. El jugador se va a dormir esa noche, teniendo mucha curiosidad por el usuario "Hally" y queriendo volver a jugar con él.

5.2.3 DÍA 3 – RELACIÓN Y EXCLUSIVIDAD

Al tercer día, el protagonista vuelve al colegio y se encuentra con Amanda. En esta situación, se le presenta al jugador dos opciones: preguntar a Amanda sobre por qué ignoró su mensaje o no evitar el tema. Esta decisión no tendrá grandes consecuencias en la historia, pero si modificará ciertos diálogos y el estado de humor del personaje en ciertos momentos. Al volver a casa, el protagonista queda con el usuario que conoció el día anterior, Hally, y, una vez más, juega toda la tarde con él.

Cuando llega la noche, "Hally" le dice que tiene que irse a cenar y es entonces que el personaje se da cuenta que sus padres no han vuelto y que él todavía no ha cenado tampoco. "Hally" insiste al jugador que también tiene que cenar y, sólo si éste le manda una foto con su cena vacía, podrán seguir jugando. El protagonista, cocinando algo rápido, termina su cena y espera a que "Hally" se conecté. Cuando lo hace, éste insiste que le mande una foto de su cena y el jugador, sintiéndose alagado de que "Hally" se preocupe por él, le manda una foto de su cara parcialmente tapada por el plato vacío de su cena. Al ver esa imagen, "Hally" comienza a insistir al jugador que le mande una foto de su cara.

Este es uno de los primeros momentos críticos de la historia, ya que, en este momento, el depredador sexual comienza a manipular al protagonista, utilizando diversas tácticas para poder ver el rostro de su víctima y establecer si entra dentro de sus estándares. Entre esas tácticas, envía una foto de un joven adolescente, haciéndose pasar por él para que el protagonista piense que ambos de la misma edad y se sienta más seguro hacía él. Además, si el personaje se niega a enviar la foto, el depredador seguirá insistiendo, volviendo a

aparecer la opción de enviar o no la foto. Esta es otra de las características de los depredadores: a pesar de que los usuarios se nieguen a una petición suya, estos utilizan diversos métodos de engaño y manipulación para conseguirlo, insistiendo constantemente hasta cumplir con su objetivo. Finalmente, la decisión de este momento se guardará en la variable "***$foto***" y tendrá repercusiones en el final de la historia.

Después de esta conversación, sin importar la decisión que haya tomado el jugador, el protagonista desvelará al depredador que ha tenido un mal día. Éste preguntará sobre el tema, intentando obtener de esta manera información privada del usuario. Una vez más, el jugador se puede negar, no queriendo compartir información privada con un usuario online. Si decide no contarlo, una vez más, el depredador insistirá, intentando convencer al jugador de contarle sus secretos y vida privada. Esta decisión también tendrá importancia en el futuro y se guardará en la variable "***$contar***".

Al final del día, el jugador seguirá jugando con "Hally". En este momento el depredador comentará lo atractivo que sale el jugador en la imagen (el diálogo cambiará si se le ha enviado foto del rosto del protagonista o no, refiriéndose entonces a la imagen de la cena que había enviado al principio). En ese momento le preguntará al protagonista si tiene pareja y si ha besado alguna vez a alguien, introduciendo de esa manera un tema sexual en la conversación de manera leve, evitando que el usuario se alarme al comienzo de la relación. Continuará explicando las fantasías que éste tiene y le preguntará al protagonista sobre las suyas, convirtiendo poco a poco la conversación en un tema exclusivamente sexual.

A continuación, se muestra una tabla con los valores de las variables presentadas a lo largo de este día según la decisión que haya tomado el jugador y que, además, determinarán los finales obtenidos por éste.

Variables	***Decisiones/ Valores***	
	Acceder	**No Acceder**
$foto	True	False
$contar	True	False

Tabla 1. Valores de decisiones en el Día 3 - Enviar foto y Contar información privada

5.2.4 DÍA 4 – EVALUACIÓN

En el cuarto día, el protagonista se despierta cansado y con dolor de cabeza, replanteándose saltarse colegio. Cuando decide volver a irse a dormir, suena el timbre de su casa y se encuentra a Amanda en el portal. Le dice que su madre le había pedido que acompañase al protagonista al colegio, por lo que deciden ir juntos a la escuela. Cuando terminan las clases, Amanda se acerca al protagonista y le pregunta si después de clase pueden quedar juntos a dar una vuelta. En este momento el personaje tiene la posibilidad de aceptar la oferta a Amanda o volverse a casa. El resultado de esta decisión se almacenará en la variable **$amandaTrust.** En este momento ocurren dos grandes cambios de diálogo. Si el personaje decide irse con Amanda, ambos se disculpan sobre su comportamiento y el personaje se da cuenta de la amistad que tiene con Amanda, confiando de esta manera en ella. En caso contrario, se vuelve directamente a casa y el protagonista se alejará aún más de ella.

Cuando vuelve a casa, se encuentra con su madre en el salón. Ella le ofrece al protagonista si le apetece ver una película juntos y pasar cierto tiempo en familia. Si el jugador se niega, va directamente a su cuarto. En caso contrario, se queda viendo una película con su madre y se vuelven más cercanos. Al igual que en el caso con Amanda, está solución tendrá repercusiones futuras ya que el protagonista sentirá que puede confiar en sus padres si ha decidido acceder a su oferta. En caso contrario, el protagonista sentirá que no es cercano a sus padres y no confiará en ellos. En cualquier caso, el resultado de esta decisión se almacena en la variable **$mamaTrust.**

Al entrar en su cuarto, el protagonista se conecta al videojuego online y continúa jugando con “Hally” una vez más. Dependiendo de las decisiones tomadas a lo largo de este día saltarán distintos diálogos con el usuario. En la conversación, “Hally” le pregunta al protagonista si utiliza su ordenador propio para jugar con él y si juega sólo en su habitación. Esto define uno de los métodos de los acosadores sexuales, evaluando el entorno en el que se encuentra sus víctimas antes de comenzar conversaciones sexuales más arriesgadas y evitar de esta manera que sea descubierto por sus familiares. Cuando descubre que en efecto el protagonista tiene su propio portátil y juega sólo en su cuarto, “Hally” vuelve una vez más

a sacar temas sexuales en la conversación, en este caso mencionando directamente actos sexuales y preguntando al protagonista si ha visto alguna vez vídeos pornográficos. Al final, el usuario le envía un vídeo sexual al protagonista, haciéndole sentir muy incómodo, pero sin llegar a romper la confianza que éste siente en él. Antes de que pueda continuar la conversación, y que el usuario pueda manipular todavía más al protagonista a que vean dicho video juntos, la madre del protagonista toca la puerta, exigiendo al jugador que se vaya a dormir. Éste obedece a su madre y se despide con su amigo online, sintiéndose muy inseguro e incómodo por lo que acaba de ocurrir en la conversación.

Las decisiones tomadas en este día se pueden observar en la *Tabla 2* que se muestra a continuación:

Variables	***Decisiones/ Valores***	
	Acceder	**No Acceder**
$amandaTrust	True	False
$mamaTrust	True	False

Tabla 2. Valores de decisiones en el Día 4- Pasar tiempo con su amiga Amanda y pasar tiempo con su madre

5.2.5 DÍA 5 – ETAPA SEXUAL

El último día varía según las decisiones tomadas por el protagonista. Al principio, éste se despierta, se prepara para ir al colegio y desayuna con su madre. En caso de que haya pasado tiempo con ella el día anterior, mencionarán posibles planes que pueden hacer toda la familia juntos. En caso contrario, continuarán desayunando en silencio.

Al terminar, si el día anterior el protagonista salió con Amanda, ella visitará su casa para ir juntos al colegio con su madre y harán planes para quedar al día siguiente. En caso contrario, el protagonista irá al colegio y Amanda le ignorará, cansada de ofrecer planes el protagonista para que siempre sea rechazada.

Al volver a casa tras terminar las classes, el protagonista se conectará una vez más y jugará con "Hally". Recordará el vídeo que le envió el día anterior y se sentirá una vez más incómodo. El depredador, mencionará ese mismo video y cambiará una vez más el tema de conversación por uno sexual, preguntando si lo había visto o no. Seguirá insistiendo sobre el tema hasta que decide enviar el mismo una foto sexual, volviendo a utilizar una foto del mismo adolescente que había mandado días antes esta vez sin camiseta (haciendo entender que este adolescente era una víctima suya anterior y que está usando sus fotos para manipular a más posibles víctimas). Una vez enviada la foto, insistirá al protagonista que envíe una suya, utilizando diversas técnicas manipulativas como, por ejemplo, diciendo que es "justo", dado que él ya había enviado una imagen o que se sentirá "dolido" si el protagonista no confía en él.

En este momento, el jugador tendrá que decidir si enviar la foto sexual o no. En caso de que se niegue, el acosador insistirá una vez más que confíe en él. Si el usuario le ha enviado una foto suya antes (decisión guardada en **$foto**), Hally le dirá que es de confianza dado que ya le ha enviado una foto antes. Además, si previamente el protagonista le ha contado sus secretos y preocupaciones (decisión guardada en **$contar**), Hally manipulará al jugador, haciéndole creer que ya había confiado en él antes sobre sus secretos y que podría hacer lo mismo ahora con la imagen.

Para los jugadores, es evidente cuál es la decisión "correcta" en este momento. Sin embargo, para simular que el protagonista ha sido completamente manipulado por el acosador, la decisión "No enviar foto" sólo aparecerá si tanto la variable **$foto** y **$contar** son "false", es decir, si el jugador no ha confiado en ningún momento en el usuario, no enviándole ninguna foto previa ni contándole su información privada. En caso de que el jugador haya accedido en alguna de las dos opciones previas, el jugador se verá obligado en elegir la decisión "Mandar foto", simulando que el acosador ha tenido éxito en la manipulación del protagonista.

Esta decisión diferenciará entre los finales "Normales" y "Malos" de la historia, creando en este momento dos grandes ramas en el árbol. Asimismo, los finales obtenidos dentro de estas

ramas se diferenciarán según de los valores guardados en las variables **$amandaTrust**, **$mamaTrust**. Todos los finales de esta historia, junto con los valores de sus variables, se encuentran en la *Tabla 3* mostrada a continuación.

Variables anteriores	***Decisiones/ Valores***	
	Enviar foto sexual	**No Enviar foto**
$amandaTrust = false; $mamaTrust = false	**Malo 1**: *Bullying* escolar, fotos online y el protagonista no se lo cuenta a nadie	**Normal 3:** al final al protagonista no le pasa nada, pero no le cuenta a nadie lo que ha pasado
$amandaTrust = false; $mamaTrust = true	**Malo 2:** *Bullying* escolar, el protagonista se lo cuenta a sus padres, pero no a Amanda	**Normal 2:** No ocurre nada grave, pero el protagonista cuenta la experiencia a sus padres
$amandaTrust = true; $mamaTrust = false	**Malo 3:** *Bullying* escolar, el protagonista no se lo cuenta a los padres, pero sí a Amanda	**Normal 1:** no ha ocurrido nada grave, pero cuenta la experiencia que ha vivido a Amanda
$amandaTrust = true; $mamaTrust = true	**Malo 4:** *Bullying* escolar, pero el protagonista confía en padres y en Amanda y se lo cuenta	**Bueno:** no ha ocurrido nada grave, pero el protagonista cuenta la experiencia tanto a su amiga como a sus padres

Tabla 3. Tabla de decisiones previas y finales obtenidos

5.2.6 FINALES

Como se ha podido observar en la tabla anterior, existen un total de 8 finales en esta historia: 4 finales malos, 3 normales y un único final bueno.

En los finales malos, el jugador se tiene que enfrentar a las amenazas del acosador. Al enviarle la foto sexual, al día siguiente "Hally" exigirá más imágenes al jugador. Éste se negará y comenzarán entonces las amenazas, asustando al protagonista e indicando que, si decide no enviar más, subirá todas las imágenes a Internet. Dado que el protagonista se niega a volver a enviar otra foto, el depredador cumple con su amenaza y sube las imágenes,

consiguiendo que todos los compañeros de clase del protagonista las vea y comiencen a hacer comentarios online. Este proceso marca el inicio del *ciberbullying* que el protagonista sufrirá por sus compañeros de clase. Mientras ve cómo sus compañeros comentan en las imágenes, el protagonista recibe un mensaje de Amanda preguntándole sobre lo ocurrido y verdaderamente preocupada por el protagonista. En este momento, se obtendrán distintos finales malos según las decisiones tomadas por el protagonista.

5.2.6.1 Final Malo 1

En caso de que el protagonista no haya pasado tiempo ni con Amanda ni con sus padres ($amandaTrust = false y $mamaTrust =false), el jugador no contestará a Amanda ni les contará a sus padres sobre lo ocurrido, soportando al *bullying* y la humillación en silencio.

5.2.6.2 Final Malo 2

En este final, a diferencia del anterior, el protagonista sí cuenta lo ocurrido a sus padres ($amandaTrust = false y $mamaTrust =true) dado que confía lo suficiente en ellos como para hacerlo. En este caso, los padres contactan con las autoridades, solicitando que quiten las fotos online y que investiguen sobre el acosador online que ha manipulado a su hijo.

5.2.6.3 Final Malo 3

Este final es el caso contrario al anterior: el protagonista responde a Amanda y le cuenta todo lo ocurrido. Ella, al entender la situación, le dirá que debe contárselo a sus padres, pero el protagonista, dado que no confía lo suficiente en ellos, no tiene la seguridad para hacerlo y le pide a Amanda que lo mantenga en secreto. Ella, a pesar de no estar de acuerdo, accede y le promete que no se lo contará ($amandaTrust = true y $mamaTrust =false). En el colegio, al enfrentarse al *bullying*, el protagonista soporta los comentarios con el apoyo de su amiga Amanda y se convierte en la única amiga que permanece a su lada dada las circunstancias. Sin embargo, no tomarán las medidas necesarias para denuncia a Hally e impedir que alguien más caiga víctima de sus manipulaciones.

5.2.6.4 Final Malo 4

Este final, el protagonista confía tanto en Amanda como en sus padres ($amandaTrust = true y $mamaTrust =true) por lo que cuando le cuenta a Amanda y ésta le dice que tiene que contárselo a sus padres, el protagonista accede y juntos se lo cuentan. En este final, a pesar de las circunstancias, el protagonista tiene el apoyo de su amiga Amanda durante el *bullying* y, además, los padres contactan con la policía para eliminar las fotos de Internet y denunciar a Hally como depredador sexual.

5.2.6.5 Final Normal 1

En estos finales, el protagonista no ha mandado las fotos sexuales al depredador, por lo que no se tiene que enfrentar ni al *bullying* online ni a la humillación de tener sus fotos colgadas en Internet. Sin embargo, ha sufrido una experiencia relacionada al acoso sexual y, dependiendo de su relación con sus familiares y amigos, contará lo sucedido o lo mantendrá en secreto. En este final específicamente, al día siguiente el protagonista queda con Amanda en su casa y juegan juntos al juego online. Cuando Amanda utiliza por un momento el ordenador del protagonista, ve por accidente los mensajes que ha recibido de "*Hally*". En ese momento, Amanda pregunta sobre la conversación que ha leído preocupada y, al saber que puede confiar en ella, el protagonista le cuenta lo ocurrido. Al terminar, Amanda le indica que tienen que contárselo a sus padres, pero como el protagonista no confía en ellos, pide que lo mantengan en secreto ($amandaTrust = true y $mamaTrust =false).

5.2.6.6 Final Normal 2

Este final es el caso contrario al anterior. Al día siguiente el protagonista pasa tiempo con su madre y su padre en vez de con Amana, preparando la cena todos juntos y pasando tiempo en familia. Durante la cena, al protagonista se le escapa que ha hecho un amigo online y cuando los padres preguntan sobre el tema preocupados, el protagonista confiesa todo lo que ha ocurrido ($amandaTrust = false y $mamaTrust =true). En ese momento, los padres contactan con la policía, informándoles sobre el acoso que ha sufrido su hijo.

5.2.6.7 Final Normal 3

Este final es el más "cotidiano" de los tres, donde le protagonista pasa tiempo con sus amigos y con sus padres, pero no menciona nada sobre el tema de su amigo online ni de las peticiones que ha recibido en los últimos días, no dándole importancia a la situación ya que al final no ha ocurrido nada grave. ($amandaTrust = false y $mamaTrust =false).

5.2.6.8 Final Bueno

El único final bueno de este juego ocurre cuando el protagonista no solo no ha enviado las fotos, sino que además ha pasado tiempo con Amanda y con sus padres. De esta manera, cuando Amanda ve el portátil del protagonista y le dice que tiene que contárselo a los padres, éste accede y juntos cuentan todo lo sucedido. Los padres contactan con la policía y les informa sobre el acoso de su hijo, consiguiendo en este final que nada malo hubiese ocurrido al jugador y además haya podido denunciar la situación a las autoridades, manteniendo en el proceso su relación con sus padres y con Amadna ($amandaTrust = true y $mamaTrust =true).

5.2.7 ESCENA FINAL

Al final del juego, una vez más el protagonista se tiene que mudar de casa. Dependiendo del final obtenido, las circunstancias durante la mudanza difieren. Al final del juego, hay una escena extra donde se observa una televisión, donde se presentan las noticias y se habla sobre el tema de *Online Grooming*. En caso de que el jugador haya confiado en los padres y contado lo ocurrido, contactando en el proceso con la policía, se indicará que el depredador sexual "Hally" ha sido detenido gracias a las aportaciones de las víctimas. En caso contrario, se indicará la dificultad de detener a los depredadores debido al silencio que mantienen las víctimas al sufrir la ofensa y el anonimato que caracteriza a Internet. En este caso, aparece el personaje "Hally" hablando con un jugador, presentándose y pidiendo jugar una partida. En otras palabras, se observa al depredador libre y en búsqueda de otra víctima para manipular.

5.2.8 ÁRBOL DE DECISIONES

En la *Figura 54* mostrada a continuación se puede observar el árbol de decisiones utilizado en esta aplicación, representando los días en los que se transcurre esta historia y las decisiones presentadas en cada uno de ellos. También se pueden observar el valor de las variables guardadas al tomar la decisión y, a consecuencia de ello, el fina obtenido según dichos valores.

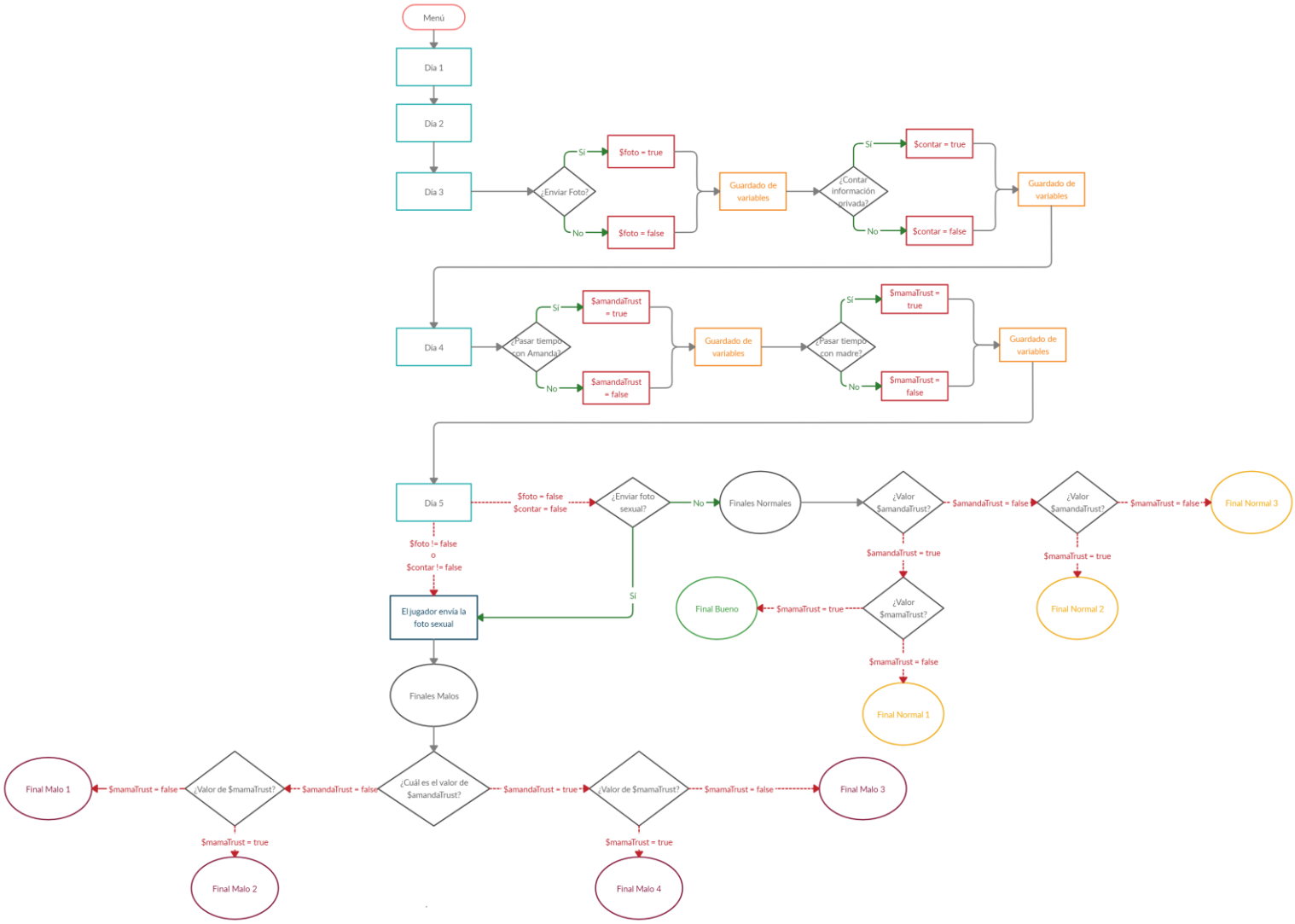

Figura 54. Árbol de decisiones de la historia

Capítulo 6. VALIDACIÓN DEL VIDEOJUEGO

En el desarrollo de este videojuego, se han realizado diversas validaciones y comprobaciones para asegurar el correcto funcionamiento de la aplicación, tanto en la conexión con la base de datos como con la compatibilidad del programa con los dispositivos IOS y Android.

6.1 VALIDACIÓN DURANTE EL DISEÑO DEL VIDEOJUEGO

En este proceso, dado que la aplicación no estaba completaba, era necesario una validación donde se pudiera ver que todas las modificaciones realizadas en el proyecto funcionaban correctamente en el dispositivo. Por lo menos, observar que las dimensiones de la pantalla de la aplicación se mantuvieran estables a pesar de la variación de ésta, simulando de esta manera diversos dispositivos de diferente tamaño

Unity proporciona una aplicación para dispositivos móviles denominada *Unity Remote 5* [23]. Esta plataforma proyecta el videojuego de *Unity* al dispositivo móvil conectado al ordenador, pudiendo obtener de esta manera una visualización previa a la posible aplicación resultante al finalizar el proyecto. En la *Figura 55*, se puede observar la captura de pantalla realizada desde un iPhone, mostrando la aplicación en el dispositivo móvil mientras ésta es ejecutada en *Unity*.

Figura 55. Aplicación en IPhone utilizando Unity Remote5

Como se puede observar en la figura, la calidad de imagen en la aplicación *Unity Remote5* se deteriora. Sin embargo, dado que esta aplicación simplemente proyecta la imagen de la aplicación ejecutada en *Unity* en el dispositivo, esta calidad no se verá reflejada en el resultado final de este proyecto. En otras palabras, *Unity Remote5* se ha utilizado para comprobar que la posición y la dimensión de las imágenes del proyecto son las correctas, no representa el resultado final de este proyecto.

6.2 VALIDACIÓN CON DISPOSITIVOS MÓVILES

Este proceso se ha realizado una vez finalizado la historia y las imágenes del proyecto, junto con las modificaciones y correcciones que se han realizado a lo largo de esta validación. Al obtener un videojuego funcional en *Unity*, en este proceso del proyecto se procedió a realizar diversas pruebas en dispositivos Android y IOS para comprobar su correcto funcionamiento. Debido a la diferencia entre ambas compañías, el proceso de validación de cada uno de ellos difiere. Sin embargo, en todos ellos se han validado las siguientes funciones: guardado de datos del usuario en la base de datos, guardado de partida, carga de partida y comprobación del uso de la información del usuario dentro de la historia.

Para la validación en un dispositivo iOS, debido a los problemas de compatibilidad que éstos presentan con portátiles Windows (sistema operativo donde se ha desarrollado este proyecto), se ha utilizado un segundo ordenador MAC para realizar este proceso. Una vez más, *Unity* proporciona un tutorial para usuarios de Apple donde se indican todos los pasos a seguir en el proceso de exportación un proyecto *Unity* a una aplicación iOS [24]. En resumen, se han realizado el siguiente procedimiento: desde *Unity* se ha exportado el proyecto y, una vez finalizada la ejecución, se ha abierto el archivo resultante en *Xcode*. Desde esta plataforma, siguiendo los pasos establecidos en el tutorial, se han modificado los ajustes necesarios para conseguir un correcto funcionamiento de la aplicación. Al finalizar, se ha conectado el dispositivo móvil al ordenador, en este caso un *IPhone 6S*, y se ha ejecutado el programa desde *Xcode*, instalando la aplicación resultante en el *IPhone*. Se puede observar el resultado obtenido en la *Figura 56* mostrada a continuación*:*

Figura 56. Validación en dispositivo IOS - IPhone 6

En el caso de validación para Android, dado que no se disponía de un teléfono móvil perteneciente a esta compañía, se ha utilizado un emulador denominado *BlueStacks* [25]. Este emulador permite utilizar y jugar aplicaciones móviles Android en un ordenador Windows. Este procedimiento era considerablemente más sencillo en comparación con el caso anterior, simplemente exportando el proyecto *Unity* a una aplicación tipo APK, una aplicación Android, y se ha abierto el programa exportado en este emulador. El resultado de esta validación se puede observar en la siguiente *Figura 57:*

Figura 57. Validación en Android – BlueStacks

En ambos casos se puede observar que la calidad de imagen de la aplicación mejora considerablemente en comparación al resultado obtenido con *Unity Remote 5*, primer paso establecido en esta sección, demostrando de esta manera que la imagen resultante en ese proceso se debía simplemente a la aplicación de *Unity* y no a la calidad de este proyecto.

6.3 VALIDACIÓN CON LA BASE DE DATOS

Finalmente, al validar el funcionamiento de la aplicación en diversos dispositivos, se procedió a comprobar la base de datos, observando si la información de los usuarios se ha guardado correctamente en ella.

En la siguiente *Figura 58* se puede observar la base de datos obtenida tras finalizar con el proceso de validación de compatibilidad previamente realizado.

Figura 58. Validación de escritura de la base de datos utilizando diversos p3aaerfiles y dispositivos

Como se puede observar en la imagen, en una de las ramas pertenecientes al ordenador PC utilizado en este proyecto, se han almacenado los datos relacionados con una jugadora llamada "*Laura*". En él podemos encontrar la rama "*INFO*", donde se ha realizado la

escritura de sus datos personales. Existe otra rama, sin embargo, denominada "*TimesPlayed*" donde se puede observar que se han obtenido 3 finales diferentes con este jugador, guardando en el proceso de escritura todas las decisiones tomadas en cada uno de ellos.

Como se puede observar, además, existen otros dos usuarios "*Ana*" y "*Laura*", guardados en un identificador diferente, indicando que se han realizado desde otro dispositivo. Esta conexión se ha realizado desde *BlueStacks*, simulando la conexión desde un dispositivo Android.

Por último, existen otros dos identificadores en la base de datos: el primero, donde se encuentra el usuario "*Luis*", es el resultado obtenido al realizar la validación en un dispositivo iOS, mientras que el segundo, donde se encuentra el jugador "*Pablo*", se ha obtenido al ejecutar el proyecto de *Unity* desde un ordenador Mac.

En conclusión, estas escrituras realizadas desde diversos dispositivos demuestran que la conexión a la base de datos se realiza correctamente y que la estructura obtenida al realizar la escritura cumple con los objetivos establecidos en este proyecto, almacenando tanto la información del jugador como todas sus rutas y decisiones tomadas a lo largo de la historia.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

El mundo de los videojuegos ha sufrido un gran cambio en los últimos años. A medida que las tecnologías van evolucionando, surgen nuevos géneros en este mercado, además de dispositivos de tecnología avanzada que introducen una nueva herramienta a utilizar en esta industria. A pesar de que esta evolución es beneficiosa para la humanidad, ya sea en el mundo del entretenimiento o en nuestras necesidades de la vida cotidiana, junto con estos nuevos descubrimientos también surgen nuevas herramientas que los criminales en Internet pueden utilizar para su propio beneficio. Ante nuevas tecnologías, nuevos peligros acechan, ya sea hacia nosotros mismos o hacia los más vulnerables de nuestra sociedad, los jóvenes.

Entre estas amenazas se encuentra el *Online Grooming*, una técnica manipulativa que utilizan los depredadores sexuales en Internet para ganarse la confianza y amistad de los más jóvenes en las redes sociales, con el objetivo de interaccionar de manera sexual con el menor, ya sea a través de imágenes, mensajes o incluso realizando un encuentro en la vida real. Dado que estas nuevas tecnologías han contribuido en el desarrollo de esta amenaza, estas herramientas pueden ser usadas una vez más para nuestro beneficio, desarrollando de esta manera un método interactivo que capte el interés de los jóvenes, además de su atención, y utilizarlo para enseñarle sobre los engaños que se encuentran en Internet.

7.1 CONCLUSIONES

En este proyecto, que tiene la finalidad de enseñar sobre las técnicas manipulativas que utilizan los depredadores sexuales en Internet, se ha desarrollado una aplicación móvil donde se muestra la historia de un adolescente que, debido a su soledad y carencia de relaciones con familiares y amigos, recurre a Internet para relacionarse con usuarios desconocidos. En esta novela visual el jugador toma control de la vida del protagonista y es responsable de las decisiones que debe tomar a lo largo de la historia. Dado que este juego tiene un fin educativo, se han utilizado diversos trabajos de investigación relacionados con el *Online*

Grooming con el fin de mostrar el *Modus Operandi* que utilizan estos acosadores online. De esta manera, el jugador pueda aplicar dichos conocimientos en la vida real y protegerse de los engaños de Internet.

Este videojuego desarrollado en *Unity*, utilizando *Yarn* como lenguaje de diálogo, muestra una temática que apenas se ha desarrollado en el mercado actual de los videojuegos. Los centros educativos están realizando diversos eventos y actividades para enseñar a los jóvenes cómo manejar y defenderse en Internet. Sin embargo, si éstos no muestran interés hacia la materia, muchas de estas iniciativas pueden resultar ineficientes. Por ese motivo, al desarrollar un videojuego donde los jóvenes se sientan familiarizados y se interesen por la temática, se cuenta una historia que existe actualmente en nuestra sociedad y nos afecta gravemente, una historia que trata sobre manipulación y que, desgraciadamente, múltiples usuarios de Internet caen víctimas de su engaño.

Desde el punto de vista personal y de aprendizaje, este Trabajo Fin de Grado me ha permitido conocer mejor los riesgos a los que los menores se enfrentan en Internet y, concretamente, en qué consiste el *Online Grooming*, así como conseguir experiencia de primera mano con tecnologías como *Unity*, *Yarn* o *Firebase*, que no habían sido cubiertas en el Grado. Esta experiencia puede ser clave de cara a mi futuro desarrollo profesional.

7.2 TRABAJOS FUTUROS

En este proyecto se ha realizado el desarrollo de una aplicación móvil, compatible tanto para iOS como para Android, donde se ha realizado el diseño de un árbol de decisiones que forman la base de la historia y, además, el diseño artístico del entorno que le rodea. En esta aplicación, se ha realizado también el diseño de una base de datos utilizando *Firebase*, donde se almacenan tanto los datos personales del jugador como los finales obtenidos por éste, junto con sus decisiones tomadas a lo largo de la historia.

A pesar de que todos los objetivos planteados para este proyecto se han cumplido en la actualidad, existen ciertos trabajos futuros que no solo tienen la posibilidad de mejorar la

historia de éste, sino que se pueden añadir otros elementos adicionales que mejoren la jugabilidad y la experiencia del jugador. Asimismo, también se plantea como trabajo futuro el envío de un artículo sobre el Trabajo Fin de Grado a las Jornadas Nacionales de Investigación en Ciberseguridad (JNIC), que este año han sido suspendidas debido a la pandemia de COVID-19.

7.2.1 COLABORACIÓN CON PSICÓLOGOS

Un proyecto futuro relacionado con este trabajo que se puede realizar es la colaboración con psicólogos y expertos en *Online Grooming* que puedan aconsejar en diversos aspectos de la historia y, de esta manera, mejorar el entorno de la aplicación. Con esta colaboración, se puede conseguir incluso la implementación de otros elementos o temáticas que no se hayan tenido en cuenta en este proyecto y que permitan ampliar el conocimiento del jugador.

A pesar de que en este trabajo se hayan utilizado diversos informes de investigación sobre *Online Grooming*, la colaboración con expertos y psicólogos da la oportunidad de conseguir una historia más realista y eficiente en la enseñanza al jugador sobre las técnicas manipulativas utilizadas por los agresores.

7.2.2 ESTUDIO DE RESULTADOS DE JUGADORES

Como se ha mencionado previamente, esta aplicación utiliza una base de datos donde se almacenan los datos de los jugadores y sus decisiones tomadas en el videojuego. Esta funcionalidad da la posibilidad de utilizar esta información en futuros estudios cuantitativos sobre el conocimiento de los usuarios en relación con *Online Grooming*, realizando de esta manera un perfil genérico de las decisiones tomadas por los usuarios teniendo en cuenta su edad y género, por ejemplo.

Para poder realizar este perfil, se necesitan jóvenes dispuestos a participar en este proyecto, obteniendo de esta manera sus datos personales y decisiones tomadas. Una posibilidad es colaborar con diversas instituciones académicas que puedan realizar una actividad donde todos sus estudiantes juegan a esta aplicación. De esta manera, no sólo se les educará a estos jóvenes sobre *Online Grooming*, sino que además toda su información será almacenada en

la base de datos y estará disponible para futuros estudios donde se deseen cubrir esta temática.

7.2.3 JUEGO MÁS DINÁMICO E INTERACTIVO

Esta aplicación se trata de una novela visual, género de la industria de los videojuegos que consiste en mostrar narraciones y conversaciones entre personajes, utilizando imágenes como representación de los protagonistas y fondos para establecer el entorno de la escena. A pesar de que es una metodología altamente utilizada en la industria, debido a la carencia de actividades interactivas dentro del videojuego, este género suele atraer a un público más adulto en comparación con otros estilos de videojuegos.

Por este motivo, dado que se desea atraer a jóvenes preadolescentes de edades entorno los 10-14 años, se plantea para un trabajo futuro introducir dinámicas dentro de esta aplicación que ayuden al entretenimiento del usuario. Éstas pueden consistir en mini-juegos, tomando como ejemplo el videojuego mencionado en el *Capítulo 2* de esta memoria *Cyberscouts*, donde se ponga a prueba al jugador realizando diversas preguntas que debe responder o incluso recreando niveles con diversas actividades donde el usuario interacciona para continuar con la historia.

Estos niveles pueden compartir la misma temática que la aplicación, ésta siendo *Online Grooming*, o incluso tratar una temática distinta, abarcando conceptos de *ciberseguridad*. Un ejemplo de un nivel a superar puede consistir en el protagonista navegando por la red y debe evitar diversos virus o *hacker* que desean acceder a su dispositivo.

Dado que la novela visual se encarga de educar y mostrar al jugador las técnicas manipulativas que utilizan los depredadores sexuales en la red, con este trabajo se puede conseguir una aplicación dinámica y entretenida para el usuario, captando de esta manera su interés y atención, consiguiendo de esta manera un resultado más eficiente en la educación sobre los peligros de Internet.

Capítulo 8. BIBLIOGRAFÍA

[1] R. A., «"¿Cuáles son los origenes de Internet?",» 5 Septiembre 2015. [En línea]. Available: https://www.mastermarketingdigital.com/everriculum/2014/09/05/cuales-son-los-origenes-de-internet/.

[2] J. F. C. Lobo, «Juegos Serios: Alternativa Innovadora,» II Congreso en líne en Conocimiento Libre y Educación CLED2011, 2014. [En línea]. Available: http://erevistas.saber.ula.ve/index.php/cled/article/viewFile/4862/4680.

[3] M. Gámez-Gaudix y P. de Santistebam, «Online Grooming y Explotación Sexual de Menores a Través de Internet,» Revista Victimología, 2017. [En línea]. Available: http://www.huygens.es/journals/index.php/revista-de-victimologia/article/view/103.

[4] «Unity,» [En línea]. Available: https://unity.com/es/products/core-platform.

[5] «Unity - Bases de Unity,» [En línea]. Available: https://docs.unity3d.com/es/530/Manual/UnityBasics.html.

[6] «Unity - Tutoriales,» [En línea]. Available: https://unity.com/es/learn/get-started.

[7] Secret Lab y Yarn Spinner Contributors, «YarnSpinner,» 2020. [En línea]. Available: https://yarnspinner.dev/docs/tutorial.

[8] «Clip Studio Paint,» [En línea]. Available: https://www.clipstudio.net/es/.

[9] «Firebase,» [En línea]. Available: https://firebase.google.com/.

[10] «Firebase - Agrega Firebase a tu proyecto de Unity,» [En línea]. Available: https://firebase.google.com/docs/unity/setup.

[11] «Industria de los videojuegos,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Industria_de_los_videojuegos.

[12] «Secret Lab,» How Night in the Woods Uses YarnSpinner, 14 Noviembre 2017. [En línea]. Available: https://www.secretlab.com.au/blog/2017/11/14/how-night-in-the-woods-uses-yarn-spinner.

[13] «"Juego Cyberscouts",» Internet Segura for Kids, [En línea]. Available: https://www.is4k.es/de-utilidad/cyberscouts.

[14] e-UCM, «"Conectado, un videojuego para concienciar sobre el acoso y ciberacoso escolar",» e-Learning group, [En línea]. Available: https://www.e-ucm.es/es/portfolio-item/conectado/.

[15] «"Break time! 6 Cybersecurity Games that You'll Love",» HelpSystems, 20 Marzo 2019. [En línea]. Available: https://www.helpsystems.com/blog/break-time-6-cybersecurity-games-youll-love.

[16] Fundación ANAR y Mutua Madrileña, «ANAR,» 2016. [En línea]. Available: https://www.anar.org/wp-content/uploads/2017/04/INFORME-II-ESTUDIO-CIBERBULLYING.pdf.

[17] «Save the Children,» Análisis de la violencia contra la infancia y la adolescencia en el entorno digital, 2019 Julio. [En línea]. Available: https://www.savethechildren.es/publicaciones/informe-violencia-viral-y-online-contra-la-infancia-y-la-adolescencia.

[18] P. A. Gonzalez, «Online Grooming,» Abril 2019. [En línea]. Available: https://repositorio.comillas.edu/xmlui/bitstream/handle/11531/30848/TFG-%20Alonso%20Gonzalez%2C%20Patricia.pdf?sequence=1&isAllowed=y.

[19] EuropaPress, «EpData,» 22 Octubre 2019. [En línea]. Available: https://www.epdata.es/datos/uso-jovenes-internet-datos-graficos/271.

[20] Y. R., «"Roperwork Visual Novel",» GitHub, [En línea]. Available: https://github.com/radiatoryang/ropework.

[21] «Unity Store,» [En línea]. Available: https://store.unity.com/#plans-business.

[22] «Clip Studio Pain - Subscripciones,» [En línea]. Available: https://www.clipstudio.net/en/.

[23] «Unity Remote 5,» Unity Documentation, [En línea]. Available: https://docs.unity3d.com/Manual/UnityRemote5.html.

[24] «Tutorial Unity - Exportar Unity a IOS,» [En línea]. Available: https://learn.unity.com/tutorial/building-for-mobile#5c7f8528edbc2a002053b4a2.

[25] «BlueStacks,» [En línea]. Available: https://www.bluestacks.com/es/index.html.

[26] «Unity,» [En línea]. Available: https://unity.com/.

[27] «Firebase,» [En línea]. Available: https://firebase.google.com/.

[28] «Clip Studio Paint,» [En línea]. Available: https://www.clipstudio.net/en/.

ANEXO: RELACIÓN DEL TRABAJO FIN DE GRADO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE (ODS)

Los Objetivos y Metas de Desarrollo Sostenible (ODS) es un conjunto de objetivos globales que los líderes mundiales adoptaron de manera colectiva con el objetivo de erradicar los problemas principales que se encuentran actualmente en nuestro planeta. Entre ellos, se encuentran: la pobreza, protección del planeta y asegurar la prosperidad para toda la humanidad. Este proyecto se divide en 17 objetivos individuales, cada uno abarcando metas específicas, y se marcó como fecha límite de la compleción de todas estas metas para 2030.

Este proyecto elaborado por los líderes mundiales tiene como objetivo conseguir un futuro mejor y más sostenible para todos nosotros, objetivo que se desea colaborar con este trabajo de fin de grado. Existen múltiples proyectos ingenieriles que pueden realizar una gran aportación en esta colaboración de ayuda para toda la humanidad y, a pesar de que en este trabajo no se obtenga resultados inmediatos y de mayor dimensión como otros existentes, tiene un propósito que puede beneficiar a futuras generaciones y en la educación de la juventud actual.

CONTEXTUALIZACIÓN

Este proyecto, como se ha establecido a lo largo de esta memoria, tiene como objetivo educar a la población joven, especialmente a los adolescentes, sobre las amenazas existentes en la red mundial a la que se conectan diariamente y confían sin temor alguno. En ella, existen depredadores sexuales que aprovechan el anonimato que aporta Internet para manipular a los más vulnerables con el objetivo de obtener resultados sexuales que dañan gravemente la salud del menor, ya sea de manera psicológica como, en caso de consecuencia de *bullying* o encuentro en persona, física.

Ya que este proyecto se basa en la educación y protección de menores, se ha podido identificar las siguientes conexiones con los objetivos existentes dentro de ODS (en la *Tabla 4* de este Anexo se puede observar la representación gráfica de estas relaciones):

- Objetivo 3 Salud y bienestar: *Garantizar una vida saludable y promover el bienestar para todos y todas en todas las edades*. Este proyecto se centra en educar sobre las tácticas de manipulación que suele utilizar un depredador sexual en Internet, con el objetivo de que el usuario pueda identificar esas señales en la vida real y actuar con precaución, evitando la manipulación del agresor y el daño contra el menor. En otras palabras, esta aplicación es una medida educativa y preventiva que protege la salud y bienestar de los jóvenes usuarios de Internet
- Objetivo 4 Educación de calidad: *Garantizar una educación de calidad inclusiva y equitativa, y promover las oportunidades de aprendizaje permanente para todos.* La aplicación desarrollada en este trabajo pertenece al género denominado "*juegos serios*", ya tiene como objetivo educar a los jugadores de manera interactiva, objetivo que se encuentra dentro de las iniciativas establecidas en la ODS.
- Objetivo 11 Ciudades y comunidades sostenibles: *Conseguir que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles*. Con este proyecto, se desea proteger a los más vulnerables de los peligros que existen en Internet. Si el resultado es positivo, la consciencia de los usuarios hacia los depredadores sexuales y manipuladores que se encuentran en la red crecerá, por lo que se podrá observar una reducción en el número de víctimas relacionadas a este tipo de acoso online. En otras palabras, al aumentar el conocimiento sobre esta amenaza, al navegar por Internet los jóvenes podrán identificar en cierta medida a estos depredadores, consiguiendo que la red se convierta un lugar poco más seguro.

Dimensión ODS	*ODS Identificados*	*Papel*	*Objetivos*
Salud y Bienestar	ODS 3: Garantizar una vida saludable y promover el bienestar para todos y todas en todas las edades	Primario	Proteger a los más vulnerables de los peligros de Internet, entre ellos los depredadores sexuales online
Educación	ODS 4: Garantizar una educación de calidad inclusiva y equitativa, y promover las oportunidades de aprendizaje permanente para todos.	Primario	Educar sobre los métodos manipulativos utilizados por los acosadores sexuales para poder identificarlos en la vida cotidiana
Ciudades y comunidades sostenibles	ODS 11: Conseguir que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles.	Secundario	Mejorando la educación de la juventud, se plantea disminuir de esta manera el número de víctimas dañadas por las consecuencias del *Online Grooming*

Tabla 4. Análisis de la relación entre ODS y el proyecto planteado

Los resultados de este proyecto dependen en gran medida en la psicología de estos depredadores y, dado que ésta no se puede determinar de manera exacta y definitiva, el método educativo presentado en este trabajo puede ser más o menos efectivo en la detección de las técnicas manipulativas utilizadas en *Online Grooming*. Cada depredador utiliza diversos métodos, por lo que se sólo se ha podido estimar un comportamiento genérico de éste. Por ese motivo, a pesar de que los objetivos de este proyecto coincidan con las iniciativas planteadas en la ODS, no se podrán observar resultados definitivos hasta que se realice un estudio exhaustivo con usuarios que hayan jugado a esa aplicación y hayan podido identificar técnicas de *Online Grooming* en su vida cotidiana.

A pesar de que no se puedan obtener datos precisos en la fecha actual, se debe tener en cuenta la finalidad que se presenta en este trabajo. En el desarrollo de este proyecto se han realizado diversos estudios que analizan en el comportamiento de estos depredadores sexuales, se han observado los dispositivos que más utilizan los jóvenes en la actualidad y, además, se ha descubierto la existencia de un gran número de víctimas de *Online Grooming* que carecían el conocimiento necesario para protegerse de esta amenaza. Todos estos procedimientos se han realizado con el objetivo de educar y proteger al jugador. Aunque los datos no sean

inmediatos ni se puedan calcular de manera cuantitativa, esta aplicación presenta una aportación en la protección y seguridad de los más vulnerables, colaboración que coincide con los objetivos establecidos por la ODS.