Cat_ch&Care

Trabajo Final Profesionalizador

Autor: Joaquín Bea Bonet
Tutor: Gustau Marcos Ballester.
Profesor: Joan Arnedo Moreno

Grado de Ingeniería Informática
Itinerario de Computación

Créditos/Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Cat_ch&Care*
Nombre del autor:	*Joaquin Bea Bonet*
Nombre del colaborador/a docente :	*Gustau Marcos Ballester*
Nombre del PRA:	*Joan Arnedo Moreno*
Fecha de entrega (mm/aaaa):	*01/2024*
Titulación o programa:	*Grado de Ingeniería informática*
Área del Trabajo Final:	*TFG - Videojuegos*
Idioma del trabajo:	*Español o inglés*
Palabras clave	*Juego, gato, tamagotchi, Trabajo de Final de Grado, Memoria, Realidad Aumentada, Unity*

Resumen del Trabajo (máximo 250 palabras):

La finalidad del presente TFG es crear un videojuego para plataformas móviles que, a través de su uso, conciencie sobre las dificultades que experimentan los gatos para sobrevivir en las calles de nuestras ciudades y la labor que las protectoras de animales realizan en ese sentido.

Se pretende obtener una visión vertical del juego en forma de “DEMO”, donde se muestren los elementos principales del mismo, con un nivel de calidad parecido al que tendrían en su versión final. Este documento incluye la descripción de las fases del proyecto, desde el concepto inicial hasta la implementación final, pasando por la mecánica del juego, el diseño de los protagonistas de la historia y la IA que determina su interacción con el usuario. También se da un repaso a la parte técnica del proyecto al tiempo que se explican los detalles del motor de juego utilizado y los lenguajes que se han requerido para su programación.

El entregable final, junto con esta memoria, será una versión del juego en forma de “DEMO” para que pueda evaluarse en primera persona el resultado del trabajo realizado.

Abstract (in English, 250 words or less):

The purpose of this TFG is to create a video game for mobile platforms that, through its use, raises awareness about the difficulties that cats experience to survive on the streets of our cities and the work that animal protection orgs perform in that regard.

The aim is to obtain a vertical view of the game in the form of a “DEMO”, where the main elements of the game are shown, with a level of quality like what they would have in their final version. This document includes the description of the project phases, from the initial concept to the final implementation, going through the game mechanics, the design of the protagonists of the story and the AI that determines the user interaction. The technical part of the project is also reviewed while explaining the details of the game engine used and the languages that have been required for its programming.

The final deliverable, together with this report, will be a version of the game in the form of a “DEMO” so that the result of the work may be evaluated in first hand.

Dedicatoria/Cita

John Lennon dijo que "*La vida es lo que te pasa, mientras estás ocupado haciendo planes*". Yo digo que la vida es lo que te pasa mientras estás ocupado estudiando y trabajando. Dedico este grado de ingeniería informática a Olga y a nuestros gatos rescatados de la calle, Puça, Mama G y Chester, que han tenido que soportarme durante todos estos años de estudio. A mis hijos Pau y Marc, que me han animado a no abandonar y seguir hasta el final. A mi Sobrino David, por sus largas horas de *beta tester*. Y a mis padres Maribel y Joaquin, que me han enseñado que lo mejor de la vida se consigue sólo con mucho esfuerzo y amor.

Agradecimientos

A la UOC y a todo el profesorado del grado de Ingeniería Informática por lo mucho que he aprendido en estos años de estudio.

A Montserrat Fernández Bartra, en su labor de tutora, por ayudarme a elegir el mejor camino dentro del grado, siempre con una cordial sonrisa virtual 😊.

A Gustau Marcos Ballester, por su efectiva guía en este trabajo de fin de grado y por su infinita paciencia conmigo durante estos meses de interminables desafíos.

Resumen

La finalidad del presente TFG es crear un videojuego para plataformas móviles que, a través de su uso, conciencie sobre las dificultades que experimentan los gatos para sobrevivir en las calles de nuestras ciudades y la labor que las protectoras de animales realizan en ese sentido.

Se pretende obtener una visión vertical del juego en forma de “DEMO”, donde se muestren los elementos principales del mismo, con un nivel de calidad parecido al que tendrían en su versión final. Este documento incluye la descripción de las fases del proyecto, desde el concepto inicial hasta la implementación final, pasando por la mecánica del juego, el diseño de los protagonistas de la historia y la IA que determina su interacción con el usuario. También se da un repaso a la parte técnica del proyecto al tiempo que se justifica y explican los detalles del motor de juego utilizado y los lenguajes que se han requerido para su programación.

El entregable final, junto con esta memoria, será una versión del juego en forma de “DEMO” para que pueda evaluarse en primera persona el resultado del trabajo realizado.

Palabras clave

Juego, gato, tamagotchi, Realidad Aumentada, Unity, Trabajo de Final de Grado, Memoria.

Abstract

The purpose of this TFG is to create a video game for mobile platforms that, through its use, raises awareness about the difficulties that cats experience to survive on the streets of our cities and the work that animal protection orgs perform in that regard.

The aim is to obtain a vertical view of the game in the form of a “DEMO”, where the main elements of the game are shown, with a level of quality like what they would have in their final version. This document includes the description of the project phases, from the initial concept to the final implementation, going through the game mechanics, the design of the protagonists of the story and the AI that determines the user interaction. The technical part of the project is also reviewed while justifying and explaining the details of the game engine used and the languages that have been required for its programming.

The final deliverable, together with this report, will be a version of the game in the form of a “DEMO” so that the result of the work may be evaluated in first hand.

Key words

Game, cat, tamagotchi, Augmented Reality, Unity, Project Abstract.

Notaciones y Convenciones

- Se han utilizado fuentes *itálicas* en los casos en que ha sido necesario el uso de anglicismos.
- En el resto del documento se ha utilizado fuente Arial 11 ppt a 1.5 puntos de espaciado.
- La bibliografía se ha anotado siguiendo el estilo Harvard fecha-autor/origen, por orden de aparición en el texto.
- Se ha utilizado fuente `Cascadia Mono` para los fragmentos de código.
- Las funciones y clases se han nombrado siguiendo la notación PascalCase
- Las variables se han declarado siguiendo la notación camelCase

Índice

Figuras y tablas

Índice de figuras

Índice de tablas

1. Introducción

1.1. Prefacio

"Caminando por la calle yo te vi y me enamoré de ti. Pero seguí caminando y te volví a ver, en un descampado, en un vertedero, en las aceras, en los parques, buscando en la basura, bajo los coches… en todas partes. Porque no eras uno, eras uno tras otro. Todos desamparados, abandonados, en un limbo jurídico, porque nadie quiere hacerse responsable, legalmente, de vosotros: Gatos callejeros". [1]

Los gatos callejeros forman parte de la fauna de grandes y pequeñas ciudades, tan integrados en el paisaje urbano, que tienden a pasar desapercibidos en el frenético devenir de nuestras atareadas vidas cotidianas. La mayoría de ellos se encuentran en estado feral, algunos abandonados, otros huidos y muchos nacidos en estado semisalvaje, en un ambiente que para ellos se presenta hostil, han aprendido a malvivir en libertad, sorteando episodios de hambruna, sequia, pandemias, persecuciones y exterminios.

Su extraordinaria fertilidad hace que, a menudo, las colonias de gatos se conviertan en comunidades incontroladas que pueden llegar a deteriorar las condiciones de salubridad de su entorno. En ocasiones, los ayuntamientos no asumen totalmente las competencias municipales que les corresponden y delegan la gestión de las colonias felinas, a asociaciones protectoras de animales que no suelen contar con los recursos necesarios para hacer frente a la captura, vacunación, esterilización y control de las comunidades gatunas.

Figura 1: Colonia Felina, proyecto CES. Extraída de Aragón Noticias (CARTV) el 4 de Oct 2023

1.2. Descripción

La finalidad del presente TFG es crear un videojuego para plataformas móviles que, a través de su uso, conciencie sobre las dificultades que experimentan los gatos para sobrevivir en

las calles de nuestras ciudades y la labor que las protectoras de animales realizan en ese sentido.

Se pretende obtener una visión vertical del juego en forma de “DEMO”, donde se muestren los elementos principales del mismo, con un nivel de calidad parecido al que tendrían en su versión final. Se explorará, pero no se abordará en este TFP, la posibilidad de monetizar la experiencia de juego, bien en forma de donaciones, bien en forma de publicidad, con el ánimo de recaudar fondos, sin ánimo de lucro, para las asociaciones que ayudan y protegen a los animales.

Este documento incluye la descripción de las fases del proyecto, desde el concepto inicial hasta la implementación final, pasando por la mecánica del juego, el diseño de los protagonistas de la historia y sus formas de interacción con el usuario. También se da un repaso a la parte técnica del proyecto, explicando la justificación y los detalles del motor de juego utilizado, los algoritmos de IA que gobiernan a los protagonistas y los lenguajes que se han requerido para su programación.

El entregable final, junto con esta memoria, será una versión del juego “DEMO” para que pueda evaluarse en primera persona el resultado del trabajo realizado.

1.2.1. Experiencia de juego

La experiencia de juego se centrará en conseguir motivar e implicar al usuario a través, principalmente, de las siguientes estrategias *[2]*: **Una historia única** para cada gato, **coleccionar,** tanto gatos como *ítems* y obtener ***Feedback*** instantáneo del progreso del jugador.

1.2.2. Mecánica del juego

El objetivo principal del juego consiste en establecer un vínculo afectivo con el gato virtual, de forma que la mascota se sienta feliz y prefiera vivir con la persona que la cuida, a la posibilidad de escapar y regresar a la calle. Como se trata de una parte fundamental de la experiencia de juego, los medidores de felicidad y de vínculo se encontrarán siempre visibles en la pantalla principal, a modo de ejemplo, se muestra una funcionalidad similar en la Figura 2, extraída del juego Pokemon Go *[3]*:

Figura 2: En la imagen, en la esquina superior izquierda, medidor del nivel de compañerismo.

Después de detectar uno o varios planos con la cámara, se deberá tocar en un punto de uno de ellos para que aparezca el gato virtual. El jugador deberá reforzar el vínculo con la mascota de las siguientes formas: alimentándolo, acariciándolo o jugando con él o ella.

1.3. Objetivos generales

Se describen, a continuación, los objetivos del TF por orden de relevancia.

1.3.1. Objetivos principales

Los objetivos principales que se pretende conseguir con el desarrollo e implementación de este trabajo de final de grado son varios y distintos, en función del área de interés.

Desde el punto de vista del videojuego:

- Desarrollar una aplicación para dispositivos Android basada en un juego de RA.
- Definición e implementación de la IA de los gatos virtuales, en función de su personalidad.
- Definición e implementación de las mecánicas principales y secundarias.

Para el cliente/usuario:

- Estrechar vínculos con un gato callejero virtual cuidando de él (alimentar, jugar, acariciar) para que la mascota del juego permanezca fiel, junto a la persona que pretende adoptarle, creciendo a salvo de los peligros de la calle.

- Progresar en el mundo de las casas de acogida hasta convertirse en “hotel de acogida” de 5 estrellas (nivel 5).
- Acumular monedas (Euro-Cats) que podrán intercambiarse por ítems.
- Conseguir diferentes *Items* durante el juego, a través de intercambiarlos por monedas como, por ejemplo: juguetes, super juguetes, comida normal, super comida, etc.

Para el autor del TF:

- Aprender a programar en entorno UNITY y lenguaje C#
- Aprender a definir la arquitectura general de un videojuego de RA
- Aprender e implementar los algoritmos de IA necesarios para la correcta evolución del videojuego.

1.3.2. Objetivos secundarios

Objetivos adicionales que enriquecen el TF.

- Concienciar sobre las dificultades que experimentan los gatos para sobrevivir en las calles de nuestras ciudades y la labor de las protectoras de animales.
- Conocer el estado del arte de los videojuegos de RA.
- Conocer las principales plataformas de desarrollo de videojuegos.
- Conocer formas de monetizar un videojuego de estas características.

1.4. Metodología y proceso de trabajo

Para desarrollar Cat-ch&Care se ha optado por crear un producto nuevo, basado en el concepto clásico del Tamagotchi, pero dando un enfoque de coleccionismo felino, es decir, en lugar de criar a una mascota que va evolucionando en el tiempo, en este caso se trata de adoptar la mayor cantidad posible de gatos. Quien evoluciona, en lugar de la mascota, es el usuario al progresar en el mundo de las casas de adopción.

La base del juego serán los algoritmos de inteligencia artificial de las diferentes mascotas virtuales, por tanto se dedicará tiempo, especialmente durante la fase de conceptualización y diseño, a estudiar los ya existentes y proponer adaptaciones o evaluar la necesidad de nuevos desarrollos.

La implementación de un videojuego es un proyecto que requiere conocimientos multidisciplinares en diferentes áreas, como por ejemplo, diseño 2D y 3D, programación, integración multimedia, etc. Por tanto, se intentará aprovechar *assets* ya existentes, sobre

todo en los ámbitos que no sean de dominio del autor como, por ejemplo, en el diseño de modelos 2D-3D.

En cuanto a la metodología de desarrollo, se utilizará un modelo mixto entre *waterfall* y *agile*. De esta manera, de acuerdo con el paradigma clásico de desarrollo *waterfall*, se establecerán fases por las que el proyecto deberá pasar (Concepto, Diseño, Implementación y Pruebas), pero dentro de cada fase se establecerá un modelo *agile* de trabajo, donde se definan funcionalidades específicas que puedan ser probadas y evaluadas, bien en forma de concepto, en las fases iniciales, bien en forma de prototipo en las fases finales. El objetivo del desarrollo será ir teniendo conceptos o prototipos jugables, con funcionalidad creciente, para ir probando con usuarios continuamente y no tener que esperar a la fase final de pruebas para realizar las primeras evaluaciones del juego.

1.5. Planificación

El trabajo se ha planificado siguiendo el esquema de las PECs de la asignatura. Para cada funcionalidad desarrollada, se seguirán las siguientes fases: concepto, diseño, implementación y pruebas. Como funcionalidades destacadas, se deberán implementar: las mascotas virtuales, las mecánicas del juego, los distintivos de tipo de casa de acogida y los objetos.

Se recoge la planificación en el diagrama de Gantt de la Figura 5, generado con el *software* on line Tom's planner:

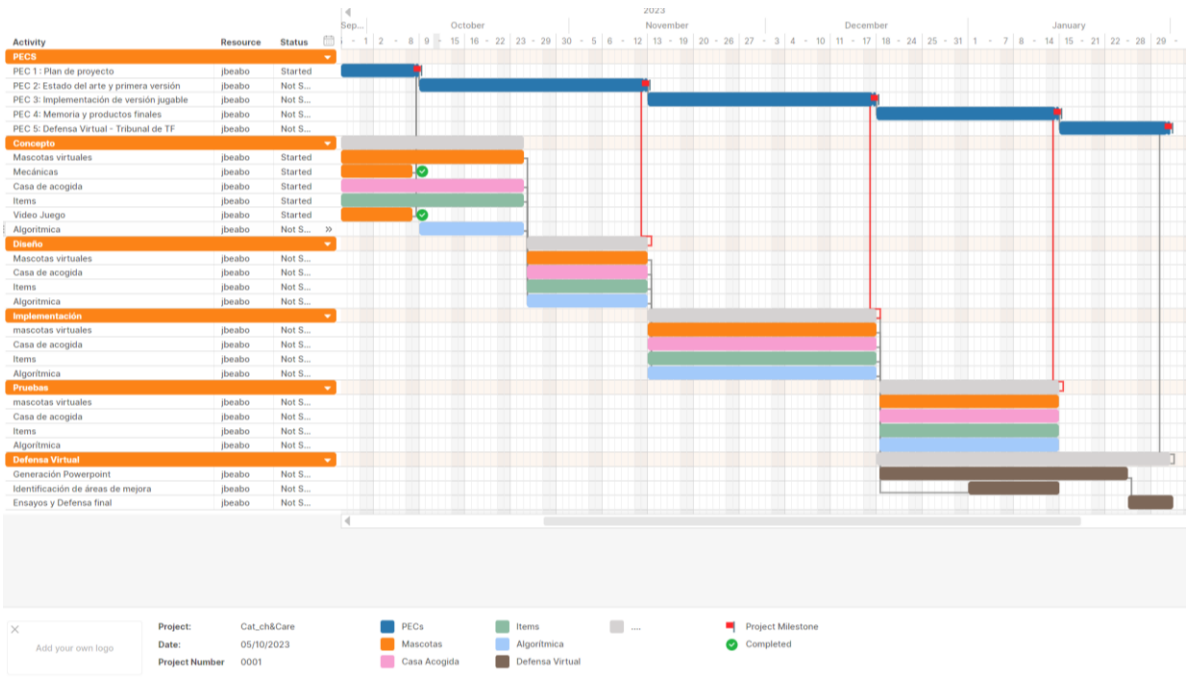


Figura 3: Diagrama Gantt del desarrollo del TFG al inicio del proyecto

Se han incluido las PECs de la asignatura como entregables–hitos durante el proyecto y se ha realizado una estimación de las duraciones y una evaluación de las dependencias entre tareas. En el sentido de las dependencias, he encontrado que Tom's planner, en su versión gratuita, es poco potente, ya que no permite definir dependencias como "finish to finish" o "start to finish" y simplemente se pueden modelizar dependencias "finish to start". Se han buscado dependencias de cada una de las fases (concepto, diseño, implementación, test y defensa) con cada una de las PECs de la asignatura. Se resume finalmente, en la Tabla 1, el tiempo invertido en cada una de las tareas identificadas:

Activity	Resource	Status	Start	End	Days	
PECS			**27-09-23**	**31-01-24**		**127.0**
PEC 1 : Plan de proyecto	jbeabo	Started	27-09-23	08-10-23	12.0	
PEC 2: Estado del arte y primera versión	jbeabo	Not S...	09-10-23	12-11-23	35.0	
PEC 3: Implementación de versión jugable	jbeabo	Not S...	13-11-23	17-12-23	35.0	
PEC 4: Memoria y productos finales	jbeabo	Not S...	18-12-23	14-01-24	28.0	
PEC 5: Defensa Virtual - Tribunal de TF	jbeabo	Not S...	15-01-24	31-01-24	17.0	
Concepto			**27-09-23**	**24-10-23**	**28.0**	**150.0**
Mascotas virtuales	jbeabo	Started	27-09-23	24-10-23	28.0	
Mecánicas	jbeabo	Started	27-09-23	09-10-23	11.0	
Casa de acogida	jbeabo	Started	27-09-23	24-10-23	28.0	
Items	jbeabo	Started	27-09-23	24-10-23	28.0	
Video Juego	jbeabo	Started	27-09-23	09-10-23	11.0	
Algoritmica	jbeabo	Not S...	09-10-23	24-10-23	16.0	
Diseño			**25-10-23**	**12-11-23**	**18.5**	**92.5**
Mascotas virtuales	jbeabo	Not S...	25-10-23	12-11-23	18.5	
Casa de acogida	jbeabo	Not S...	25-10-23	12-11-23	18.5	
Items	jbeabo	Not S...	25-10-23	12-11-23	18.5	
Algoritmica	jbeabo	Not S...	25-10-23	12-11-23	18.5	
Implementación			**13-11-23**	**17-12-23**	**35.0**	**175.0**
mascotas virtuales	jbeabo	Not S...	13-11-23	17-12-23	35.0	
Casa de acogida	jbeabo	Not S...	13-11-23	17-12-23	35.0	
Items	jbeabo	Not S...	13-11-23	17-12-23	35.0	
Algorítmica	jbeabo	Not S...	13-11-23	17-12-23	35.0	
Pruebas			**18-12-23**	**14-01-24**	**27.5**	**137.5**
mascotas virtuales	jbeabo	Not S...	18-12-23	14-01-24	27.5	
Casa de acogida	jbeabo	Not S...	18-12-23	14-01-24	27.5	
Items	jbeabo	Not S...	18-12-23	14-01-24	27.5	
Algorítmica	jbeabo	Not S...	18-12-23	14-01-24	27.5	
Defensa Virtual			**18-12-23**	**01-02-24**	**45.0**	**104.0**
Generación Powerpoint	jbeabo	Not S...	18-12-23	25-01-24	38.0	
Identificación de áreas de mejora	jbeabo	Not S...	01-01-24	14-01-24	14.0	
Ensayos y Defensa final	jbeabo	Not S...	25-01-24	01-02-24	7.0	

Tabla 1: Tareas desglosadas en el tiempo

1.5.1. Actualización

En la siguiente figura se muestra el progreso del proyecto a fecha de publicación de esta versión final de la memoria, para que pueda compararse con la planificación original. La principal diferencia ha sido que no se ha diseñado ninguna casa de acogida, pero en cambio se ha dedicado mucho tiempo al diseño del UI. También ha habido retraso en la generación de los algoritmos de AI de las mascotas virtuales, se ha implementado primeramente una máquina de estados, para el primer gato y un árbol de comportamiento para los siguientes.

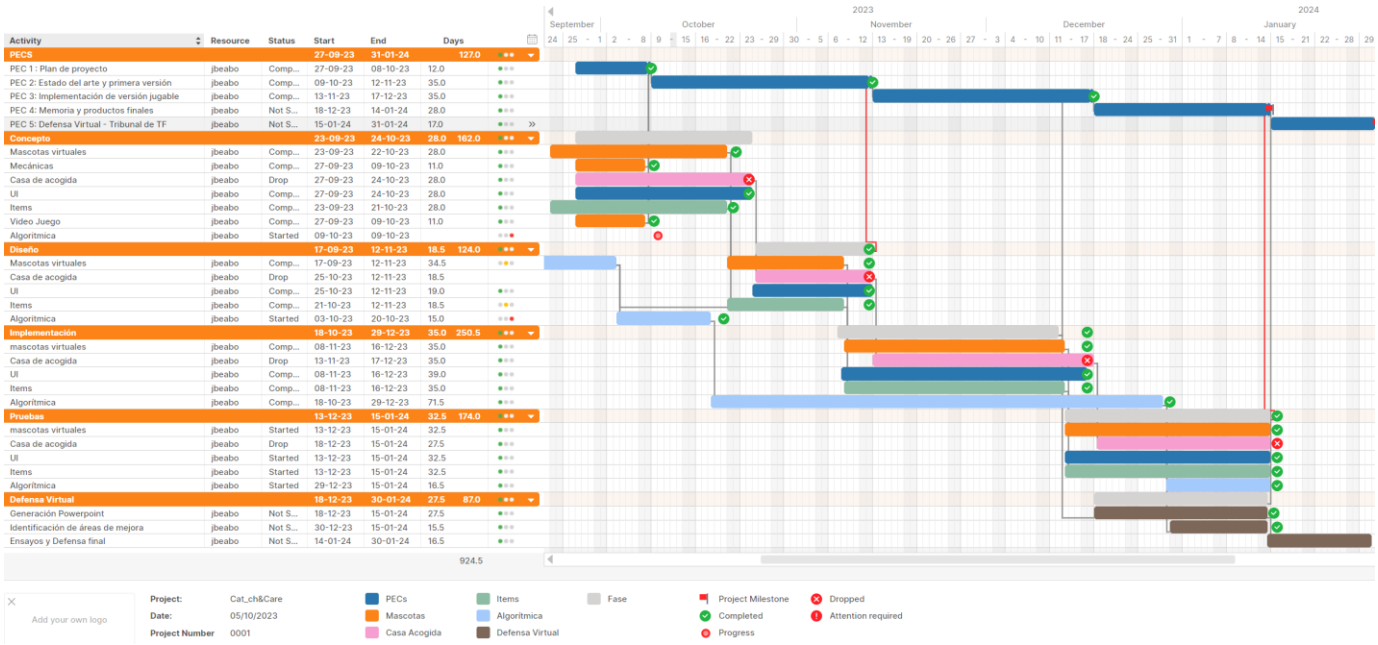


Figura 4: Diagrama Gantt actualizado del desarrollo del TFG

1.6. Presupuesto

El presupuesto, al igual que la memoria del proyecto o el cronograma de actividades, es un documento vivo, que evoluciona de la misma forma en que lo hace el propio proyecto, por tanto, el presente apartado se ha ido actualizando convenientemente. No se ha previsto la necesidad de adquirir nuevo equipamiento, utilizar recursos de terceros o licencias uso, por tanto, se muestra en la tabla 2, un presupuesto confeccionado en función de las horas dedicadas por el autor a desarrollar el TFG, a un precio estimado de 14€ por hora *[5]*.

Presupuesto Cat_ch&Care		Estimado por: Joaquín Bea Bonet		3/1/2024
		Duración horas		**Total Actividad**
TOTAL TFG		**450.0**		**€ 6,300.00**
Actividad	***Duración Días***	***Duración horas***	***Precio por hora***	***Total Actividad***
Concepto		**100.7**		**€ 1,409.33**
Mascotas Virtuales	28.0	18.7	€ 14.00	€ 261.33
Mecánicas	20.0	13.3	€ 14.00	€ 186.67
UI	28.0	18.7	€ 14.00	€ 261.33
Items	28.0	18.7	€ 14.00	€ 261.33
Video Juego	11.0	7.3	€ 14.00	€ 102.67
Algorítmica	8.0	24.0	€ 14.00	€ 336.00
	Duración Días	*Duración horas*	*Precio por hora*	*Total Actividad*
Diseño		**50.0**		**€ 700.00**
Mascotas Virtuales	10.0	6.7	€ 14.00	€ 93.33
UI	18.5	12.3	€ 14.00	€ 172.67
Items	12.0	8.0	€ 14.00	€ 112.00
Algorítmica	23.0	23.0	€ 14.00	€ 322.00
	Duración Días	*Duración horas*	*Precio por hora*	*Total Actividad*
Implementación		**150.0**		**€ 2,100.00**
Mascotas Virtuales	29.0	19.3	€ 14.00	€ 270.67
UI	35.0	23.3	€ 14.00	€ 326.67
Items	35.0	23.3	€ 14.00	€ 326.67
Algorítmica	42.0	84.0	€ 14.00	€ 1,176.00
	Duración Días	*Duración horas*	*Precio por hora*	*Total Actividad*
Pruebas		**110.0**		**€ 1,540.00**
Mascotas Virtuales	27.5	18.3	€ 14.00	€ 256.67
UI	27.5	18.3	€ 14.00	€ 256.67
Items	27.5	18.3	€ 14.00	€ 256.67
Algorítmica	27.5	55.0	€ 14.00	€ 770.00
	Duración Días	*Duración horas*	*Precio por hora*	*Total Actividad*
Defensa Virtual		**39.3**		**€ 550.67**
generación Powerpoint	38.0	25.3	€ 14.00	€ 354.67
Identificación áreas de mejora	14.0	9.3	€ 14.00	€ 130.67
Ensayos y defensa final	7.0	4.7	€ 14.00	€ 65.33

Tabla 2: Presupuesto estimado del prototipo del videojuego

El presupuesto anterior, se ha elaborado bajo la idea de desarrollar y entregar un prototipo de la versión de DEMO del videojuego, que puede suponer aproximadamente, un 20-25% del esfuerzo requerido para la implementación del producto final. Por tanto, si procediera calcular el **presupuesto del proyecto completo**, se podría suponer que el esfuerzo y coste requeridos, sería de unas 4-5 veces la cantidad calculada anteriormente y podría así **estimarse entre en unos 24,000€ y 30,000€.**

1.7. Estructura del resto del documento

Se describen brevemente, a continuación, los contenidos de cada capítulo y su relación con el trabajo en global.

Capítulo 1. Introducción: se explica y justifica el TFG, describiendo a grandes rasgos los objetivos, metodología, mecánica, planificación y presupuesto del videojuego.

Capítulo 2. Estado del arte y análisis de mercado: se realiza un estudio de la evolución de las mejores aplicaciones existentes en el ámbito de desarrollo del proyecto, así como de la situación actual del mercado en el que se enmarca el trabajo. Para hacerlo, se estudia la audiencia potencial y se analiza las soluciones ofrecidas por la competencia.

Capítulo 3. Propuesta: a partir del análisis de mercado realizado en el capítulo anterior, aquí se explica resumidamente la propuesta del TF, explicando las particularidades que lo diferencian de la competencia.

Capítulo 4. Diseño: se explica la arquitectura del videojuego y los detalles del producto resultante.

Capítulo 5. Implementación: se explican los detalles de la implementación y, de ser necesario, las instrucciones de instalación.

Capítulo 6. Demostración: en este capítulo se incluyen las instrucciones y las capturas de pantalla necesarias para mostrar la instalación y operación del videojuego, incluyendo las pruebas realizadas y la guía de usuario con ejemplos de uso del producto.

Capítulo 7. Conclusiones y líneas de futuro: como se indica, se extraen conclusiones de la realización del TFG y se exponen las posibles líneas de investigación o desarrollo futuro que se han considerado durante el desarrollo de este.

Bibliografía: listado de citas y referencias bibliográficas registradas en orden de aparición en el TFG

Anexos: listado de apartados complementarios adicionales.

2. Estado del arte y análisis de mercado

En este capítulo se realizará un estudio de la evolución y estado de las aplicaciones desarrolladas en el ámbito de este trabajo de fin de grado, así como un análisis de las mejores tecnologías de desarrollo existentes en el mercado.

2.1. Público objetivo y perfiles de usuario

Este videojuego está dirigido a todos los públicos, sea cual sea su edad, identidad de género y condición social. En lo referido a la accesibilidad, se facilitará la proporcionada por cada sistema operativo y dispositivo *Smart Phone,* donde se instale la aplicación. Si bien es cierto que se requerirá cierto nivel de coordinación motriz para realizar tareas esenciales como, por ejemplo, dar de comer o acariciar a la mascota virtual, la aplicación se diseñará de forma que no sea necesario tener habilidades precisas en ese sentido.

2.2. Marco teórico y estado del arte

2.2.1. Ámbito del juego

Como se ha explicado en la introducción, la experiencia de juego se basa en el coleccionismo y en crear una historia única para cada gato virtual. En este sentido se puede encontrar un antecedente claro, el juego Tamagotchi de Bandai *[6].*

Tamagotchi: creado por Aki Maita, una desarrolladora japonesa que trabajaba en Bandai, fue lanzado al mercado el 23 de Noviembre de 1996. Se trataba de un aparato electrónico que cabía en el bolsillo, contaba con tres botones y una pantalla LCD en blanco y negro. El usuario utilizaba los botones para proporcionar cuidados constantes a la mascota virtual, con el fin de que esta creciera sana y feliz, evitando la muerte. Al poco tiempo de lanzarse, Tamagotchi se convirtió en un fenómeno cultural a nivel mundial, que presentaba por primera vez el concepto de “responsabilidad digital”. Desde entonces se han introducido 47 diferentes versiones, creando un auténtico mercado de “mascotas virtuales”.

Bajo el paraguas de este nuevo mercado, han nacido y crecido los siguientes desarrollos:

Digimon: mascota virtual creada por Akiyoshi Hongo en 1997 *[7]*, también de Bandai. Como diferenciador con el anterior, el juego permitía luchas entre criaturas y usuarios. Nacieron, como los Tamagotchi, en forma de aparato electrónico y se expandieron a series de televisión, películas, *anime*, revistas, libros, *manga*, videojuegos y *merchandising*.

Figura 5: Modelo del Virtual Pet Digimon, de Bandai, fotografía extraída de https://www.es.wikipedia.org el 13 Oct 2023

Petz: videojuego para computador, actualmente comercializado por Ubisoft *[8]*, que vió la luz por primera vez en el año 1995 de la mano de P.F.Magic, en el que el usuario puede adoptar y cuidar a diversas mascotas virtuales (Dogz, Catz, Bunniz, Hamsterz, …).

Neopets: creados el 15 de Noviembre de 1999 *[9]* por Adam Powell, se trata de un sitio web de mascotas virtuales, totalmente gratuito y para todos los públicos. En él los usuarios pueden cuidar de múltiples criaturas, interactuar con otros usuarios, participar en juegos y en diferentes actividades que se desarrollan dentro del propio dominio de internet.

Figura 6: sitio web de Neopets, fotografía extraída de https://www.neopets.com el 13 de Octubre de 2023

Nintendogs: videojuego lanzado en el año 2005 *[10]* para la consola Nintendo DS, la aplicación permite al usuario adoptar y cuidar (alimentar, bañar, entrenar y competir con otros usuarios) a cachorros virtuales. Como novedad, se utiliza una pantalla táctil y el micrófono de la cónsola para interactuar con las mascotas. También se vale del reloj interno de la DS para parametrizar cuándo el perro crecerá y simulará estar hambriento o necesitará un baño. Nintendogs es considerado como un juego "no-juego" ("*non-game game*"), término acuñado por el presidente de Nintendo Satoru Iwata, que describe una forma de entretenimiento donde

no existe un ganador ni se llega a un final. Esto permite al usuario desarrollar libremente su propia forma de juego y establecer sus propios objetivos y retos personales.

Webkinz: creados originalmente por la empresa de juguetes canadiense Ganz *[11],* se lanzaron por primera vez al mercado el 15 de Abril del 2005. La Novedad consiste en lincar un peluche físico con un sitio web. Cada muñeco Webkinz lleva una etiqueta con un código único que permite a su dueño jugar con la versión digital de su peluche desde un dispositivo IOS, Android o PC. A pesar de que inicialmente sólo se comercializó en EEUU y Canadá, en el año 2006 ya contaba con más de 1 millón de jugadores registrados. En el mundo virtual, los usuarios pueden ganar dinero adoptando nuevas mascotas, jugando en línea, respondiendo a preguntas de cultura general o realizando las actividades (mini-juegos) que se proponen a diario. Con cada peluche comprado, se añade más dinero y artículos virtuales a la cuenta del usuario. Con el dinero virtual se pueden comprar y coleccionar artículos como, por ejemplo, gemas, estrellas o libros de cocina.

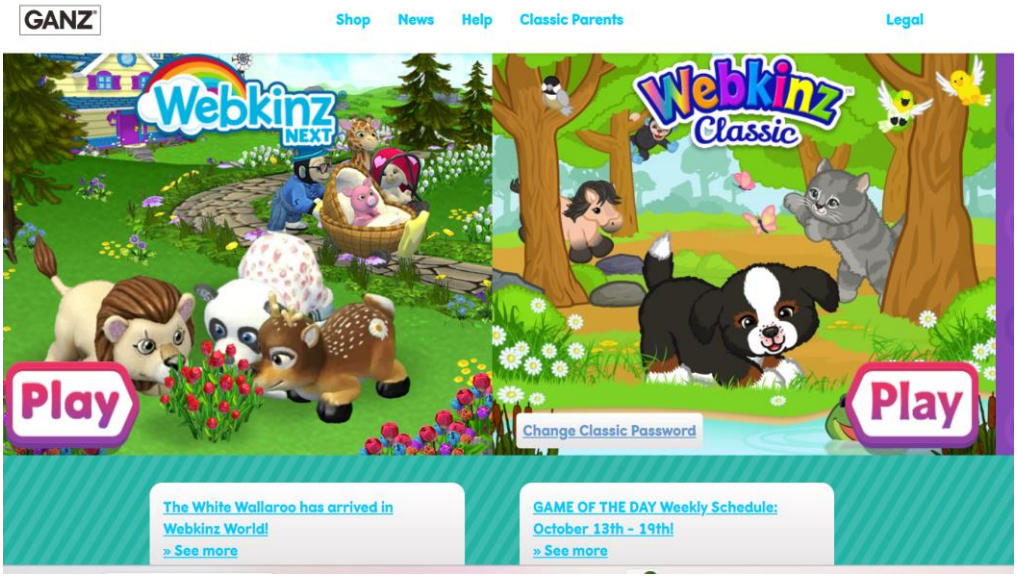


Figura 7: sitio web de Webkinz, fotografía extraída de https://www.webkinz.com/ el 13 de Octubre de 2023

My Talking Tom: *[12]* es una aplicación para IOS y para Android en la que el jugador debe adoptar y cuidar de un gato virtual (Tom) que habla, a la par que debe coleccionar complementos de moda y decoración. Fue desarrollada y lanzada al mercado el 11 de Noviembre de 2013 por Outfit7. La aplicación fue objeto de denuncia ante la ASA (UK Advertising Standards Authority) por ofrecer publicidad para adultos no apropiada para público infantil.

A la aplicación anterior para móviles han seguido varias de un estilo parecido, culminando en la entrega, el 9 de Mayo de 2023 de Peridot *[4]*.

Peridot: se trata de un juego de mascotas virtuales en Realidad Aumentada (AR) publicado por Niantic para equipos IOS y Android. En este caso, cada Peridot es único y el usuario es su cuidador desde el nacimiento hasta su edad adulta, encargándose de alimentarlo, jugar, acariciar y explorar los alrededores, estableciendo de esta forma vínculos con él o ella.

Figura 8: sitio web Peridot, fotografía extraída de https://playperidot.com/es el 14 de Octubre de 2023

2.2.2. Herramientas de desarrollo

El software AR debe ser instalado, generalmente, en dispositivos del tipo tabletas y teléfonos inteligentes que contienen sensores, cámaras, pantallas táctiles y software, permitiendo integrar o proyectar objetos digitales en el mundo real e interactuar con ellos.

Estas soluciones tienen diversos usos, que van desde el marketing, la educación y la medicina, a aplicaciones de reparación y mantenimiento, o incluso de venta minorista, además de utilizarse con fines de entretenimiento, en el ámbito de los videojuegos.

Como se trata de un campo relativamente reciente, las herramientas de desarrollo se están construyendo al tiempo que se aplican. Existen actualmente varias en el mercado que, o bien se han creado expresamente para desarrollar AR, o bien han adaptado sus plataformas para integrar la AR o, de una forma más amplia, la Realidad Virtual (VR) en sus motores de desarrollo.

Actualmente destacan tres herramientas, de entre las disponibles en el mercado para desarrollo de AR: Vuforia *[13]*, Unity *[14]* y Unreal *[15].*

Unity: se trata de uno de los motores de juego más utilizados para crear videojuegos, aplicaciones industriales, educativas y de marketing, tanto en juegos tradicionales como de realidad Virtual para diferentes plataformas (IOS, Android, PlayStation, …). Está ampliamente adoptado por la industria y ofrece gran variedad de *assets* que pueden ser adquiridos o utilizados libremente en el desarrollo de las aplicaciones. Integra un módulo de Vuforia con el que poder desarrollar y mejorar experiencias de Realidad Virtual y Realidad Aumentada. Su uso es gratuito para estudiantes o profesionales con ingresos anuales inferiores a 100,000 $.

Unreal: es otro poderoso motor de desarrollo que ofrece un conjunto completo de herramientas de VR y AR para desarrolladores. Al igual que Unity, Unreal se utiliza no sólo para desarrollar videojuegos, si no también para otras industrias, como la del cine, arquitectura, automoción, etc. Unreal es también gratuito si el producto generado con la plataforma no supera un millón de dólares de ingresos. Desde el punto de vista de calidad de gráficos 3D, Unreal es superior a Unity, pero tiene una curva de aprendizaje mucho más pronunciada, al no ser tan intuitivo, lo cual hace que resulte muy complejo para principiantes. Probablemente esto sea motivo suficiente para descartarlo como herramienta en este TFG, en el que se dispone sólo de 4 meses para implementar el videojuego.

Vuforia: es una plataforma de desarrollo de aplicaciones de AR y Realidad Mixta (MR) multiplataforma y multidispositivo (gafas AR / MR, SmartPhones, Gear VR…). Es una plataforma gratuita para prototipado, pero es de pago cuando se pretende saltar a producción. La integración del motor de Vuforia en Unity permite utilizar la IA de visión por computador desarrollada por Vuforia para crear aplicaciones y juegos en Unity para IOS y Android.

Dado que, de ser necesario, se puede integrar el motor de Vuforia en Unity y que Unity tiene una curva de aprendizaje más suave que Unreal, parece indicado utilizar Unity como motor de desarrollo de AR en este proyecto. Al definir el alcance del proyecto, se ha tenido en cuenta que deberá dedicarse una parte considerable de tiempo al aprendizaje de Unity.

3. Propuesta

En este capítulo se explica, de manera resumida, la propuesta del TF, haciendo énfasis en las particularidades que lo diferencian de la competencia.

3.1. Descripción del juego

La experiencia de juego se centra en conseguir motivar e implicar al usuario a través, principalmente, de las siguientes estrategias [2]:

- **Una historia única** para cada gato. En el próximo apartado, veremos que el objetivo del jugador es crear un vínculo afectivo con el gato virtual, de forma que nuestra mascota se sienta feliz y prefiera vivir con la persona que le cuida, a la posibilidad de escapar y regresar a la calle. La forma y los medios utilizados para establecer ese vínculo vendrán determinados por el usuario y dependerán de la personalidad de cada gato virtual, de los objetos disponibles para interactuar en ese instante y, finalmente, del tiempo y frecuencia de las interacciones. Por tanto, la historia de cada gato será única y, de alguna forma, será creada "sobre la marcha" por cada usuario.
- **Coleccionar,** tanto gatos como ítems. Por un lado, los ítems (comida, jaulas atrapa gatos, juguetes para gatos…) junto con las caricias, serán la base de la interacción entre la mascota virtual y el jugador. Cuantos más ítems se posean, más fácil será establecer un vínculo con el gato de forma que este se sienta feliz y decida no huir. Cabe mencionar que existirán ***ítems* especiales** que podrán, por ejemplo, convencer a un gato tímido para que se alimente o a un gato temeroso para que juegue. Por otro lado, una vez el gato haya establecido el vínculo máximo con el usuario, habrá decidido quedarse con él y podrá ser añadido a la galería de gatos adoptados.
- ***Feedback*** instantáneo. Es fundamental que el usuario sepa, en todo momento, si está progresando adecuadamente con cada mascota virtual, para ello, se crearán medidores de felicidad felina y amistad en tiempo real y un monitor del nivel de vínculo alcanzado entre el gato y el usuario.

3.2. Descripción de la pantalla principal

La trama del juego gira alrededor de establecer un vínculo afectivo con el gato virtual, de forma que la mascota se sienta feliz y prefiera vivir con la persona que la cuida, a la posibilidad de escapar y regresar a la calle. Cuando el nivel de vínculo sea máximo, la mascota virtual (MV) podrá añadirse a la galería de colección.

Como se trata de una parte fundamental de la experiencia de juego, los medidores de felicidad (*Happiness*), amistad (*Friendship*) y vínculo (*Bond*) se encontrarán siempre visibles en la parte superior de la pantalla principal. También lo estarán el contador del nivel de "casa de acogida" (*Level*) y el número de monedas conseguidas (*Euro Cats),*

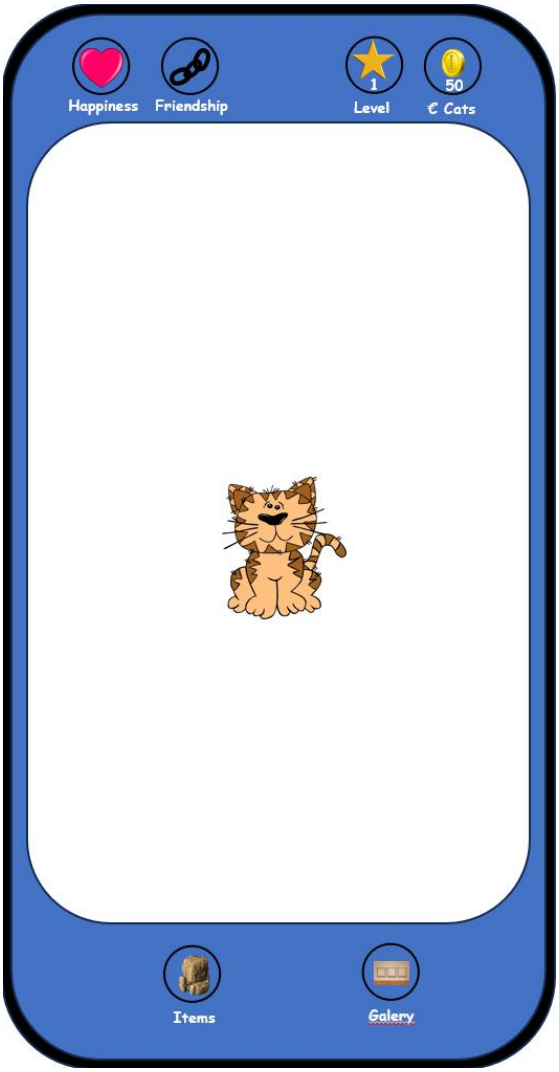


Figura 9: En la parte superior de la aplicación, medidores de Happiness, Friendship, Level y €-Cats. Abajo, los botones de juego.

También en la pantalla principal, pero en la parte inferior, se podrán encontrar los botones de interacción necesarios para realizar las diferentes actividades del juego:

- **Botón *Gallery***: al presionarlo se accederá a la galería de gatos adoptados, que al empezar el juego se encontrará vacía.

- **Botón *Items***: al presionarlo se accederá a un menú que contiene todos los accesorios del juego, tanto los disponibles como los que pueden ser comprados. Al inicio, el único *ítem*

disponible será una jaula atrapa gatos. Al ser tocada por el usuario, la jaula aparecerá en la escena, junto a la MV. El resto de *los ítems*, que tendrán como "cantidad disponible" = 0, deberán ser comprados por el usuario utilizando los *€ Cats* disponibles.

Figura 10: El inventario de accesorios aparece al tocar el botón Items.

Dentro del menú *Items*, encontraremos objetos que habilitarán funcionalidades como son alimentar o jugar.

Alimentar al gato: tras haberlo comprado, al presionar sobre el bol de alimento, el usuario podrá colocar en la escena un bol virtual con comida para gatos. Si el gato está hambriento, acudirá al bol para alimentarse y saldrán volando corazoncitos desde el gato virtual, que harán crecer los indicadores de *Happiness* y de *Friendship*. Cada vez que se utilice un bol, se descontará una unidad de boles en el menú de ítems. Super comida: los gatos tímidos sólo aceptarán super comida.

Figura 11: Alimentando a la mascota virtual.

Jugar con el gato: al tocar alguno de los juguetes que ha sido adquirido, el usuario podrá posicionarlo en escena, o moverlo, para atraer la atención de la mascota. Si el gato decide jugar, saldrán volando corazoncitos que harán crecer los indicadores de *Hapiness* y de *Friendship*. Al iniciar la aplicación, en el inventario no se encontrará disponible ningún juguete. Super juguetes: los gatos miedosos sólo aceptarán jugar con Super Juguetes.

Figura 12: Jugando con la mascota virtual.

Acariciar al gato: al mover el dedo sobre el gato en la pantalla del dispositivo, se acariciará virtualmente al gato y saldrán volando corazoncitos, desde el gato virtual, que harán crecer los indicadores de *Happiness* y de *Friendship*. No habrá límite en el número de caricias.

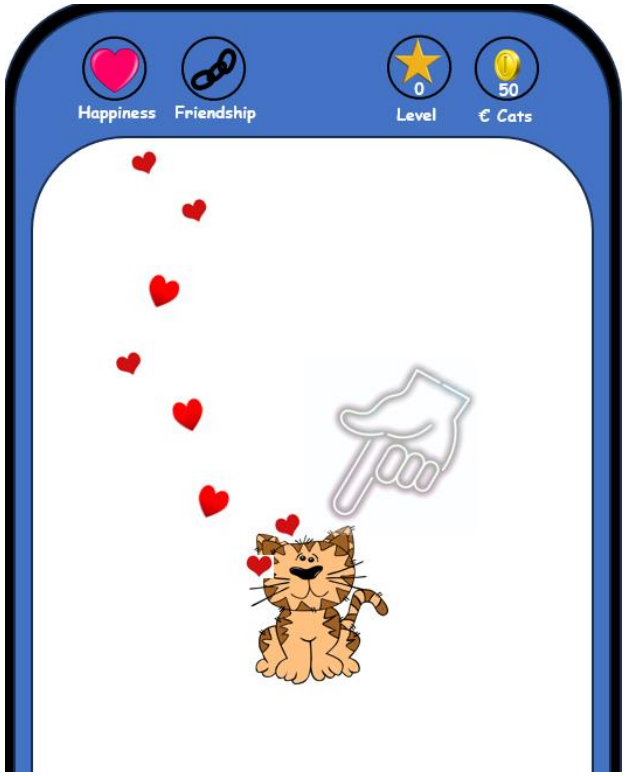


Figura 13: Acariciando a la mascota virtual.

3.3. Trama o *storyboard*

Por tratarse de un juego de RA, al arrancar la aplicación, aparecerá un *pop-up* para que el usuario permita el acceso a la cámara de su *Smart Phone*.

Desde la pantalla principal, indicada en la Figura 9, se podrá acceder tanto a la Galería de adopciones (*Gallery*) que se encontrará vacía al inicio, como a la página de inventario (*Items*). Se empezará el juego con sólo una jaula atrapa gatos en el inventario, el resto de *Items* deberán ser adquiridos antes de poder utilizarlos, para ello se contará al inicio con 100 monedas (Euro-Cats).

Después de permitir un tiempo para que el usuario se familiarice con la aplicación, la cámara empezará a detectar planos horizontales que aparecerán en pantalla. Cuando el usuario toque en un punto de uno de los planos, aparecerá un gato feral con el que usuario deberá estrechar vínculo. Cada gato virtual se comportará de forma diferente, por ejemplo, un gato cariñoso será dócil y facilitará la interacción, mientras que un gato tímido o asustadizo necesitarán el uso de Super comida o Super juguetes (respectivamente) para poder interaccionar con ellos. De una forma u otra, a través de la interacción se deberá conseguir ir aumentando el nivel de vínculo con el gato virtual. Cuando se consiga el máximo nivel de

vínculo, se podrá lanzar a escena una jaula atrapa gatos. Si la nueva mascota se introduce en la jaula, ¡habremos logrado convertirnos en casa de acogida!

Aparecerá un *pop-up* recompensando la acción en forma de monedas (+100) y convirtiendo al jugador en "casa de acogida" (CA) de primer nivel. Por tanto, se incrementará el valor de *Level* y se actualizará el contador de *€ cats* en la pantalla principal, como se indica en la Figura 14.

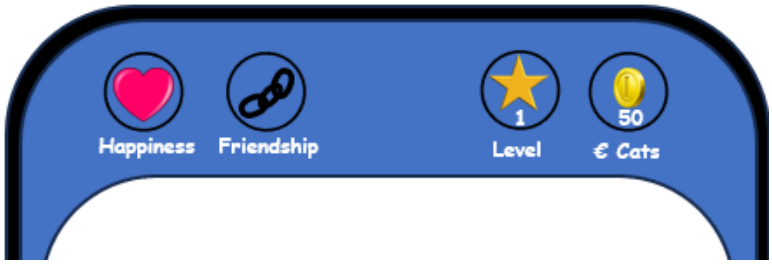


Figura 14: En la parte superior derecha, medidor de nivel de CA.

Una forma de progresar en el mundo de Cat-ch&Care será aumentar el nivel de CA, que crecerá en función del número de gatos que se vayan adoptando. Se iniciará el juego con nivel 0, pero al vincularse con el primer gato, el jugador se convertirá en "albergue de acogida" de nivel 1 (persona principiante), para finalmente llegar al último nivel, "hotel de acogida" de nivel 5 (persona veterana). Para subir de CA, en esta versión DEMO, se deberá haber adoptado a un gato en ese nivel. Cuando se consiga, se ganará un número determinado de monedas, con las que podrán comprarse nuevos ítems especiales para poder estrechar más fácilmente vínculos con el resto de gatos. De forma que, al ir subiendo de categoría, será más fácil conseguir estrechar vínculos con la nueva mascota, pero en cambio, los gatos que se vayan encontrando, cada vez serán más difíciles de vincular.

Se resumen los niveles de esta versión DEMO en la Tabla 3:

Nivel	**Monedas ganadas al acceder al nivel**	**Número de adopciones (acumulado) necesarias para acceder al nivel**
1	100	1
2	100	2
3	100	3
4	100	4
5	100	5

Tabla 3: Resumen de los niveles de casa de acogida de la DEMO del videojuego

El usuario podrá estrechar el vínculo con el gato virtual a través de:

- Alimentar de forma periódica.
- Acariciar regularmente.
- Jugar a menudo.

Cada vez que se culmine la adopción de una MV se conseguirán 100 monedas.

3.4. Definición de las mascotas virtuales

Una de las partes que requiere más recursos en el desarrollo de un videojuego, es el diseño de los personajes, del escenario y de los accesorios del juego. Por limitaciones claras de tiempo, de falta de competencia en la materia de diseño de modelos 3D del autor del TF y para permitir focalizar el mismo en la IA del juego, se utilizarán modelos 3D ya desarrollados, bajo licencia de uso y gratuitos, disponibles bien en la biblioteca de *assets* de *Unity*, llamada *Unity Asset Store*, en https://assetstore.unity.com/, bien en la biblioteca de *assets* para *Unity* de SketchFab, en https://sketchfab.com/

Se han encontrado en el Unity Asset Store, tres modelos que podrían cumplir con los requisitos del TFG:

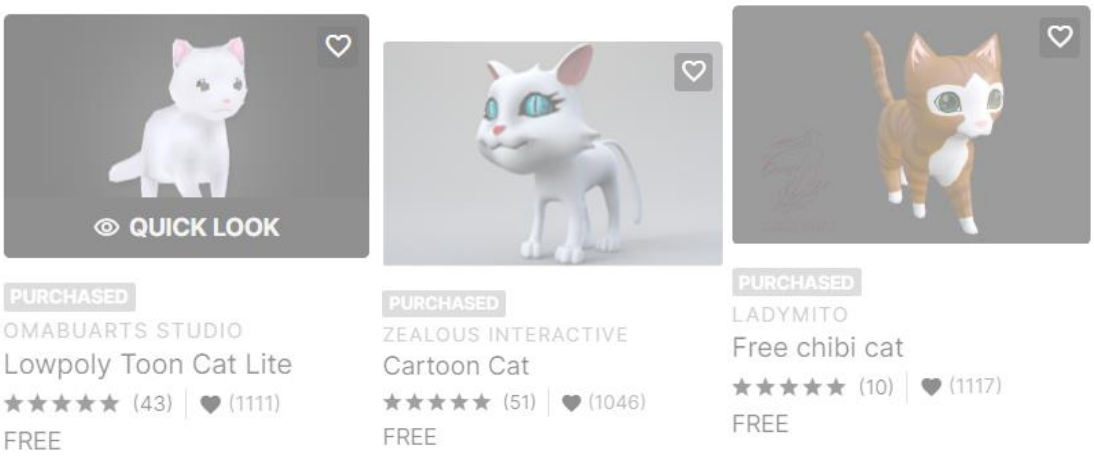


Figura 15: Modelos de gato extraídos del Asset Store de Unity.

De los tres modelos, el único que contiene un número aceptable de animaciones es el segundo, Cartoon Cat. Se ha utilizado este modelo para toda la versión de demo, modificando el color de los gatos instanciados en cada nivel.

4. Diseño

Se explican, en este capítulo y en los sucesivos, los detalles del Videojuego.

4.1. Arquitectura general de la aplicación

La arquitectura de esta aplicación está basada en un patrón de eventos, donde tenemos *publishers* (uno o más eventos, o uno o más estados) y *subscribers* (una parte, uno, o varios *scripts* C#), cuando se llama a un evento, este notifica a los correspondientes *subscribers* para que sean ejecutados.

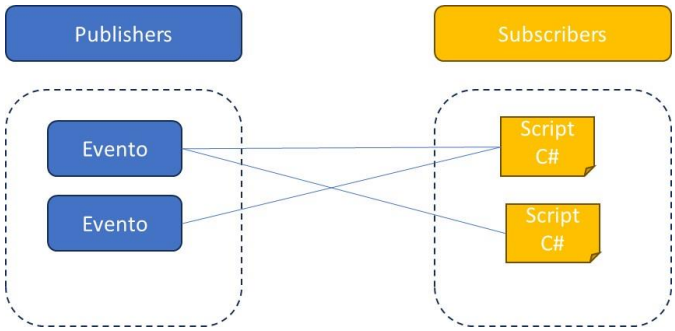


Figura 16: Relación entre eventos y Scripts.

En el juego tendremos ocho tipos de evento:

- **Menú de bienvenida,** *Welcome* muestra los botones principales y destaca el objetivo del juego.
- **Main Menu**: *MainMenu,* menú principal desde el que se inicia la aplicación y permite acceder al resto de estados.
- **Menú de Galería**: *GalleryMenu,* donde podemos observar los gatos coleccionados
- **Menú de Items**: *ItemsMenu,* donde podemos observar los ítems adquiridos o comprar los que estén disponibles.
- **Menú de posicionamiento de los Items**: *ARPosition,* hemos comprado un Item y queremos posicionarlo en escena para interactuar con el gato virtual.
- **Menú de compra de los Items**: *Shop,* queremos utilizar monedas para comprar uno de los *ítems* del inventario que tienen cantidad 0.
- **Menú para descartar compra**: *NoShop*, el usuario no tiene suficientes €cats para adquirir el artículo elegido.
- **Menú de adopción**: *Adoption,* debemos decidir si adoptamos o no al gato virtual.
- **Menú de adotado**: *Adopted,* banner de felicitación por haber adoptado a un gato.

- **Menú de Fín de Demo**: GameOver, indica el fin de la demo y permite cerrar la aplicación.

Los eventos anteriores forman parte del *GameManager*, un script encargado de direccionar cada evento al script o la parte del script al que esté subscrito:

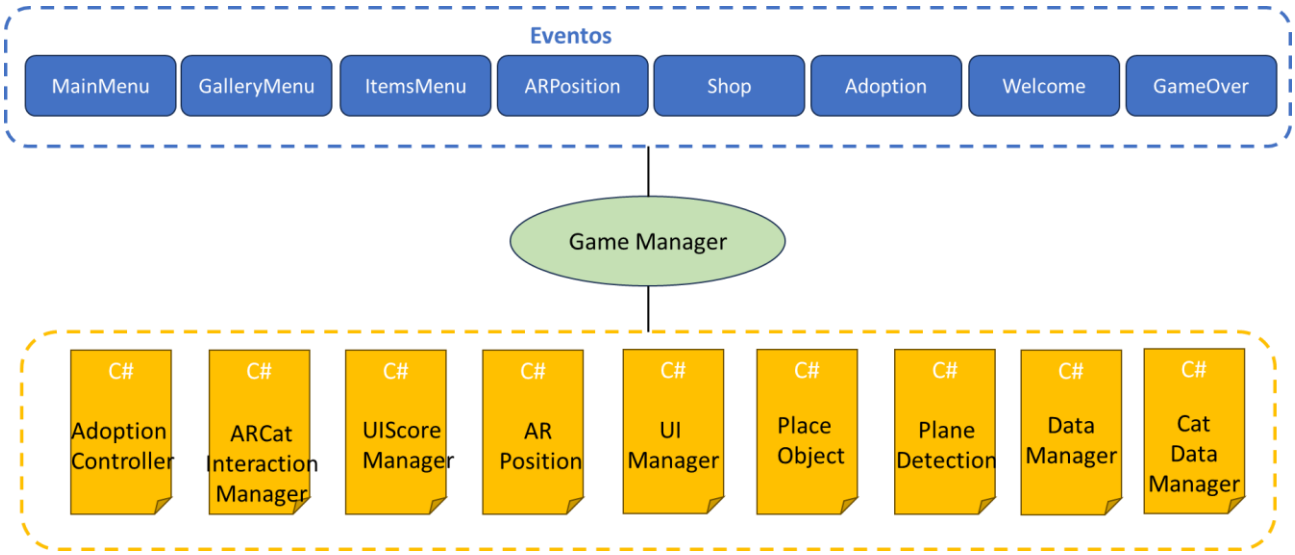


Figura 17: Función del Game Manager

4.1.1. Gestión del UI:

Se utiliza un Script para gestionar los menús de interacción con el usuario, llamado ***UIManager***, el cual contiene los menús de usuario de los diferentes estados del juego:

- *WelcomeCanvas*: con una explicación de los iconos de marcador, UI y una breve sinópsis del objetivo del juego.
- *MainMenuCanvas*: con los iconos y UI del menú principal.
- *GalleryMenuCanvas*: conteniendo el UI de la galería donde se muestran las mascotas adoptadas.
- *ItemsMenuCanvas*: donde se encuentran los Items adquiridos y que pueden adquirirse en el juego.
- *ARPositionCanvas*: con los iconos para posicionar en escena el item elegido.
- *ShopCanvas*: con los iconos y la lógica para comprar los *Items* disponibles.
- *NoShopCanvas*: donde se encuentra el UI que aparece cuando no se dispone de suficiente dinero como para comprar un determinado *Item*.
- *AdoptionCanvas*: donde se encuentra el UI necesario para adoptar a un gato.
- *AdoptedCanvas*: donde se encuentra el UI que aparece después de adoptar a un gato.

- *GameOverCanvas*: donde se encuentra el UI que aparece cuando se alcanza el nivel 5 de esta versión de DEMO.

De manera análoga, se utiliza un Script para gestionar los menús de puntuación, llamado ***ScoreManager***:

- HapinessScore: con el icono y el estado del nivel de *Hapiness* de la mascota virtual.
- FriendshipScore: donde se encuentra el icono que indica el nivel de amistad con el gato.
- LinkWheel; marcador en forma de donut que indica el nivel de vínculo con la mascota virtual.
- LevelScore: conteniendo el icono que indica el nivel de Casa de Acogida actual.
- eCatsScore: con el número de monedas conseguidas.

Figura 18: Contadores del ScoreManager

4.2. Arquitectura de la información y diagramas de navegación

Para almacenar los GameObjects que contienen los Items del juego se utilizan *ScriptableObjects*. Se trata de contenedores de datos que pueden ser utilizados para guardar grandes cantidades de datos, independientemente de las instancias de clase. Una de las ventajas de utilizar *ScriptableObjects,* es la de reducir el uso de memoria por parte de la aplicación, al evitar duplicar valores. Se utilizan los *ScriptableObjects* como un *Prefab* que almacena datos inmutables dentro de los scripts. Cada vez que se instancia un *Prefab*, se genera una copia de los datos sin almacenarlos por duplicado, de forma que sólo existe una copia de los datos en memoria. En este proyecto, definiremos el *ScriptableObject Item* con los siguientes atributos: nombre, imagen (2D), precio, cantidad disponible y modelo (3D). Y los *ScriptableObject Cat* con los atributos nombre, imagen (2D) y modelo (3D).

Se han definido e implementado los siguientes *scripts* en *C#*:

ARCatInteractionManager: script que utiliza Raycast para saber si el usuario está interaccionando con la mascota virtual y proporciona la funcionalidad de acariciar. Contiene las funciones:

- `public bool isTapOverUI(Vector2 touchPosition)`: comprueba si el usuario ha presionado sobre un botón de UI.
- `public bool isTapOver3DCat(Vector2 touchPosition)`: comprueba si el usuario ha presionado sobre el gato virtual.

ARInteractionManager: script que utiliza Raycast para posicionar los ítems en el punto que aparece en el centro de la pantalla para que luego el usuario los pueda rotar y posicionar a su antojo. Contiene las funciones:

- `private void SetItemPosition()`:posiciona el modelo 3D seleccionado.
- `public void DeleteItem()`: borra el item seleccionado de la escena.
- `public bool isTapOverUI(Vector2 touchPosition)`: comprueba si el usuario ha presionado sobre un botón de UI.
- `public bool isTapOver3DModel(Vector2 touchPosition)`: comprueba si el usuario ha presionado sobre un modelo 3D.

Cat: *ScriptableObject* que contiene la información de los gatos que se instancian en escena. Contiene los campos:

```
public string CatName;
public Sprite CatImage;
public GameObject Cat3DModel;
```

CatButtonManager: se trata de un *script* que permite instanciar y asignar valores a los botones que aparecerán en el menú de *Gallery* (*GalleryMenuCanvas*). Implementa la clase *CatButtonManager* con los *setters* de los atributos definidos.

CatDataManager: script que crea botones en la galería de gatos y asigna valores a cada uno de ellos utilizando los setters del CatButtonManager descritos anteriormente. Contiene la función:

- `private void CreateButtons()`: instancia los botones de los gatos de Gallery.

DataManager: script que contiene la lista de objetos para los que hay que crear botones y asigna valores a cada uno de ellos utilizando las funciones del *ItemButtonManager* descrito anteriormente. Contiene la función:

`private void CreateButtons()`: instancia los botones de los Items.

Item: scriptableobject que contiene la información de los *Items* que se instancian en escena. Contiene los campos:

```
public string ItemName;
public Sprite ItemImage;
public string ItemPrice;
public string ItemQuantity;
public GameObject Item3DModel;
```

ItemButtonManager: se trata de un *script* que permite instanciar y asignar valores a los botones que aparecerán en el menú de *Items* (*ItemsMenuCanvas*). Implementa la clase *ItemButtonManager* con los *setters* de los atributos definidos y añade la función de instanciar el modelo 3D elegido cuando se presiona el botón del *item*. También implementa la lógica necesaria para lanzar el menú de Shop si el Item no se encuentra disponible. Contiene las funciones:

- `private void Create3DModel ()`: instancia el modelo 3D seleccionado.
- `private void GrowButton ()`: si la cantidad disponible es 0, hace que el botón parpadee al ser presionado, abriendo el menú de Shop.
- `public void ConfirmBuy ()`: Confirma la compra, restando el precio del item de la cantidad dispoinible de Ecats.
- `public void NotConfirmBuy ()`: sale del menú sin realizar cambios.

Controller: script que controla al primer gato del juego. Contiene las funciones:

- `float CalculateDistance(GameObject gObj):` calcula la distancia entre el GameObject Input y el gato.
- `public void AdoptAndDestroyCat():` instancia corazones y destruye el GameObject gato que ha sido instanciado.
- `public void DestroyCat():` destruye el GameObject gato que ha sido instanciado.
- `private void SearchForFood():` Hace que el gato busque comida.
- `private void SearchForToy():` Hace que el gato busque juguetes.
- `private void SearchForCage():` Hace que el gato busque una jaula.
- `private void SearchForRandom(): NO IMPLEMENTADA.` Hace que el gato busque una posición aleatoria dentro de la escena.

Y se complementa con la siguiente máquina de estados:

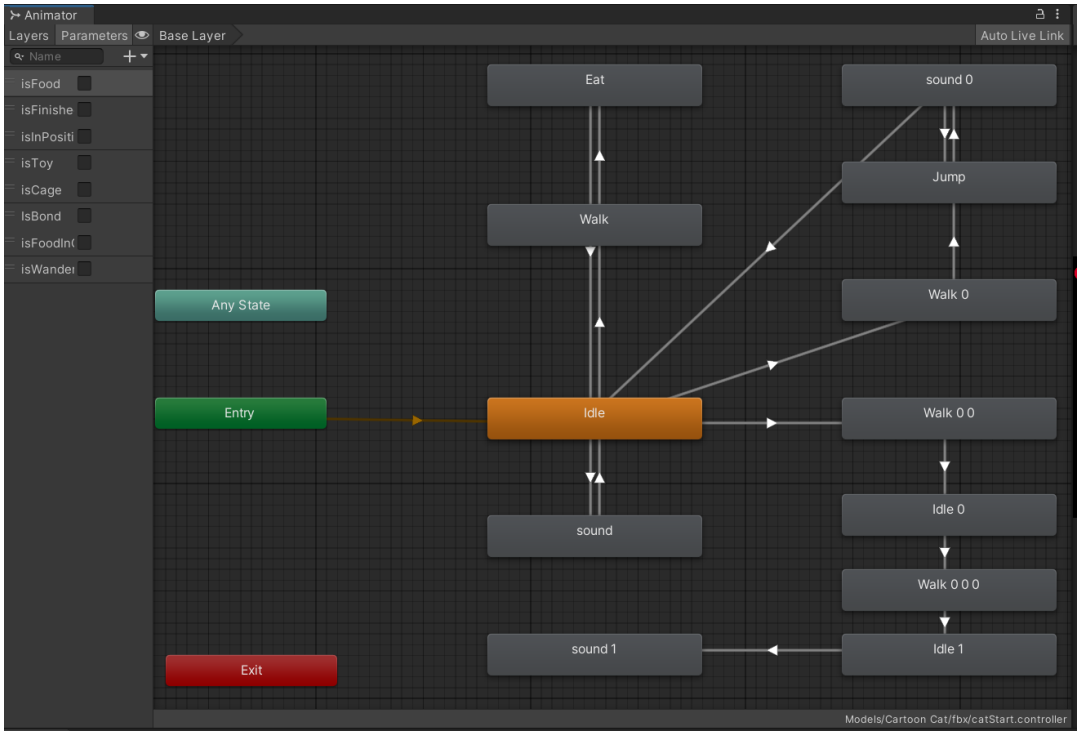


Figura 19: Máquina de estados de Controller.cs

PlaceObject: script que permite posicionar a la mascota virtual. Contiene la función:

- `private bool TryGetTouchPosition(out Vector2 touchPosition)`: registra el punto en que el usuario toca la pantalla y devuelve true si registra la posición.
- `public Cat GetCat()`: retorna el scriptable object que contiene el gato que se encuentra instanciado en escena. Es utilizado por el script *CatDataManager* para generar los botones del menú de gallery.

UIManager: se trata de un *script* que instancia los menús de UI, implementando las diferentes funciones que activan cada uno de los menús descritos con anterioridad.

UIScoreManager: script que actualiza los contadores del juego al alimentar, acariciar o jugar con el gato. Contiene las funciones:

- `private void LinkWheelFiller()`: rellena el donut de progreso del vínculo (Bond).
- `private void ColorChanger()`:cambia el color, de rojo a verde, del donut de progreso del vínculo.
- `public void AddScore(float points)`: añade puntos a los contadores de happy, friend y link.
- `public void AddEcats(int addEc)`: añade monedas al contador de €Cats.

- `public bool EnoughEcats(string price)`: true si hay suficientes €Cats en el Contador para comprar el Item seleccionado.
- `public bool IsLink()`: Devuelve true si el marcador de LINK (Bond) se encuentra al máximo.
- `public void LevelUp()`: Añade 1 al marcador de nivel.
- `public void ResetScore()`: Pone a 0 los marcadores de Link, Happy y Friend.

Se resumen, en la siguiente figura, los Game Objects y los scripts utilizados:

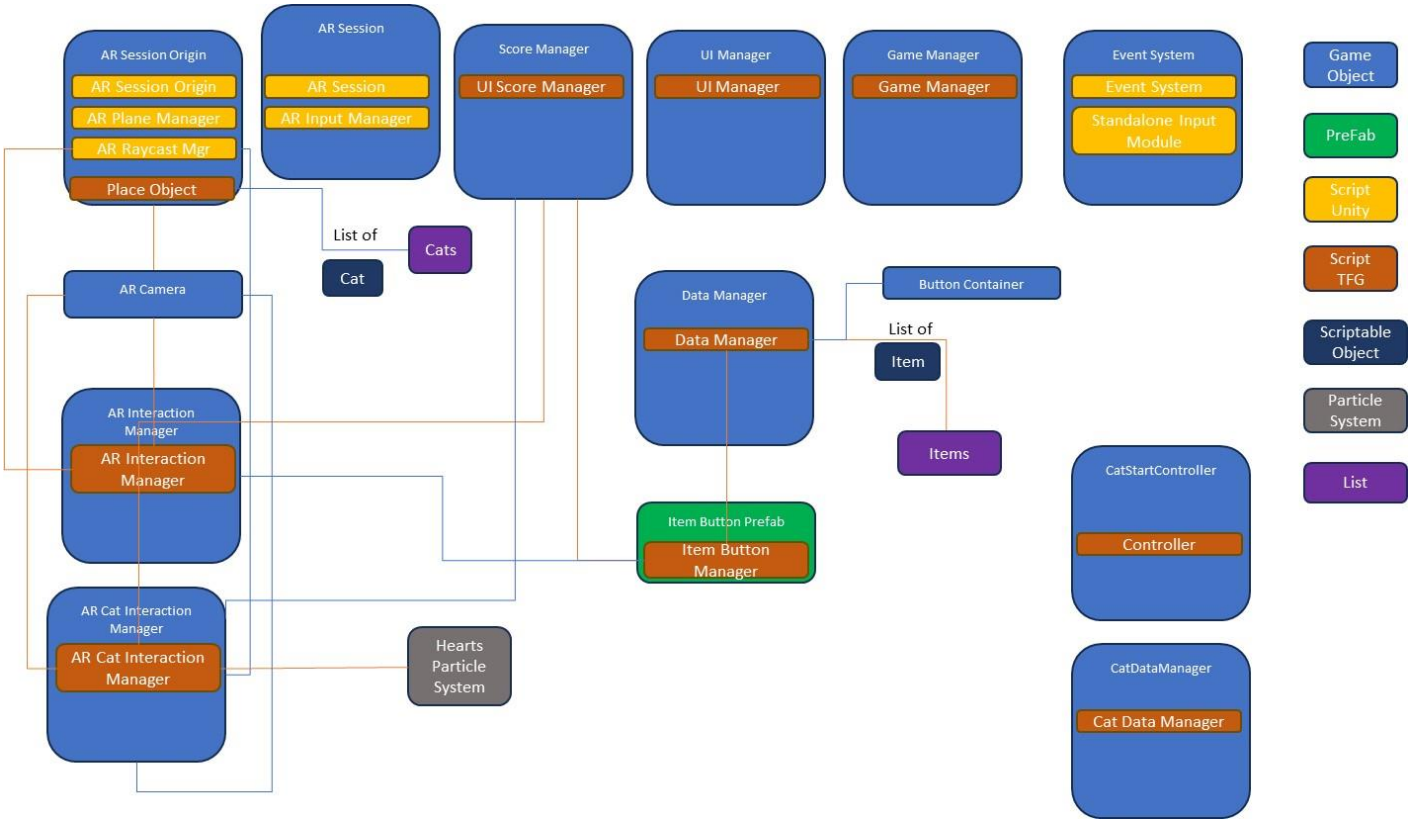


Figura 20: GameObjects y Scripts utilizados

4.3. Diseño gráfico e interfaces

4.3.1. Historias de usuario:

Se describen a continuación, las historias de usuario que servirán para completar los requerimientos de la aplicación:

- Como usuario quiero saber en todo momento cual es mi nivel, para monitorizar mi progreso.
- Como usuario quiero saber en todo momento cuántas monedas tengo, para saber si puedo comprar nuevos *items*.

- Como usuario quiero saber en todo momento cual es mi vínculo con mi mascota virtual para monitorizar mi progreso.
- Como usuario quiero saber en todo momento si mi mascota virtual es feliz y si nuestro nivel de amistad aumenta, para monitorizar mi progreso.
- Como usuario quiero acceder a mi galería de gatos adoptados, para monitorizar mi progreso.
- Como usuario quiero acceder a mi inventario de *items* para poder interaccionar con ellos.
- Como usuario quiero saber qué *ítems* tengo disponibles y qué ítems no tengo, para poder organizar mis compras.
- Como usuario quiero saber cuánto cuesta cada *item* para poder organizar mis compras.
- Como usuario quiero poder comprar ítems para interaccionar con mi mascota virtual.
- Como usuario quiero poder posicionar los *ítems* sobre la escena para interactuar con la mascota virtual.
- Como usuario quiero poder mover *ítems* que se encuentran sobre la escena para interactuar con la mascota virtual.
- Como usuario quiero poder eliminar *ítems* de la escena para limpiarla de lo que ya no se utilice.

4.3.2. Casos de uso:

Los actores que intervienen en este videojuego son:

- Usuario o jugador: es la persona que descarga la aplicación, la instala en su equipo y la utiliza.
- Gato Virtual: protagonista del videojuego que responde a la interacción con el usuario según su programación.
- Aplicación: sistema informático que responde a la interacción con el usuario según su programación.

Los casos de uso se dividen en los diferentes estados del juego: Pantalla inicio, Modo Normal, Cazar al gato, Modo Items, Modo Galería, Modo Posicionar. Se detallan a continuación los diagramas de casos de uso para cada estado de juego:

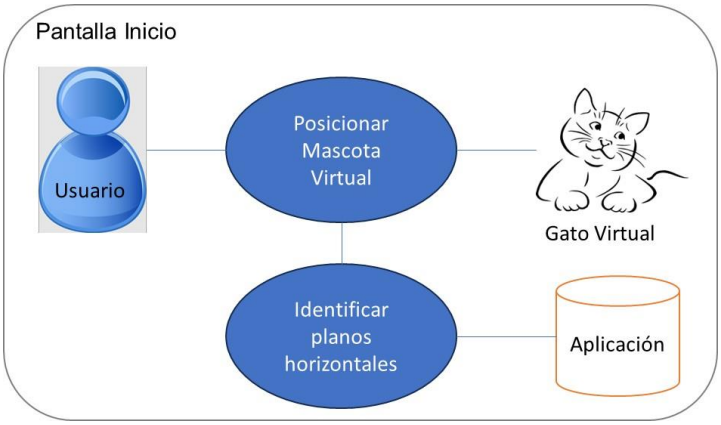


Figura 21: Diagrama de Casos de Uso Pantalla Inicio

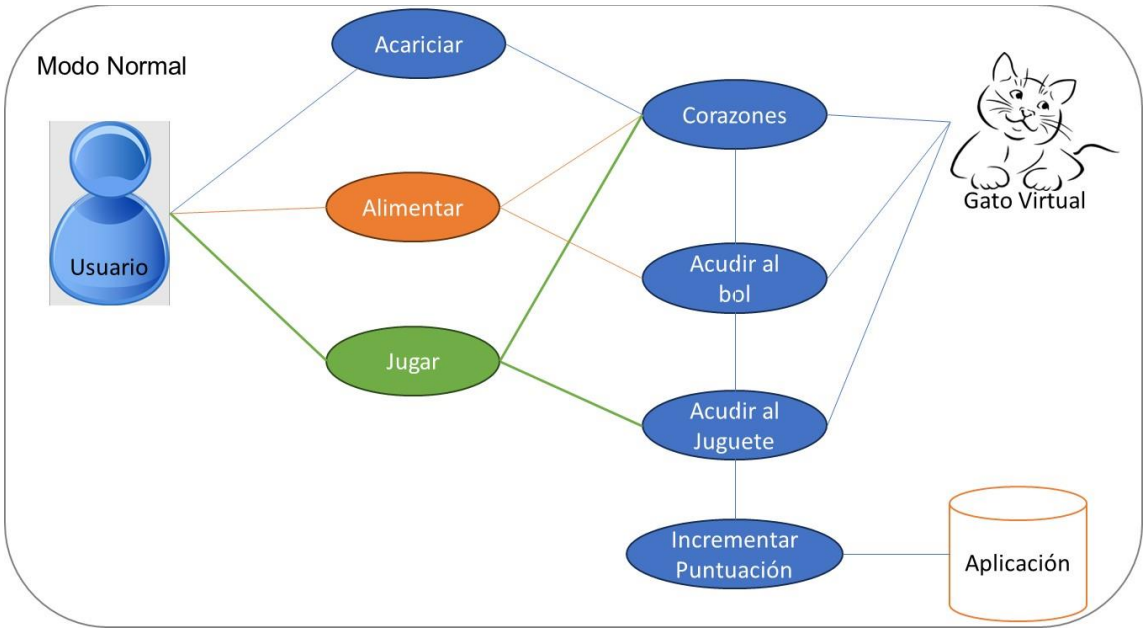


Figura 22: Diagrama de Casos de Uso Modo Normal Parte 1

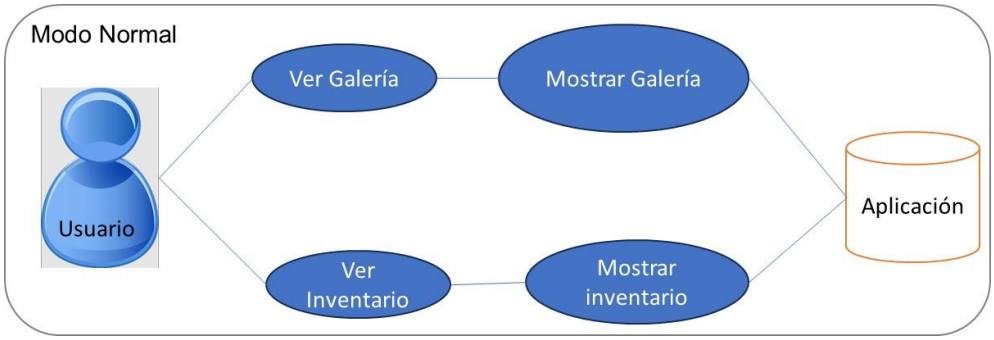


Figura 23: Diagrama de Casos de Uso Modo Normal Parte 2

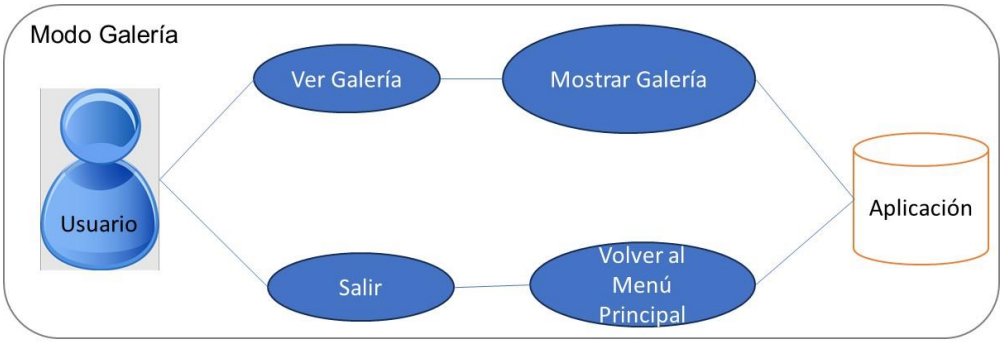


Figura 24: Diagrama de Casos de Uso Modo Galería

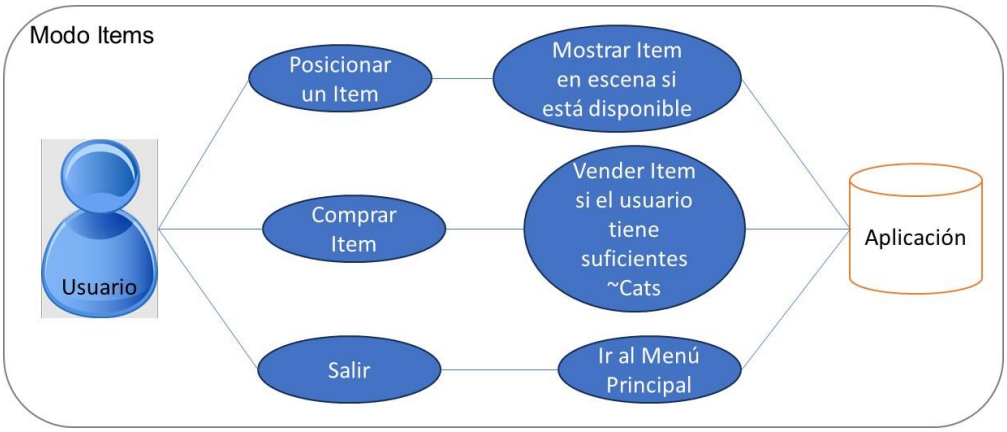


Figura 25: Diagrama de Casos de Uso Modo Items

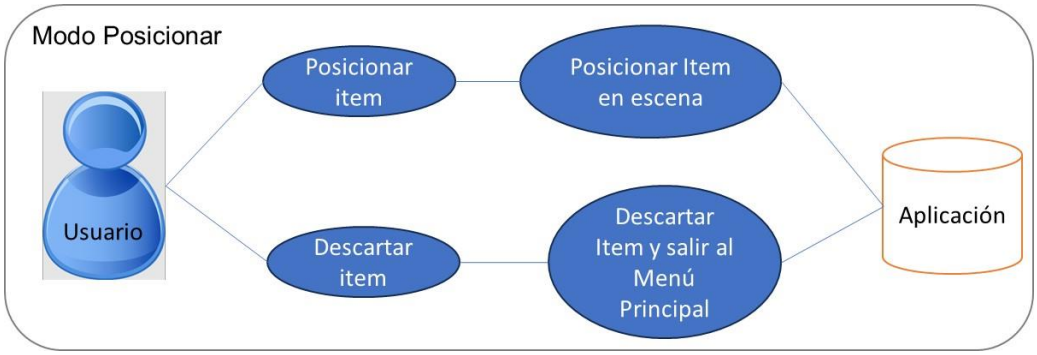


Figura 26: Diagrama de Casos de Uso Modo Posicionar

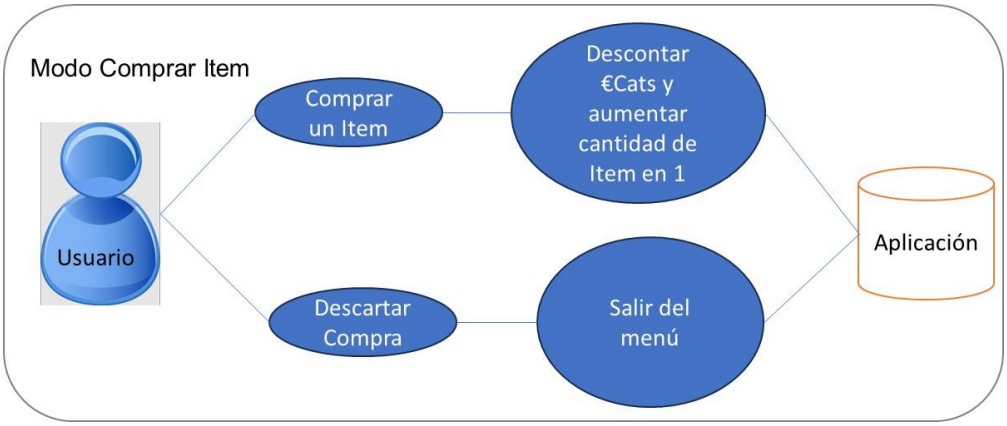


Figura 27: Diagrama de Casos de Uso Modo Comprar Item

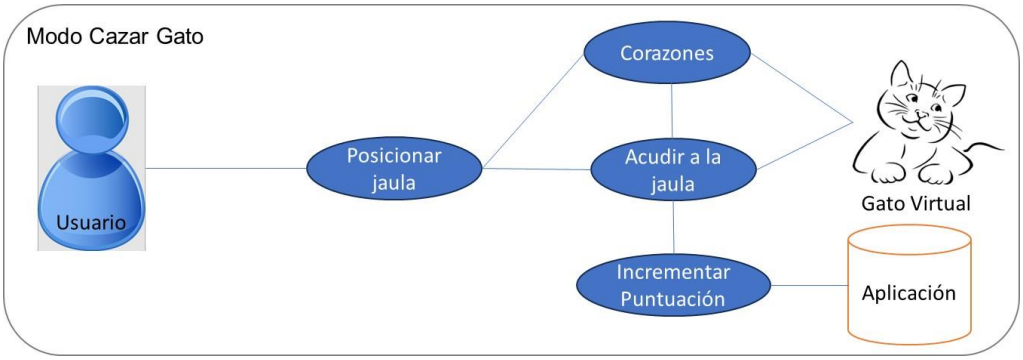


Figura 28: Diagrama de Casos de Uso Modo Comprar Cazar Gato

Caso de Uso	**Posicionar Gato Virtual**		
UC ID	001	**Estado**	Pantalla Inicio
Propósito	Instanciar gato virtual en la escena de juego		
Actores	Usuario, Aplicación, Gato Virtual		
Precondición	El usuario debe haber abierto la aplicación y haber permitido que la aplicación acceda a la cámara del equipo.		
Eventos Flujo Normal	Aparecen planos horizontales donde puede posicionarse la mascota virtual El usuario pulsa con el dedo sobre uno de los planos. Aparece el gato virtual en el punto elegido por el usuario.		
Postcondición	Desaparecen los planos horizontales de referencia.		

Caso de Uso	**Acariciar Gato Virtual**		
UC ID	002	**Estado**	Modo Normal
Propósito	Acariciar al gato virtual para ganar puntos de vínculo		
Actores	Usuario, Aplicación, Gato Virtual		
Precondición	El usuario debe haber instanciado y posicionado a un gato virtual en escena.		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre el gato y lo mueve como si lo acariciara sobre la pantalla.		

	Salen volando corazones desde el gato virtual.
Postcondición	Incrementan los marcadores de Happy, Friend y Link.

Caso de Uso	**Alimentar Gato Virtual**		
UC ID	003	**Estado**	Modo Normal
Propósito	Alimentar al gato virtual para ganar puntos de vínculo		
Actores	Usuario, Aplicación, Gato Virtual		
Precondición	El usuario debe haber instanciado y posicionado a un gato virtual en escena. El usuario debe haber comprado un BOL en el menú de Items		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre el BOL y lo posiciona en escena. El gato acude a comer. Salen volando corazones desde el gato virtual.		
Postcondición	Incrementan los marcadores de Happy, Friend y Link.		

Caso de Uso	**Jugar con el Gato Virtual**		
UC ID	004	**Estado**	Modo Normal
Propósito	Jugar con el gato virtual para ganar puntos de vínculo		
Actores	Usuario, Aplicación, Gato Virtual		
Precondición	El usuario debe haber instanciado y posicionado a un gato virtual en escena. El usuario debe haber comprado un JUGUETE en el menú de Items		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre el JUGUETE y lo posiciona en escena o lo mueve en escena. El gato acude a jugar. Salen volando corazones desde el gato virtual.		
Postcondición	Incrementan los marcadores de Happy, Friend y Link.		

Caso de Uso	**Acceder a la Galería de adopciones**		
UC ID	005	**Estado**	Modo Normal
Propósito	El usuario quiere inspeccionar la galería de adopciones para comprobar cuántos y qué gatos han sido adoptados.		
Actores	Usuario, Aplicación		
Precondición	Haber abierto la aplicación		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de GALERIA-caja cerrada		
Postcondición	Se abre el menú de galería de adopciones donde pueden verse los gatos adoptados		

Caso de Uso	**Acceder al Inventario**		
UC ID	006	**Estado**	Modo Normal

Propósito	El usuario quiere inspeccionar el inventario de Items.
Actores	Usuario, Aplicación
Precondición	Haber abierto la aplicación
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de Items
Postcondición	Se abre el menú de Items

Caso de Uso	**Ver gatos en Galería de adopciones**		
UC ID	007	**Estado**	Modo Galería
Propósito	Inspeccionar la galería de adopciones para comprobar cuántos y qué gatos han sido adoptados.		
Actores	Usuario, Aplicación		
Precondición	El Usuario ha pulsado sobre el icono Galería- caja cerrada		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre el scroll con los gatos para deslizarlo hacia la izquierda o hacia la derecha		
Postcondición	El scroll se detiene donde el usuario ha elegido		

Caso de Uso	**Cerrar Galería de adopciones**		
UC ID	008	**Estado**	Modo Galería
Propósito	Cerrar la galería de adopciones.		
Actores	Usuario, Aplicación		
Precondición	El Usuario ha pulsado sobre el icono Galería		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de GALERIA – caja abierta		
Postcondición	Se abre el menú principal		

Caso de Uso	**Ver *Items***		
UC ID	009	**Estado**	Modo Items
Propósito	Inspeccionar el inventario para comprobar cuántos y qué Items han sido adquiridos.		
Actores	Usuario, Aplicación		
Precondición	El Usuario ha pulsado sobre el icono Items- caja cerrada		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre el scroll con los Items para deslizarlo hacia la izquierda o hacia la derecha		
Postcondición	El scroll se detiene donde el usuario ha elegido		

Caso de Uso	**Cerrar *Items***		
UC ID	010	**Estado**	Modo Items
Propósito	Cerrar Inventario de Items.		

Actores	Usuario, Aplicación
Precondición	El Usuario ha pulsado sobre el icono Items-caja cerrada
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de Items – caja abierta
Postcondición	Se abre el menú principal

Caso de Uso	**Instanciar en escena un *Item***		
UC ID	011	**Estado**	Modo Items
Propósito	Instanciar en escena un item para interaccionar con él o con la mascota virtual.		
Actores	Usuario, Aplicación		
Precondición	El Usuario ha pulsado sobre el icono Items-caja cerrada El Item está disponible (muestra cantidad 1)		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono del Item que desea posicionar		
Postcondición	Se abre el menú de Posicionar Item		

Caso de Uso	**Comprar un *Item***		
UC ID	012	**Estado**	Modo Items
Propósito	Comprar un Item para, posteriormente, Instanciarlo en escena e interaccionar con él o con la mascota virtual.		
Actores	Usuario, Aplicación		
Precondición	El Usuario ha pulsado sobre el icono Items-caja cerrada El Item no está disponible (muestra cantidad 0)		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono del Item que desea comprar		
Postcondición	Se abre el menú de Comprar Item		

Caso de Uso	**Posicionar un *Item***		
UC ID	013	**Estado**	Modo Posicionar Item
Propósito	Posicionar un Item en escena para interaccionar con él o con la mascota virtual.		
Actores	Usuario, Aplicación		
Precondición	El usuario ha pulsado con el dedo sobre icono del Item que desea posicionar.		
Eventos Flujo Normal	El usuario mueve el item por escena y lo rota a conveniencia. Cuando el item se encuentra en la posición deseada, pulsa con el dedo sobre icono de OK.		
Postcondición	Se abre el menú principal		

Caso de Uso	**Descartar un *Item***		
UC ID	014	**Estado**	Modo Posicionar Item

Propósito	Descartar un Item que aún no ha sido posicionado.
Actores	Usuario, Aplicación
Precondición	El usuario ha pulsado con el dedo sobre el Item que desea descartar.
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de papelera.
Postcondición	Desaparece el Item de escena y se abre el menú principal

Caso de Uso	**Mover un *Item***		
UC ID	015	**Estado**	Modo Normal
Propósito	Mover un Item posicionado.		
Actores	Usuario, Aplicación		
Precondición	El usuario ha pulsado con el dedo sobre el Item que desea mover.		
Eventos Flujo Normal	El usuario mueve el item o lo rota en la escena. El usuario pulsa con el dedo sobre icono de OK.		
Postcondición	El Item queda en la nueva posición y se abre el menú principal.		

Caso de Uso	**Descartar un *Item* ya posicionado**		
UC ID	016	**Estado**	Modo Normal
Propósito	Descartar un Item ya posicionado.		
Actores	Usuario, Aplicación		
Precondición	El usuario ha pulsado con el dedo sobre el Item que desea descartar.		
Eventos Flujo Normal	El usuario pulsa con el dedo sobre icono de papelera.		
Postcondición	Desaparece el Item de escena y se abre el menú principal		

Caso de Uso	**Cazar a un gato virtual**		
UC ID	017	**Estado**	Modo Item
Propósito	Adoptar al gato virtual.		
Actores	Usuario, Aplicación, gato Virtual		
Precondición	El usuario ha conseguido elevar el contador de BOND al máximo nivel. El usuario ha entrado en el inventario pulsando en el icono Item-caja cerrada		
Eventos Flujo Normal	El usuario posiciona la jaula en escena El gato se encamina hacia la jaula Si el gato decide no entrar, el usuario debe utilizar comida para convencerlo.		
Postcondición	Aparecen corazones en escena. Desaparece el gato de escena. Aparece el gato en la galería Se actualizan los contadores de monedas y nivel		

	Aparecen nuevos planos de posicionamiento en escena para que se posicione un nuevo gato virtual

4.3.3. Usabilidad/UX:

La navegación entre menús se produce al interaccionar con los botones del ***UIManager***, como se resume en la siguiente figura y se describe a continuación.

Al abrir la aplicación por primera vez, aparece el *Welcome Menu*, explicando el funcionamiento de los botones, del panel de contadores y una breve sinópsis del objetivo del juego. Al clickar sobre el botón se pasa al *Main Menu* **(1).** Si se hace click sobre el botón de *Gallery* (icono azul con caja cerrada), se pasa **(2)** al *Gallery Menu*. Si se vuelve a clickar sobre el botón de *Gallery* (icono azul con caja abierta), se vuelve al *Main Menú* **(3)**. Si en el *Main Menu* se hace click sobre el botón de *Item* (icono morado con caja cerrada), se pasa al *Items Menú* **(4)**. Una vez en el *Items Menu*, si se clica de nuevo sobre el botón Items ((icono morado con caja abierta), se regresa al *Main Menu* **(10)**.

En el desplegable aparecen todos los Items, los que tienen cantidad 1 están disponibles, los que tienen cantidad 0 no lo están hasta que sean adquiridos. Estando en el menú Items, si se hace click sobre un Item adquirido **(5)**, se pasa al *AR interaction Menú*, que permite posicionar el item o descartarlo. Tanto si se posiciona como si se descarta, se pasa al *Main Menu* **(6)**.

Si en el desplegable se toca un *Item* con cantidad cero **(7)**, se pasa al menú de compra (*Shop Menu*), en este menú, si se toca el botón de “aprobado” (icono azul), se descuenta del total de Ecats el importe del artículo, se actualiza la cantidad a 1 y se devuelve al *Items Menu* **(8)**. Si se toca el botón con la cruz (descartar), se regresa al *Items Menu* **(9)** sin realizar ningún cambio. Una vez el artículo se ha adquirido, si se clica de nuevo sobre él, la aplicación va al *ARInteraction Menu* **(5)**.

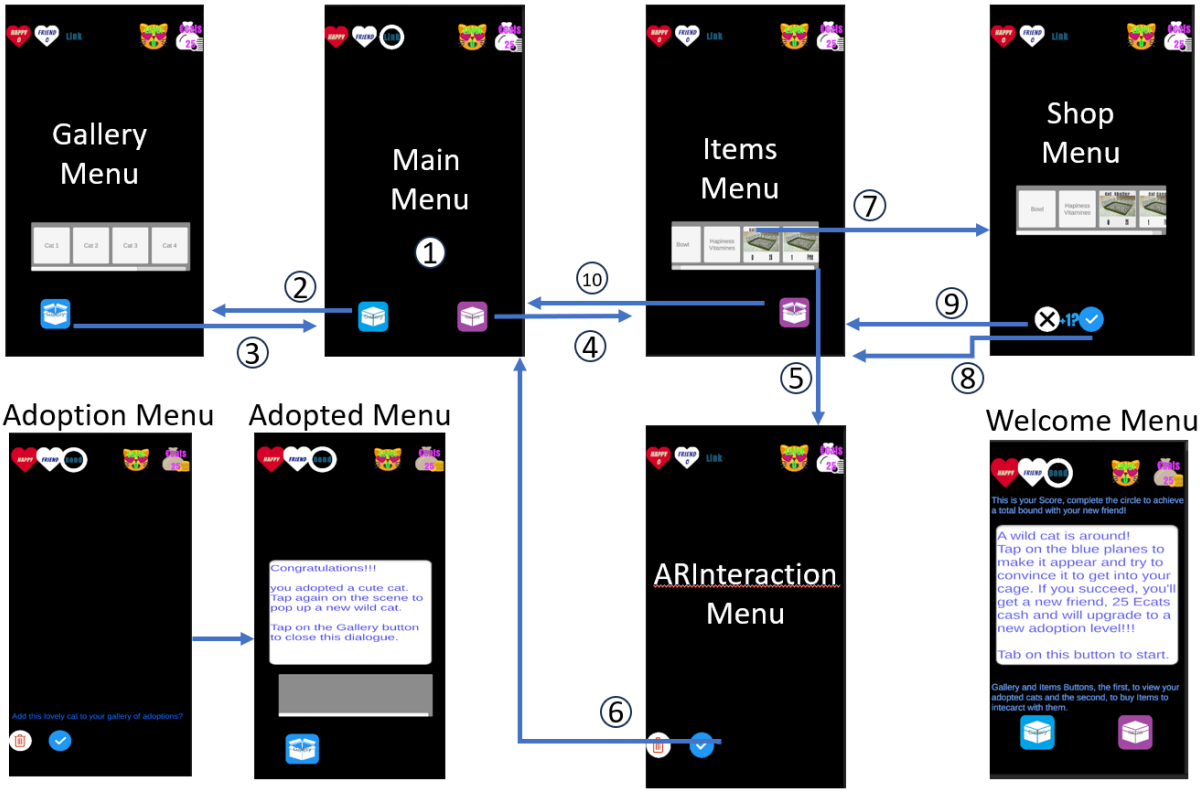


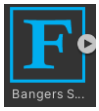
Figura 29: Navegación entre menús de Usuario

El menú de *Adoption Menu* aparece cuando un gato se ha introducido en la jaula que se ha puesto en escena. Si se hace click en descartar, se vuelve al menú principal sin realizar cambios. Si se hace click en aceptar, el gato se añade a la galería y aparece el *Adopted Menu* (en la figura, abajo a la izquierda).

4.3.4. Estilos

Se define a continuación la línea gráfica del trabajo:

- **Fuente utilizada:** Bangers SDF (TMP_Font Asset)



- **Iconos Marcadores del Juego:**

Marcador de *Happiness*

Marcador de *Friendship*

Marcador de *nivel de Vínculo*

Marcador de Nivel de *Casa de Acogida*

Marcador de número de monedas disponibles

- **Iconos de Interacción con el Usuario:**

Icono de acceso a Galería de Gatos

Icono de acceso a Inventario

Icono para cerrar la Galería de Gatos

Icono para cerrar el Inventario de *Items*

Icono para comprar una unidad de *Item* en el Menú de comprar *Item*

Icono para descartar compra del *Item* en el Menú de comprar *Item*

Icono para aceptar el posicionamiento de un *Item* en escena en el Menú de Posicionar

Icono para eliminar un *Item* de la escena en el Menú de Posicionar

4.4. IA de los gatos virtuales

4.4.1. FSM

Para el primer gato (nivel 0) se ha implementado una Máquina de Estados Finitos (Finite States Machine) que gobierna el comportamiento del gato frente a los ítems que se utilicen

para interactuar con él (juguetes, alimento, y jaula). Se describen, en la siguiente figura, los estados y las transiciones:

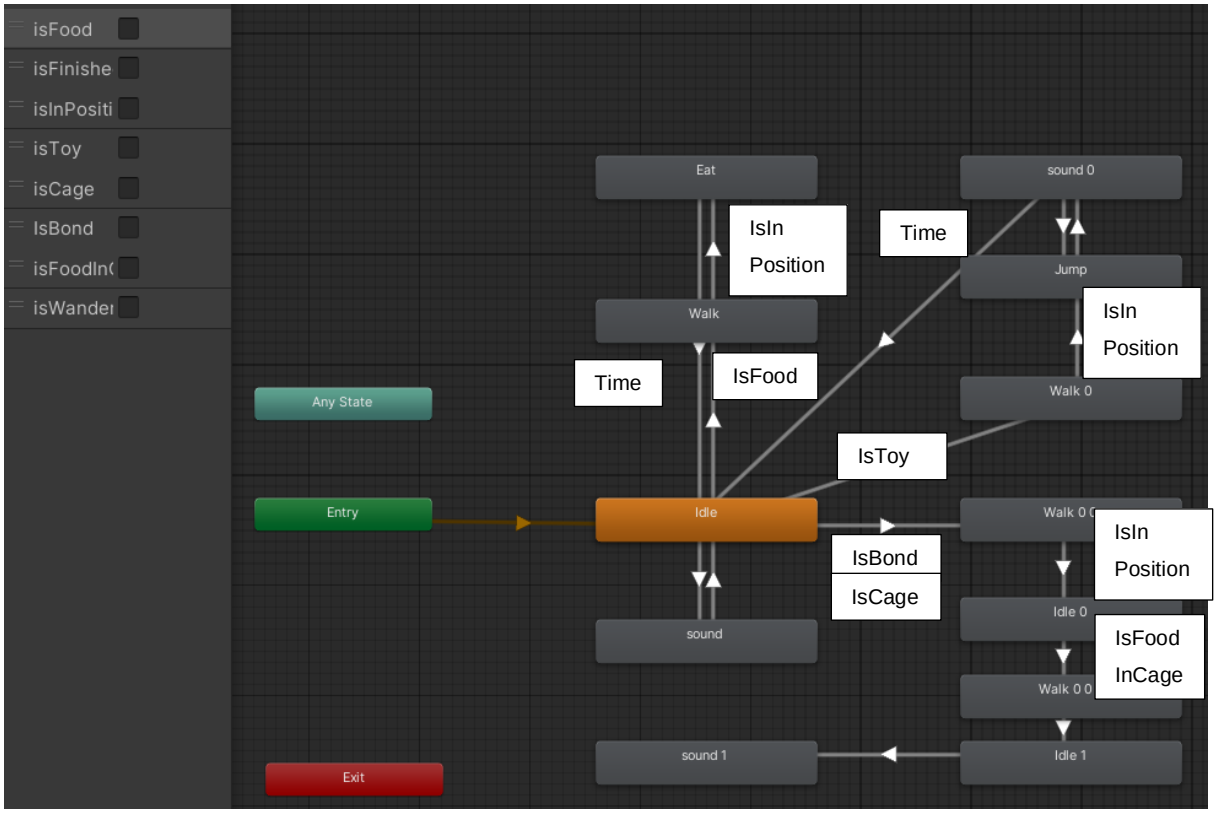


Figura 30: FSM gato nivel 0

4.4.2. Árbol de comportamiento

Para el resto de los gatos (niveles 1-4) se ha implementado un árbol de comportamiento que dota de una interacción diferente a los gatos según sea su tipo (normal, tímido o asustadizo). Las mascotas tímidas sólo aceptarán alimento “Super” y las mascotas asustadizas sólo jugarán con juguetes “Super”. El gato del nivel 4 posee ambas características. Se resume el árbol en la siguiente figura:

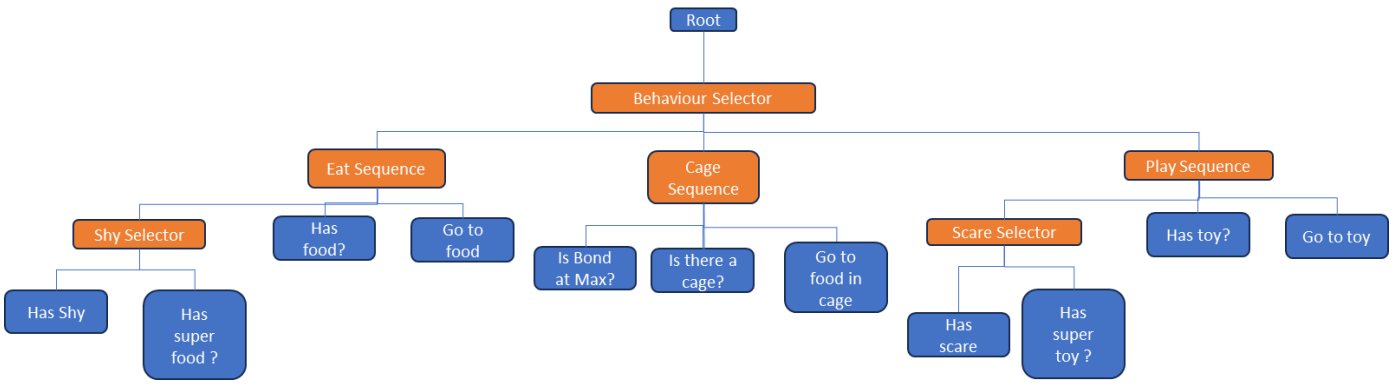


Figura 31: Árbol de comportamiento gatos nivel 1-4

Para implementar el árbol se han definido los siguientes *scripts* en *C#*:

Node: clase que implementa los Nodos del árbol. Contiene la función:

- `public void AddChild(Node n)`: añade el nodo n a la lista de hijos.

BehaviourTree: clase que hereda de Node e implementa el primer nodo (raíz) del árbol de decisión. Contiene la función:

- `public void PrintTree()`: imprime por consola el árbol de comportamiento.

Leaf: clase que hereda de Node e implementa las hojas del árbol.

Selector: clase que hereda de Node e implementa nodos de tipo selector

Sequence: clase que hereda de Node e implementa nodos de tipo secuencia.

CatBehaviour: clase Monobehaviour que implementa el caso particular del árbol de decisión de los gatos 2 a 5 y las acciones que se desarrollan en las hojas del árbol.

4.5. Lenguajes de programación y APIs utilizados

Se enumeran las principales plataformas y herramientas utilizadas para desarrollar este videojuego de realidad aumentada.

4.5.1. Lenguajes de programación

C#, llamado “C Sharp” *[16]*, es el principal lenguaje de programación utilizado en UNITY. C# es un lenguaje que evoluciona de “C” y “C++”, añadiendo funcionalidades de otros lenguajes de programación como “Java o “Visual Basic”. Se trata de un lenguaje de programación orientado a objetos que Microsoft desarrolló para su plataforma NET. Destaca por su sencillez de aprendizaje y su facilidad de uso, siendo Microsoft Visual Studio su IDE por excelencia.

4.5.2. Bibliotecas

Android SDK *[17]*, se trata de un conjunto de bibliotecas para desarrolladores de aplicaciones para el sistema operativo Android. Se compone de varios paquetes necesarios para el desarrollo de aplicaciones, incluyendo herramientas de línea de comandos, de compilación, de plataforma (como adb y fastboot) y emuladores.

JDK “Java Development Kit”, es un conjunto de bibliotecas para el desarrollo de aplicaciones Java. Se utiliza el OpenJDK 1.8.0_152.

4.5.3. Herramientas de desarrollo

Unity Hub *V3.6.1*, *[18]* aplicación de Unity independiente que agiliza la búsqueda, descarga y administración de los proyectos e instalaciones de Unity. Además, permite agregar manualmente versiones del Editor que ya se encuentren instaladas en la máquina.

Unity 2021.3.1f1, *[18] e*ditor de Unity para crear juegos, aplicaciones y experiencias 2D y 3D. Se puede descargar en www.unity3d.com. Su interfaz gráfica interactiva y relativamente fácil de usar, facilita la creación de menús y escenas.

Visual Studio Code V17.7.6, *[19]* se trata de un editor de código fuente ligero pero potente que se ejecuta en el escritorio y se encuentra disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un amplio ecosistema de extensiones para otros lenguajes como C++, C#, Java, Python, PHP, Go, .NET.

AR Core XR Plugin V4.2.3, *[20]* es el SDK de realidad aumentada de Google que ofrece API multiplataforma para crear nuevas experiencias inmersivas en Android, iOS, Unity y Web.

GitHub, *[21]* se trata de una plataforma y un servicio basado en la nube para el desarrollo de software y control de versiones, que permite a los desarrolladores almacenar y administrar su código. Utiliza el software *Git*, que proporciona control de versiones distribuidas de Git, además de control de acceso, seguimiento de errores, solicitudes de funciones de software, gestión de tareas, integración continua y wikis para cada proyecto.

Tom's Planner, *[22]* es un servicio de aplicaciones y herramientas web, para la planificación, gestión y colaboración de proyectos.

4.6. Requisitos e instrucciones de instalación

Para ver el proyecto en UNITY, versión 2021.3.1f1 (coincidir con la versión es muy importante, el juego no funcionará en versiones anteriores de UNITY y puede dar problemas de coherencia con versiones posteriores), se deben seguir los siguientes pasos:

1) Copiar el repositorio de git hub en una partición local de su disco duro
2) Abrir la aplicación Unity Hub
3) Seleccionar "Abrir"
4) Seleccionar "Agregar proyecto desde el disco".
5) Busque la carpeta donde descargó los archivos del repositorio de Git Hub.

6) haga clic en el botón "Agregar proyecto".

Una vez que tenga su proyecto abierto en UNITY, deberá compilarlo en su dispositivo Android.

1) haga clic en Archivo --> Configuración de compilación

2) seleccione ANDROID en la pestaña de la plataforma

3) conecta tu dispositivo Android a tu PC

4) haga clic en el botón Actualizar en el campo "Ejecutar dispositivo".

5) haga clic en la ventana emergente "dispositivo predeterminado", su dispositivo Android debería aparecer allí, selecciónelo. Si su dispositivo no aparece, consulte las instrucciones del fabricante para habilitar la depuración USB en su dispositivo Android. Esto permitirá a Unity implementar aplicaciones directamente en su dispositivo móvil a través de una conexión de cable USB.

Siga esta documentación de Android para habilitar la depuración USB en su dispositivo: https://developer.android.com/studio/debug/dev-options

6) haga clic en "Construir y ejecutar".

7) guarde la compilación (Build) en una carpeta "compilaciones" (Builds) en el repositorio de carpetas del juego UNITY (si aún no existe, deberá crearla).

Una vez que la aplicación esté instalada en su dispositivo Android, solo necesita desconectar la conexión USB y desbloquear la pantalla de su dispositivo (si está bloqueada). La aplicación debería abrirse automáticamente. ¡A disfrutar del juego!

5. Demostración

5.1. Instrucciones de uso

Con el fin de instanciar adecuadamente a los diferentes gatos virtuales, se recomienda utilizar la aplicación en un entorno bien iluminado y donde existan espacios despejados con superficies planas horizontales, que puedan ser detectadas con la cámara del equipo utilizado.

5.2. Prototipos

Los prototipos utilizados durante el proceso de desarrollo han ido añadiendo funcionalidad creciente, empezando por el entorno de usuario y la implementación de la funcionalidad del primer gato. Posteriormente, se ha introducido la funcionalidad de compra y uso de ítems y la gestión del resto de gatos de cada nivel. Se muestra en la siguiente tabla la relación entre los commits de GitHub y la funcionalidad construida:

Commits	**Funcionalidad**
1-2	Infraestructura juego AR
3-6	Detección de planos e instanciación primer modelo virtual
7-13	UI de contadores y botones de juego
14-16	Añadidos *Scriptable objects*
17-20	Añadida funcionalidad para mover Items en escena
21-22	Añadido sistema de partículas con forma de corazones. Eliminados botones UI
23-24	Añadido manejo de marcadores
25-27	Funcionalidad de compra de *Items*
28-32	Máquina de estados finitos para el primer gato
33-37	Funcionalidad para adoptar al gato y pantalla de bienvenida
40	Añadidos sonidos a los gatos
41-42	*Bug fixing*
43	Añadido menú de adopción
44-46	Árbol de comportamiento para gatos 2-4. Añadida pantalla de fin de juego
47-61	*Bug Fixing*

Tabla 4: Relación commits - funcionalidad

5.3. Tests y resultados

Para depurar la funcionalidad, se han ido construyendo prototipos de los diferentes *sprints* y se han ido probando para verificar que la funcionalidad tenía el comportamiento esperado.

Para el *debug* de la aplicación de realidad aumentada se ha utilizado Lunar Mobile Console, un "*plug in*" para IOS y Android gratuito, que se puede descargar del *Unity Asset Store*. Para que funcione, es importante que al realizar el Build de la aplicación desde Unity, se active el módulo de *LunarConsole* y se marque el *build* como "*Development Build*".

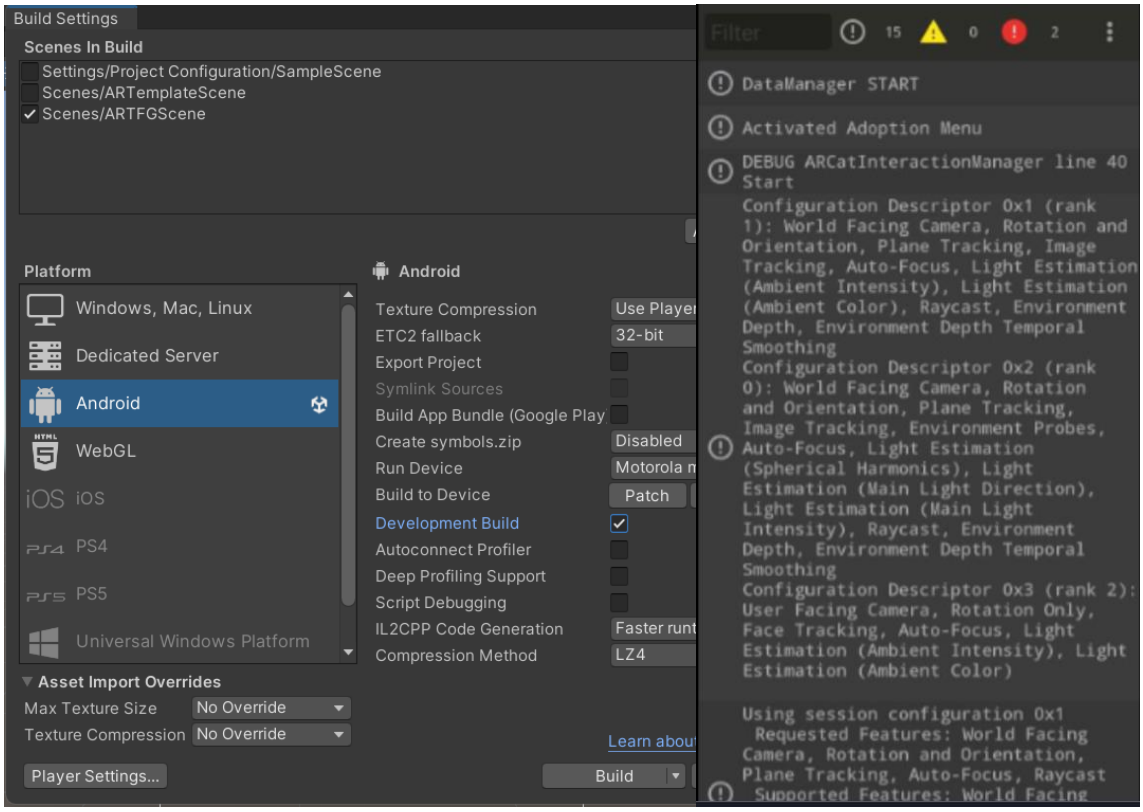


Figura 32: a la Izquierda, cómo configurar el *Build* en UNITY. A la derecha, pantalla de *Lunar M Console* en el dispositivo.

5.4. Ejemplos de uso del producto (o guía de usuario)

Al iniciar la aplicación, como se muestra en la siguiente figura, aparece la pantalla de bienvenida explicando en qué consiste el juego, así como el funcionamiento de los marcadores y de los botones de interacción con el usuario:

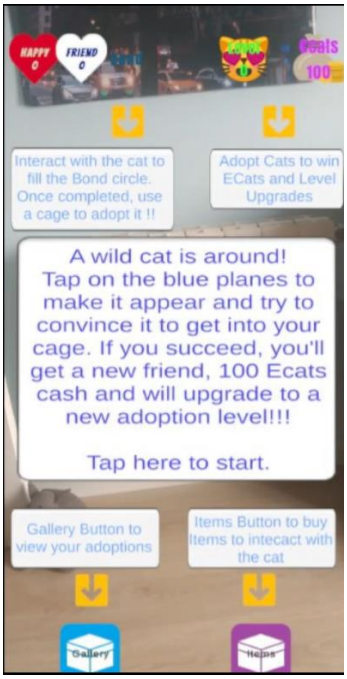


Figura 33: Pantalla de bienvenida

A continuación, la aplicación empieza a mostrar los planos horizontales que encuentra en el campo de visión de la cámara, al presionar sobre alguno de ellos, se instanciará el primer gato virtual, junto con una segunda pantalla de bienvenida explicando al usuario la forma de proceder.

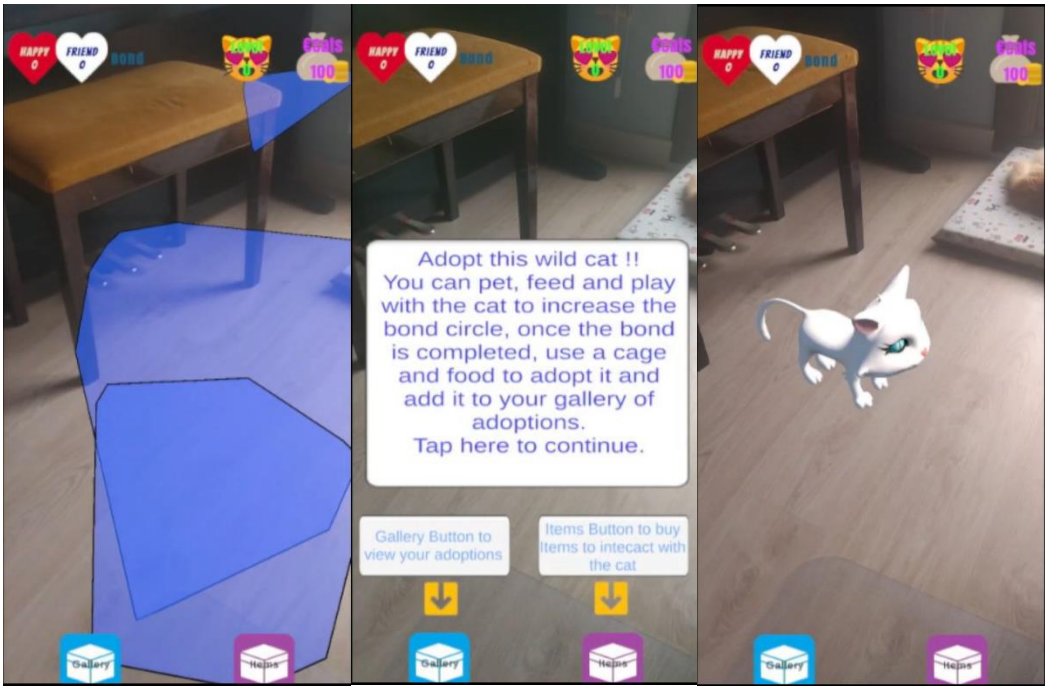


Figura 34: De izquierda a derecha: detección de planos, segunda pantalla de bienvenida y primer gato virtual

A través de acariciar al gato sobre la pantalla, darle de comer o jugar con él, el usuario incrementará los marcadores de *Happy, Friend* y *Bond*. Cuando este último esté completo, el

gato estará preparado para ser adoptado, instante en el que podremos colocar una jaula en la escena y convencerlo para que entre en ella. Una vez conseguido, se abre el menú de adopción.

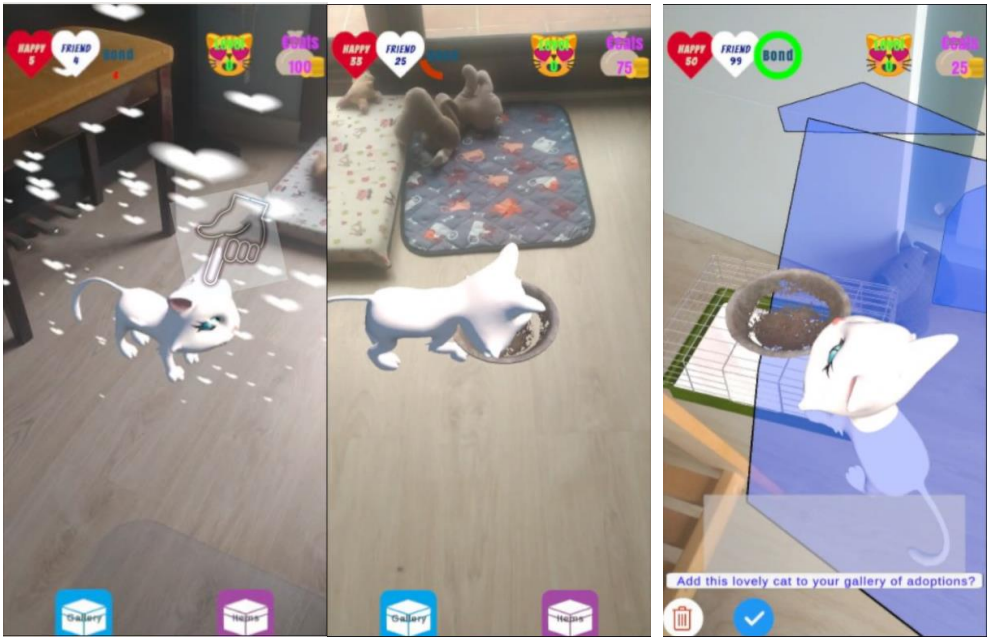


Figura 35: De izquierda a derecha: acariciar y dar de comer para completar el círculo de Bond y adoptar al gato.

Si el usuario acepta la adopción, aparece un menú de felicitación, el gato se incorpora a su galería de adopciones, sube un nivel y gana 100 € Cats. Al hacer click en cualquier lugar de la escena, se instanciará un nuevo gato virtual.

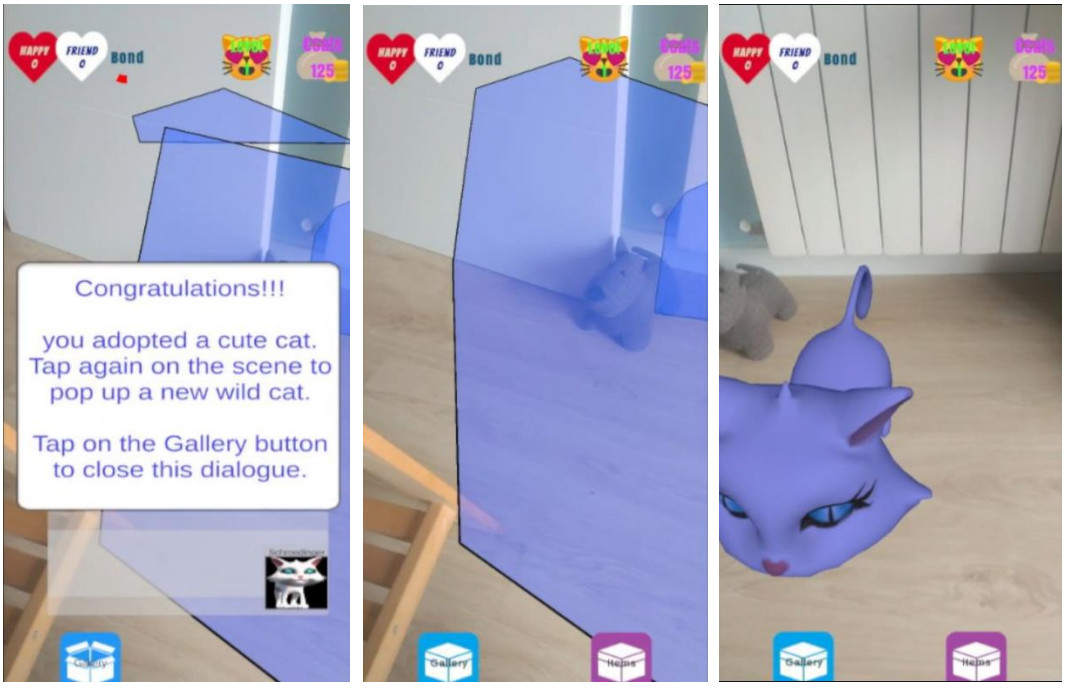


Figura 36: De izquierda a derecha: menú de felicitación, *reset* de marcadores con subida de nivel y nuevo gato virtual.

Los gatos que se encuentran en niveles más avanzados son más difíciles de vincular (incrementar círculo de Bond). Si se trata de un gato tímido, no aceptará comida normal, sólo Super comida. Si se trata de un gato miedoso, no querrá jugar con juguetes normales, sólo con Super Juguetes. El último gato de esta versión Demo, en el nivel 4, es tímido y miedoso a la vez.

6. Conclusiones y líneas de futuro

6.1. Conclusiones

Se recogen en este capítulo las conclusiones personales del autor sobre el proyecto realizado, el proceso de trabajo y los resultados obtenidos.

En general, el nivel de satisfacción con el resultado del trabajo de fin de grado es muy alto. Se ha partido desde cero, con el concepto esencial del videojuego y se ha desarrollado, sobre UNITY, una aplicación para ANDROID que consigue la mayor parte de los objetivos que se habían propuesto al inicio del proyecto. Las lecciones aprendidas durante su elaboración han sido las siguientes:

- El desarrollo del UI es complejo y muy laborioso.
- La generación de los Use Cases, son clave para la elaboración de una buena una lista de requerimientos y para poder planificar debidamente la funcionalidad que se entregará en el tiempo (*sprints*).
- Las herramientas para hacer *debug* de un videojuego de realidad aumentada son pocas y muy crípticas.
- Crear una versión “vendible” de un videojuego requiere mucho talento en diferentes disciplinas y mucho tiempo de diseño, implementación y test. Atrás quedaron los tiempos del famoso Tetris en blanco y negro con 300 *pixeles* por pulgada.

Respecto a los objetivos planteados inicialmente, se han conseguido todos los principales. En cambio, de los secundarios, no se han conseguido los siguientes:

- Concienciar sobre las dificultades que experimentan los gatos para sobrevivir en las calles de nuestras ciudades y la labor de las protectoras de animales. Dado que, para preservar el flujo del juego y la facilidad de uso, se han obviado problemas importantes que tienen los gatos callejeros como enfermedades, envenenamientos o hambruna.
- Conocer formas de monetizar un videojuego de estas características, dado que no ha habido tiempo material para explorar posibles opciones.

Finalmente, analizando la planificación y metodología utilizada a lo largo del proyecto, la planificación inicial ha debido adaptarse bastante al desarrollo del proyecto para eliminar algunas tareas que finalmente han sido consideradas innecesarias y para añadir otras, que al inicio no se habían considerado, como, por ejemplo, el desarrollo del UI. En cuanto a la metodología utilizada de trabajar por *sprints*, entregando y probando funcionalidad en dosis controladas en el tiempo, ha sido altamente satisfactoria, ya que desde una primera

implementación, se ha podido probar la funcionalidad sobre el prototipo y *debugarla*, con lo que la siguiente funcionalidad que se iba construyendo, se levantaba sobre una base muy sólida.

6.2. Líneas de futuro

En cuanto a posibles líneas de desarrollo futuro, se propondría explorar las siguientes:

- Formas de monetización del videojuego.
- Utilizar documentos JSON para actualizar, tanto *Items* como Gatos virtuales desde un servidor y así poder actualizarlos sin tener que volver a instalar la aplicación.
- Habilitar intercambios de gatos y de *Items* entre dos usuarios. Esto implicaría, entre otras cosas, un registro de usuarios y habilitar las comunicaciones entre las aplicaciones de los diferentes jugadores.
- Guardar un registro de la puntuación entre sesiones de juego.
- Tener un ranking general de usuarios ordenados por el número de adopciones conseguidas.

Bibliografía

Las citas y referencias bibliográficas se han listado según el orden de aparición en el TFG.

[1] (2017) *Abogacía.es, Del Lado de los que Sienten. Rosario Monter*. Disponible en https://www.abogacia.es/publicaciones/blogs/blog-de-derecho-de-los-animales/el-control-de-colonias-felinas-etica-y-legalidad/ (Consultado el 3 de Octubre de 2023).

[2] (01/09/2023) *Gecon.es, Fundación Iberoamericana del Conocimiento*. *Flavio Escribano*. Disponible en https://gecon.es/tamagotchi-como-base-para-un-modelo-experto-de-gamificacion/ *(Consultado el 4 de Octubre de 2023).*

[3] Pokemon Go (2016) [Videojuego]. *Niantic*. Disponible en https://pokemongolive.com/ *(Consultado el 13 de Octubre de 2023).*

[4] Peridot (2023) [Videojuego]. *Niantic*. Disponible en https://playperidot.com/ *(Consultado el 13 de Octubre de 2023).*

[5] (01/09/2023) *Jobted.es, bolsa de trabajo y ofertas de empleo en España*. Disponible en https://www.jobted.es/salario/programador-videojuegos *(Consultado el 6 de Octubre de 2023)*

[6] Tamagotchi (1996) [Juego]. *Bandai*. Información disponible en https://www.bandai.es/licencia/tamagotchi *(Consultado el 13 de Octubre de 2023).*

[7] (30/08/2023) *es.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://es.wikipedia.org/wiki/Digimon *(Consultado el 13 de Octubre de 2023)*

[8] (07/07/2023) *en.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://en.wikipedia.org/wiki/Petz *(Consultado el 13 de Octubre de 2023)*

[9] (14/08/2023) *es.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://es.wikipedia.org/wiki/Neopets *(Consultado el 13 de Octubre de 2023)*

[10] (16/09/2023) *en.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://en.wikipedia.org/wiki/Nintendogs *(Consultado el 13 de Octubre de 2023)*

[11] (04/10/2023) *en.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://en.wikipedia.org/wiki/Webkinz *(Consultado el 13 de Octubre de 2023)*

[12] (05/09/2023) *en.wikipedia.org, Wikipedia, marca registrada de Fundación Wikimedia*. Disponible en https://en.wikipedia.org/wiki/My_Talking_Tom *(Consultado el 13 de Octubre de 2023)*

[13] (17/02/2022) *beam.eyeware.tech, Beam Artificial Vision Software Development*. Disponible en https://beam.eyeware.tech/es_es/top-10-realidad-virtual-software-herramientas-de-desarrollo-jugadores/ *(Consultado el 14 de Octubre de 2023)*

[14] (16/03/2023) *keverugames.com, Game art and development studio*. Disponible en https://kevurugames.com/blog/unity-vs-unreal-engine-pros-and-cons/#:~:text=C%23%20(Unity)%20is%20more%20suitable,latter%20clearly%20has%20an%20advantage. *(Consultado el 14 de Octubre de 2023)*

[15] (Version 2018.4) *docs.unity3d.com, Unity Documentation*. Disponible en https://docs.unity3d.com/es/2018.4/Manual/vuforia-sdk-overview.html#:~:text=Vuforia%20es%20una%20plataforma%20de,HMD)%20como%20Microsoft%20HoloLens)%20. (*Consultado el 14 de Octubre de 2023)*

[16] (18/11/2023) *https://en.wikipedia.org, Wikipedia, the free encyclopedia*. Disponible en https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#:~:text=C%23%20(pronounced%20See%20sharp)%20is,C%23. *(Consultado el 27 de Noviembre de 2023)*

[17] (Octubre 2023) https://developer.android.com, *Android Studio Developers*. Disponible en https://developer.android.com/tools/releases/platform-tools. *(Consultado el 27 de Noviembre de 2023)*

[18] (Versi'on 2020.1) https://docs.unity3d.com, *Unity*. Disponible en https://docs.unity3d.com/2020.1/Documentation/Manual/GettingStartedUnityHub.html#:~:text=The%20Unity%20Hub%20is%20a,your%20Unity%20Projects%20and%20installations. *(Consultado el 27 de Noviembre de 2023)*

[19] (Noviembre 2023) https://code.visualstudio.com, *Visual Studio Code*. Disponible en https://code.visualstudio.com/docs. *(Consultado el 27 de Noviembre de 2023)*

[20] (Noviembre 2023) https://developers.google.com/ar, *ARCore*. Disponible en https://developers.google.com/ar. *(Consultado el 27 de Noviembre de 2023)*

[21] (Noviembre 2023) https://github.com/, *GitHub*. Disponible en https://github.com/. *(Consultado el 27 de Noviembre de 2023)*

[22] (Noviembre 2023) https://www.tomsplanner.com/, *Tom's Planner*. Disponible en https://www.tomsplanner.com/. *(Consultado el 27 de Noviembre de 2023)*

Anexos

Se añaden, a continuación, apartados complementarios adicionales que son demasiado extensos para incluir dentro de la memoria y tienen un carácter auto-contenido.

Anexo A: Glosario

Glosario de temas y acrónimos utilizados en el trabajo con breves definiciones de cada uno de ellos.

Android: sistema operativo móvil, basado en el núcleo Linux y otros software de código abierto, diseñado para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tabletas, relojes inteligentes, televisores, etc.

AR: Augmented Reality, tecnología que permite observar elementos físicos de la realidad a través de dispositivos y software específico.

Árbol de Comportamiento: (BTs, por las siglas en inglés, Behavior Tree), tecnología utilizada para implementar comportamientos complejos en personajes virtuales.

Asset: o activo, es una representación de cualquier objeto que puede ser utilizado en un juego o proyecto digital.

Beta Tester: Un probador beta (del inglés beta tester), es un usuario de programas que se encuentran en fase de desarrollo y aún no se consideran finales por presentar diferentes modos de fallo o características no implementadas.

Bond: del inglés, lazo o vínculo, en sentido figurado, expresa la unión o atadura de una persona o cosa con otra.

CA: Casa de Acogida, centro en el que se aloja de forma temporal a animales que se encuentran en estado de abandono.

Debug: del inglés, depuración, se trata del proceso de encontrar y solucionar errores en el código fuente de cualquier *software*.

Demo: Versión de demostración de un programa informático o de una grabación musical utilizada con fines de promoción.

Gallery: del inglés, galería, espacio dedicado a la exposición de objetos.

Game Object: objetos fundamentales en Unity que representan personajes, accesorios, y el escenario.

Feedback: del inglés retroalimentación, una respuesta dada a algún estímulo como forma de evaluarlo.

Friendship: del inglés, amistad, afecto personal y desinteresado que nace y se fortalece con el trato.

FSM: Finite States Machine.

Ganntt: diagrama de, herramienta de gestión de proyectos que ilustra el trabajo realizado durante un período de tiempo en relación con el tiempo previsto inicialmente para desarrollarlo.

Hapiness: del inglés felicidad, definido como estado de grata satisfacción espiritual y física.

IA: Inteligencia Artificial, disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico.

IOS: en inglés, Iphone Operating System, sistema operativo propietario de la empresa Apple Inc. utilizado en dispositivos de la marca como smartphones, tabletas, televisores o reproductores mp4.

Items: del inglés, artículos.

Level: del inglés, nivel.

Máquina de estados: representación de un sistema reactivo basado en eventos que pasa de un estado a otro si se cumple la condición que controla el cambio.

MV: Mascota Virtual.

PEC: Prueba de Evaluación Continuada

Publishers: del inglés editores, en el contexto de este trabajo, se refiere a los eventos que generan acciones.

Plug in: del inglés, enchufar. En el ámbito de la programación hace referencia a pequeños programas complementarios que amplían las funciones de aplicaciones web y programas de escritorio.

RA: Realidad Aumentada.

Scriptable Object: en UNITY, se refiere a una clase que permite almacenar grandes cantidades de datos compartidos independientes de instancias de script.

Smart Phone: del inglés, teléfono inteligente.

Software: del inglés, conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Subscribers: del inglés suscriptores, en el contexto de este trabajo, se refiere a los scripts que son objeto de las acciones lanzadas por los *publishers*.

Sprint: del inglés esprint. En informática, Un sprint es un período breve de tiempo fijo en el que un equipo de scrum trabaja para completar una cantidad de trabajo establecida.

TF: Trabajo Final.

TFG: Trabajo de Fin de Grado.

TFP: Trabajo Final Profesionalizador.

UI: en inglés User Interface, se refiere a la interfaz de usuario como un concepto que abarca arquitectura de información, patrones y diferentes elementos visuales que permiten interactuar de forma eficaz con sistemas operativos y softwares de diversos dispositivos.

UOC: Universitat Oberta de Catalunya

Waterfall: del inglés cascada, en el ámbito de los proyectos hace referencia al método que se ha utilizado tradicionalmente para el desarrollo de proyectos de forma secuencial, comenzando con las fases de análisis y diseño y terminando con las de testeo y puesta en producción.

Anexo B: Entregables del proyecto

Nombre	Descripción	Link
Jbeabo AR Game Video.mp4	Video clip de 2' con la sinopsis del juego.	https://drive.google.com/open?id=113VuU_UCRsbT0UwVP3bE4r-f8Mqwsipq&usp=drive_fs
TFG---Videojuegos	Repositorio de GitHub	https://github.com/JoaquimBeaBonet/TFG---Videojuegos
jbeabo_AR_Catch_Care.apk	Ejecutable para Android	https://drive.google.com/open?id=116th6z_i1r-rERuM_OQ_Oya-LPTJDAs-&usp=drive_fs
Video Presentación defensa TFG jbeabo.mp4	Video de 15' con la defensa del TFG	https://drive.google.com/file/d/1Cu_ZP1kHuEWmb_Eg3CnYQ44aXVYOIxwR/view?usp=drive_link

Tabla 5: Entregables del proyecto

Anexo C: Currículum Vitae

Overview:

Exceptional Project Management Leader with a solid track record of leading portfolios and programs. Proven success in effectively defining and implementing project and program management cultures and methods that streamline processes to reduce organizational inefficiencies, promote innovation and deliver strong business results. An advocate for mentoring, coaching, and growing talent. A strategic leader strong in operational management, business strategy, engineering management and project management.

• 20+ years' experience leading Technology and Engineering programs and projects
• 12 years' directly managing a diverse cross-functional global team of people managers, engineering and project management professionals
• Strategy/ Lean / Process improvement/ Agile Transformation / Scaled Agile/ Project Management tools / coaching and development
• Large scale PMO / PMO leadership / Portfolio Management

Executive Summary

Core competency in New Product Generation, from both technical and management roles. Currently, Program Manager within the hp Operations Lab in Barcelona, responsible to optimize the cost of world wide operations. Previously, in the R&D lab, managed the development the new latex-ink-based Hybrid platform in the Mid Volume signage market. This role includes the management of a 80+ team of R&D, Operations and Marketing, plus other external services and partners, overseeing substantial investments. Scope of work includes product co-development with several organizations across different geographies worldwide and external business' partners.

Was in charge of the R&D Vintage Chart Process (2001-2003) and started the Program Management career in year 2003, managing different kind of R&D programs, always growing in terms of Scope, Investment and Return, starting with Designjet Z6100.

In previous roles as people manager (1997-2001), handled responsibilities over Manufacturing Engineering across multi-disciplinary teams (ME, EE, FW, WS) with up to 14 direct reports plus external contractors and outsourced services.

Technical background involves part manufacturing and Quality control, as materials engineer in the hp Large Format Supply Chain (1994-1997), as well as design of ME/EE systems, servo control and sensors, on Parking Brake and Gearshift Systems in FICOSA Intl (1988-1994).

What my managers and peers say about me: fully responsible, with very effective communication skills. Shows full commitment to the project and does not hesitate to take hard calls in the best interest of final customers. Has in-depth understanding of business, market and customer needs, holding and expecting high quality standards. Good judgment skills, excellent planning and execution; strong people leadership, listens well, good learning attitude when exposed to new challenges; autonomy and initiative.

Leadership factors:

- Always Accountable and fully responsible for the Program he manages. Very effective communicator, gives simple messages which are easily understood by all interlocutors. Has earned trust by not hiding “bad news” providing solutions. Assigns high priority to Programs positive Return, reviews NPV assumptions carefully, analyzes deviations and looks for options to improve it.
- Will to Win, his full compromise with the Program inspires the whole team to do the “extra mile” when needed. Does not fear making hard calls, if it is in the best interest of customers. Defines clear plans and drives the team to complete them. Leads the Business Team with an even balance across functions, and allows members freedom of action, while preserving Value Proposition as the #1 objective.
- Passion for Customers, has an in-depth understanding of market and customer needs, gotten from many contacts with LV customers along his NPI experience, uses this knowledge to challenge others and to make decisions. Holds and expects high quality standards and does not fear changing plans or calling for action to maintain product quality.

Professional Experience

- *hp - Large Format Printing, LFPro R&D Lab*
  Operations End to End Cost Program Manager - Jan, 2022 - Present
- *hp - Large Format Printing, LFPro R&D Lab*
  Program Manager - April 2003 - NOW
- *hp - LFP R&D Lab, DesignJet LFP*
  Vintage Chart Process Manager - Jan 2001 – April, 2003

- *hp - LFP Manufacturing Operations*
  Engineering Product Manager - Jan 1997 – Dec, 2000
- *hp - LFP Supply Chain - Manufacturing Operations*
  Materials Engineer - Mar 1994 – Dec 1996
- *Ficosa Intl – R&D*
  Product Engineer - Sept 1988 – Feb 1994

Education

Degree	**Area of knowledge**	**School/University**	**Time**
CS Engineering Degree	Computer Science	Universitat Oberta de Catalunya	2015 –NOW
PMP Certification Training	Project Management	(PMI) Project Management Institute Escuela Internacional de Gestión de Proyectos	2019-2020
Scrum Master Certification	Scrum Master	Simplilearn	2021
DEA - PhD	Project Management	(UPC) Universitat Politecnica de Catalunya	2005-2010
Industrial Engineering Degree	Mechanical Engineering	(UPC) Escola Técnica Superior d'Enginyers Industrials de Terrassa	1982-1987