

App de Juegos para la estimulación de la memoria basados en el entorno del CRMF

Trabajo de Fin de Grado
Ingeniería Informática



VNiVERSiDAD
D SALAMANCA

Septiembre 2023

Roberto Tomé Moro

María Belén Pérez Lancho

Ana Belén Gil González

AUTORIZACIÓN DE DEFENSA

Las Dras. María Belén Pérez Lancho y Ana Belén Gil González, profesoras del Departamento de Informática y Automática de la Universidad de Salamanca

HACEN CONSTAR

que el presente documento titulado “*App de Juegos para la estimulación de la memoria basados en el entorno del CRMF*” ha sido realizado por Roberto Tomé Moro, con DNI 70913989K, y constituye la memoria del proyecto realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Grado en Ingeniería Informática de la Universidad de Salamanca.

Para que así conste a todos los efectos oportunos.

Salamanca, 5 de septiembre 2023

RESUMEN

El proyecto consiste en una aplicación de juegos para la estimulación de la memoria basada en el entorno del Centro de Recuperación de Personas con Discapacidad física (CRMF) de Salamanca. Este sistema, desarrollado en Unity, se divide en dos partes: una aplicación diseñada para dispositivos móviles, enfocado principalmente para tablets y un programa para ordenador. Esta aplicación no solo ofrece juegos interactivos que son beneficiosos desde un punto de vista terapéutico, sino que también incorpora una característica fundamental: la facilidad de personalizar y actualizar el contenido de los juegos sin requerir conocimientos avanzados de informática. Dicha funcionalidad permite a terapeutas o cuidadores adaptar los juegos según las necesidades individuales, lo que no solo mejora la experiencia de los usuarios, sino que también maximiza la utilidad terapéutica de la aplicación. Además, el proyecto incluye un sistema para ordenador que facilita aún más la modificación de los juegos, lo que lo convierte en una herramienta versátil y adaptable para el personal del CRMF.

ABSTRACT

The project consists of a memory stimulation game application based on the environment of the Center for Recovery of people with Physical Disabilities (CRMF) from Salamanca. This system, developed with Unity, is divided into two parts: an application designed for mobile devices, primarily aimed at tablets and a computer program. Not only does this app offer interactive games which are beneficial from a therapeutic perspective, but it also incorporates a fundamental feature: the ease of customizing and updating the game content without the need of advanced computer knowledge. This feature allows therapists or caregivers to adapt the games following the needs of each individual, which not only improves the user experience, but also maximizes the therapeutic purpose of the app. Furthermore, the project includes a computer system that eases the ability of modifying the games even more, which makes it a versatile and adaptable tool for the CRMF staff.

TABLA DE CONTENIDOS

AUTORIZACIÓN DE DEFENSA	2
RESUMEN	3
ABSTRACT	4
TABLA DE CONTENIDOS	5
TABLA DE FIGURAS	8
TABLA DE ABREVIATURAS	10
1 INTRODUCCIÓN	11
1.1 Introducción a la terapia ocupacional	11
1.2 Problema.....	12
1.3 Solución propuesta.....	12
1.3.1 Aplicación móvil	13
1.3.2 Servidor	13
2 TRABAJOS RELACIONADOS	14
2.1 Jigsaw Puzzles - puzzle games	14
2.2 Pasapalabra.....	14
3 OBJETIVOS	15
3.1 Objetivos funcionales	15
3.2 Objetivos no funcionales	15
3.3 Objetivos personales.....	15
3.4 Objetivos sociales.....	16
4 METODOLOGÍA	17
4.1 Patrón de arquitectura	17
4.2 Metodología ágil	17
5 HERRAMIENTAS	19
5.1 Entorno de desarrollo	19
5.1.1 Unity	19
5.1.2 Visual Studio 2022	19
5.2 Otras herramientas.....	19
5.2.1 Microsoft Office Word	19
5.2.2 ChatGPT	19

5.2.3	Photoshop	20
5.2.4	Dall-E 2.....	20
5.2.5	Paint	20
5.2.6	Contrast Finder	21
5.2.7	Microsoft Project	21
5.2.8	Visual Paradigm	21
5.2.9	Doxygen	21
6	ASPECTOS RELEVANTES	22
6.1	Planificación temporal	22
6.2	Análisis.....	22
6.2.1	Especificación de requisitos	22
6.2.2	Casos de uso	23
6.3	Diseño.....	26
6.3.1	Diagrama de clases	26
6.4	Implementación	27
6.4.1	Versión 0.1	27
6.4.2	Versión 0.2	31
6.4.3	Versión 0.3	35
6.4.4	Versión 0.4	38
6.4.5	Versión 0.5	40
6.4.6	Versión 0.6	43
6.4.7	Versión 0.7	46
6.4.8	Versión 0.8	46
6.4.9	Versión 0.9	50
6.4.10	Versión 0.10	52
6.4.11	Versión 0.11	54
6.4.12	Versión 0.12	56
6.4.13	Versión 0.13	57
6.4.14	Versión 1.0	57
6.4.15	Versión 1.1	61
6.4.16	Versión 1.2 (Final).....	61
6.5	Pruebas.....	61
6.5.1	Con muchos niveles.....	62

6.5.2	Nombres incorrectos.....	62
6.5.3	Nombres de usuario extraños.....	63
6.6	Descripción de la aplicación.....	63
6.6.1	Servidor	64
6.6.2	Aplicación móvil	64
6.6.3	Elección de usuario.....	65
6.6.4	Menú principal.....	66
6.6.5	Opciones	67
6.6.6	Selección de nivel	69
6.6.7	Nivel completado.....	70
6.6.8	Rosco	71
6.6.9	Puzzle.....	72
6.6.10	Juego Nombres.....	73
6.6.11	Juego Cartas.....	74
7	CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS	75
	REFERENCIAS Y BIBLIOGRAFÍA.....	77

TABLA DE FIGURAS

Figura 1. Funcionamiento de Scrum	18
Figura 2. Imágenes de Nombres y Puzzle obtenidas de Dall-E 2	20
Figura 3. Imagen editada de Nombres	20
Figura 4. Diagrama de casos de uso	25
Figura 5. Diagrama de clases final	26
Figura 6. Menú principal, primera versión	28
Figura 7. Rosco, primera versión	29
Figura 8. Editor de Rosco, primera versión	33
Figura 9. Puzzle, primera versión	34
Figura 10. Servidor, primera versión	36
Figura 11. Nombres, primera iteración	36
Figura 12. Selección de nivel, primera versión	39
Figura 13. Cambio de colores	41
Figura 14. Elección de usuario	42
Figura 15. Respuesta correcta e incorrecta	43
Figura 16. Botón de nivel aleatorio	44
Figura 17. Cartas, primera iteración, antes de jugar	45
Figura 18. Cartas, primera iteración, durante el juego	46
Figura 19. Fondo nuevo	47
Figura 20. Rosco, versión más legible	48
Figura 21. Selección de nivel con slider	49
Figura 22. Texto de estado de conexión	50
Figura 23. Consejo en selección de nivel	51
Figura 24. Barra de navegación en Puzzle	52
Figura 25. Puzzle con slider de piezas	53
Figura 26. Servidor con cambio de diseño	54
Figura 27. Pantalla de nivel completado	55
Figura 28. Rosco, con fondo transparente y letras con borde	56
Figura 29. Nombres, con imagen seleccionada enmarcada	58
Figura 30. Nombres, sin suficientes archivos	59
Figura 31. Cartas con transparencia	59

Figura 32. Usuario registrado y usuario no registrado.....	60
Figura 33. Consejos de los distintos juegos en selección de nivel.....	61
Figura 34. Archivos con nombres incorrectos.....	62
Figura 35. Archivos con nombres incorrectos, repitiendo número	63
Figura 36. Servidor, versión final.....	64
Figura 37. Elección de usuario, versión final	65
Figura 38. Menú principal, versión final.....	66
Figura 39. Opciones, versión final	67
Figura 40. Opciones, durante una conexión con el servidor	68
Figura 41. Selección de nivel, versión final.....	69
Figura 42. Nivel completado, versión final	70
Figura 43. Rosco, versión final	71
Figura 44. Puzzle, versión final	72
Figura 45. Nombres, versión final.....	73
Figura 46. Cartas, versión final.....	74

TABLA DE ABREVIATURAS

CRMF: Centro de Recuperación de Personas con Discapacidad Física

JSON: JavaScript Object Notation (Formato de archivo que permite almacenar información en texto legible por máquinas y humanos)

UI: User Interface (Interfaz de usuario)

QoL: Quality of Life (Mejoras que facilitan la vida o mejoran la experiencia de usuario)

IDE: Integrated Development Environment (Entorno de desarrollo)

MVC: Modelo Vista Controlador (Un tipo de patrón de arquitectura)

IA: Inteligencia Artificial

1 INTRODUCCIÓN

El siguiente documento constituye la memoria del trabajo de fin de grado del Grado de Ingeniería Informática en la Universidad de Salamanca.

La idea de este proyecto nace en el marco de un acuerdo de colaboración establecido entre la Facultad de Ciencias y el Centro de recuperación de personas con discapacidad física (CRMF-IMSERSO) de Salamanca. Los terapeutas responsables del centro habían identificado diferentes necesidades para el mejor funcionamiento del centro y realizaron varias propuestas de desarrollo de aplicaciones informáticas. La que se ha abordado en el presente trabajo trata de facilitar a sus pacientes una plataforma con la que estimular la memoria, con juegos de preguntas o puzles que puedan adaptarse al nivel de discapacidad y configurarse con contenidos relacionados con la ciudad de Salamanca o con la actividad del propio centro CRMF.

1.1 Introducción a la terapia ocupacional

“La WOFT, Federación Mundial de Terapia Ocupacional, (2012) define a la Terapia Ocupacional como una disciplina que promueve la salud y el bienestar a través del uso terapéutico de la ocupación. El principal objetivo de la Terapia Ocupacional es capacitar a las personas para participar en las actividades de la vida diaria. Los Terapeutas Ocupacionales logran este resultado mediante la habilitación de los individuos para realizar aquellas tareas que optimizarán su capacidad para participar, o mediante la modificación del entorno para que éste refuerce la participación” (1).

Existen diferentes aspectos que se pueden trabajar desde la terapia ocupacional: aspectos físicos, cognitivos o sensoriales. En este caso, la aplicación está orientada a trabajar principalmente la memoria, además de otros procesos psicológicos básicos como la atención, percepción o lenguaje.

“Entre los diferentes tipos de memoria que podemos encontrarnos según el sistema de clasificación de memoria de Tulving y Schacter se encuentran:

- Memoria a corto plazo. Información rápidamente disponible sobre eventos cognitivos recientes.
- Memoria semántica: Conocimiento general del mundo.
- Memoria episódica: Recolección consciente del pasado personal.
- Memoria procedimental: Hábitos y destrezas.
- Sistemas de representación perceptual: Priming” (2).

En los juegos implementados en el proyecto, se trabajan los siguientes campos:

Rosco: Lenguaje y memoria semántica.

Puzzle: Memoria a corto plazo y percepción.

Nombres: Lenguaje y memoria a corto plazo.

Cartas: Atención y memoria a corto plazo.

1.2 Problema

Las aplicaciones que usan los encargados del CRMF para tratar la terapia ocupacional con los pacientes presentan diversas desventajas, como:

- Imposibilidad de adaptar la dificultad de los juegos exactamente a las necesidades del paciente.
- No existe una forma de poder adaptar esos juegos al entorno del paciente, usando imágenes o conceptos que ellos puedan conocer.

De ahí surge este proyecto, en el que se presenta una plataforma con 4 juegos en los que los encargados poseen el control completo sobre el contenido que tiene cada juego.

Sin embargo, surgen tres problemas principales:

- No podemos cambiar los archivos de la aplicación de cada cliente de forma individual al realizar una actualización en los juegos. Para empezar, esto requeriría conocimientos de informática para los encargados de los pacientes, lo cual está especificado que deben ser muy básicos.
- Realizar estos cambios manualmente presentaría un problema de escalabilidad muy grande. Aunque este sistema no está pensado para cientos o miles de usuarios, ya sería mucho trabajo tener que cambiar manualmente los archivos de una decena de usuarios.
- Problema de seguridad. No se debería permitir que los usuarios puedan cambiar esos archivos por error o para hacer trampa en los juegos. Solo deberían poder ser modificados por los encargados del CRMF.

1.3 Solución propuesta

La primera propuesta para el problema de seguridad surge durante la primera reunión con el cliente y sería usar una contraseña para acceder a las opciones de actualización de juegos. No obstante, esta solución queda eclipsada por la siguiente.

La propuesta para solucionar los tres problemas principales es usar una arquitectura cliente-servidor, de esta forma:

- Los encargados no necesitarán conocimientos técnicos de informática, más que abrir un programa y elegir los archivos en cuestión.
- Para actualizar los cambios a un número grande de usuarios, estos solo deberán pulsar un botón y realizar una conexión con el servidor, mejorando la escalabilidad.
- Para que los usuarios modifiquen los archivos deberán tener acceso al servidor en cuestión, lo cual supone una mejora en seguridad.

Por ello se ha decidido dividir el proyecto en 2 partes:

1.3.1 Aplicación móvil

De forma resumida, se trata de una aplicación disponible para Android y Windows, diseñada para *tablets*. Es donde se centra la mayor parte del desarrollo y es donde se puede acceder a los juegos. Tiene una forma de conectar con el servidor para descargar el contenido de los juegos. Está hecho en Unity.

1.3.2 Servidor

Es un programa en Windows pensado para un ordenador de sobremesa que accede a los archivos correspondientes a los juegos. Se basa en un servidor que espera a que se conecte la aplicación móvil. Está hecho en Unity.

2 TRABAJOS RELACIONADOS

Para comenzar a abordar el trabajo propuesto se realizó un análisis de algunas plataformas de juegos comerciales con funcionalidades similares a las que se proponían para esta aplicación.

2.1 Jigsaw Puzzles - puzzle games

Se trata de una aplicación disponible en Google Play Store que presenta una gran cantidad de puzzles con distintas dificultades (3).

Se ha tomado esta aplicación como referencia a la hora de realizar el juego Puzzle, sobre todo en lo que respecta a la presentación de una gran cantidad de piezas en pantalla, que para que sea más fácil y visible para los usuarios, muestra un *slider* que el jugador puede arrastrar para ver todas las piezas o buscar la que desea.

2.2 Pasapalabra

Pasapalabra es un concurso de televisión español que ha estado en emisión desde el 24 de julio del año 2000 (4).

El juego principal en el que se basa este concurso se llama El Rosco, una prueba en la que 2 concursantes compiten en el juego que se detalla seguidamente:

El Rosco recibe ese nombre por la forma circular en la que se presentan casi todas las letras del abecedario. La prueba consiste en una serie de 25 preguntas en las que sus correspondientes respuestas empiezan por una de las letras del abecedario. (Exceptuando la K y la W). De forma que la primera pregunta tiene una respuesta que empieza por A, la segunda por B, etc. Si el concursante no sabe la respuesta puede decir "Pasapalabra" para pasar a la siguiente pregunta y tener otra oportunidad para responder después de dar una vuelta por todo el roscó.

Se ha tomado este famoso programa de televisión como referencia a la hora de diseñar el juego denominado Rosco. La esencia del juego se mantiene, pero por conveniencia se permite a los usuarios iterar bidireccionalmente o, dicho de otra forma, avanzar y retroceder en vez de solo poder pasar a la siguiente pantalla.

Además, no se presentan las letras de manera circular, sino en un bloque en la parte superior de la pantalla para facilitar su lectura y para ahorrar espacio en pantalla.

3 OBJETIVOS

Tras presentar el problema y algunas ideas propuestas, detallamos los objetivos de este trabajo, enumerando los objetivos funcionales, no funcionales y personales planteados.

3.1 Objetivos funcionales

- **Gestión de usuarios:** La aplicación permitirá identificar a los usuarios para que varios de ellos puedan jugar y guardar su progreso de forma independiente dentro de un mismo dispositivo y también a los terapeutas que pueden consultar los datos de uso o modificar los contenidos
- **Modificación de juegos:** Se deberá aportar una manera de que los encargados del centro sean capaces de modificar el contenido de estos juegos de forma sencilla y sin necesidad de poseer conocimientos avanzados de informática.
- **Plataforma de juegos:** La aplicación deberá ser una plataforma que permita jugar a una serie de juegos para trabajar la memoria de forma cómoda y simple para los usuarios.

3.2 Objetivos no funcionales

- **Adaptabilidad a cambios:** Posibilidad de adaptarse a los cambios en la configuración de los juegos sin necesidad de reinstalación en cada dispositivo.
- **Escalabilidad:** Creación de un sistema que permita instalar futuras ampliaciones, como pueden ser la introducción de nuevos juegos o distintas opciones de accesibilidad.
- **Modularidad:** Con el fin de alcanzar los propósitos previamente mencionados de escalabilidad, es esencial descomponer el sistema en elementos individuales capaces de ser sustituidos o alterados sin generar impactos en los demás componentes.
- **Documentación:** Se proporcionará a los responsables del centro una documentación e instrucciones de uso claras del trabajo realizado, sin incluir las partes más técnicas y tratando de que sea fácil de entender para personas con bajo o nulo conocimiento en informática.
- **Usabilidad:** La aplicación debe ser intuitiva y fácil de usar.

3.3 Objetivos personales

- Descubrir nuevas herramientas y adquirir habilidades en el uso, tanto de estas, como de otras conocidas previamente.
- Adquirir experiencia en el diseño y programación de sistemas en Unity, un campo de interés personal.
- Realizar un proyecto desde su fase de conceptualización y análisis de requisitos, hasta el diseño, implementación y pruebas, todo ello empleando una metodología de ingeniería del software apropiada.

3.4 Objetivos sociales

Desde otra perspectiva, el fin de este proyecto es desarrollar un sistema que ayude a los pacientes del CRMF a realizar actividades de terapia ocupacional para facilitar su recuperación, además de ayudar a los profesionales del centro a conseguir adaptar las actividades de esta terapia a las necesidades de los pacientes.

4 METODOLOGÍA

4.1 Patrón de arquitectura

Para este proyecto se ha utilizado un patrón parecido al Modelo-Vista-Controlador, pero adaptado a Unity.

Este patrón consiste en la división de las clases en 3 categorías:

- Modelo, que se usan para almacenar los datos.
- Vista, que son las únicas clases que pueden acceder a la interfaz o a la pantalla en general.
- Controlador, que son las clases intermedias que hacen todos los cálculos y lógica necesaria para el programa.

Con esto en mente, para cada escena distinta se ha creado una clase que actúa de controlador, otra para los botones de la interfaz y, si fuera necesario, clases auxiliares para objetos particulares de la escena (como las clases CajaTexto o PuzzlePiece).

El modelo se ha guardado en una clase general, llamada Player. Como esta clase está instanciada en un componente de un objeto Player, no se puede serializar porque hereda de MonoBehaviour, es decir, que no se puede guardar en memoria. Para esto se ha creado otra clase llamada PlayerData, de forma que el objeto Player tendrá desde su inicio una instancia de PlayerData y esta última podrá ser serializada y guardada.

Las clases NivelRosco, NivelPuzzle, NivelCartas y Pieza son necesarias para la correcta serialización de listas en binario.

4.2 Metodología ágil

“La metodología ágil es un método de ingeniería del software basado en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto. El trabajo es realizado mediante la colaboración de equipos autoorganizados y multidisciplinarios, inmersos en un proceso compartido de toma de decisiones a corto plazo” (5).

Scrum es un marco de trabajo para desarrollo ágil de software caracterizado por:

- “Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados.
- Solapar las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o en cascada” (6).

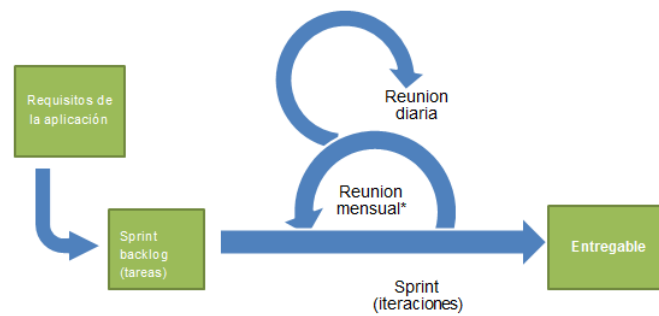


Figura 1. Funcionamiento de Scrum

Para este proyecto se ha elegido usar Scrum, un marco de metodología ágil debido a la inestabilidad de los requisitos y debido a mi inexperiencia en la planificación de desarrollo de software. Este marco de trabajo ha posibilitado desarrollar la aplicación de manera iterativa e incremental realizando reuniones mensuales o bisemanales, aportando *feedback* constante y ajustando el sistema a las necesidades del cliente.

La planificación se puede ver más en detalle en el Anexo I, Plan del proyecto software, incluyendo un diagrama de Gantt y sus correspondientes tareas en Microsoft Project.

5 HERRAMIENTAS

5.1 Entorno de desarrollo

5.1.1 Unity

Para este proyecto se ha usado Unity, un motor de desarrollo de videojuegos con compatibilidad para distintas plataformas que utiliza C#, un lenguaje de programación orientado a objetos desarrollado por Microsoft (7).

El motivo de esta elección fue que hice uso de Unity en las asignaturas Animación Digital y Prácticas Externas, así que esto representaba una ventaja en la curva de dificultad de la implementación. Además, tiene una interfaz fácil e intuitiva, y se adapta fácilmente a plataformas móviles, permitiendo usar *USB Debugging* para hacer pruebas con mi móvil, sin necesidad de usar emuladores de *Android*.

5.1.2 Visual Studio 2022

El entorno de desarrollo (*IDE*) utilizado para la realización de este proyecto ha sido Visual Studio. Se trata de un IDE desarrollado por Microsoft compatible con múltiples lenguajes de programación que permite programar aplicaciones y sitios web (8).

Se ha elegido, entre otros motivos, por las funcionalidades de autocompletado y recomendación de líneas de código, que, junto con una serie de atajos de teclado, agilizan la implementación. Además, tiene una interfaz muy clara y una estética por defecto que, en mi opinión, supera al resto de *IDEs*.

5.2 Otras herramientas

5.2.1 Microsoft Office Word

Word es un software de tratamiento de textos creado por Microsoft (9).

Esta herramienta se ha usado para redactar la memoria del proyecto. Se ha elegido por sus prestaciones de *QoL* o, dicho de otra forma, que facilitan la vida del usuario. Dichas prestaciones incluyen la numeración automática de los apartados, el control de cambios y las tablas de contenidos y figuras.

5.2.2 ChatGPT

ChatGPT es una aplicación de chat basada en inteligencia artificial desarrollada en 2022 por OpenAI. Este chatbot permite, entre otras cosas, solicitar información que se encontrara disponible en internet hasta 2021 o crear fragmentos de código, todo utilizando lenguaje natural (10).

Esta herramienta no se ha usado para la redacción de la documentación, sin embargo, sí se ha usado como ayuda en la programación, por ejemplo, en la generación de la lógica del servidor en lo que respecta a enviar carpetas de forma recursiva.

5.2.3 Photoshop

Photoshop es un editor de imágenes desarrollado por Adobe Systems Incorporated, creado en 1986 (11).

Se ha usado para cambiar el fondo en algunas imágenes a uno transparente.

5.2.4 Dall-E 2

Dall-E 2 es un sistema de inteligencia artificial creado por OpenAI capaz de generar imágenes a partir de una petición en lenguaje natural (12).

Se ha usado para generar las siguientes imágenes para usarse en botones del menú principal.



Figura 2. Imágenes de Nombres y Puzzle obtenidas de Dall-E 2

5.2.5 Paint

Paint es un programa de edición de imágenes desarrollado por Microsoft (13).

Se ha utilizado para realizar pequeños retoques a imágenes en los que no hacía falta utilizar Photoshop, un software mucho más pesado. Un ejemplo de estos retoques es la siguiente imagen, obtenida de Dall-E 2 a la que se le insertó un texto para poder usarse en un botón del menú principal:



Figura 3. Imagen editada de Nombres

5.2.6 Contrast Finder

Contrast Finder es una página web en la que se pueden introducir dos colores y obtener un número que representa el contraste que hay entre estos (14).

Esta página se ha usado para revisar el contraste entre textos y sus respectivos fondos para mejorar la legibilidad, usando un ratio mínimo de 4.5.

5.2.7 Microsoft Project

“Microsoft Project (o MSP) es un software de administración de proyectos y programas de proyectos desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo” (15).

Se ha utilizado este programa para realizar la planificación temporal.

5.2.8 Visual Paradigm

Visual Paradigm es una herramienta que permite construir diagramas UML (16).

Se ha utilizado esta herramienta para diseñar los diagramas de Ingeniería del Software, incluyendo los de clases, casos de uso y despliegue.

Se ha utilizado la versión de la herramienta online o de navegador, por su ligereza y la disponibilidad de modo oscuro.

5.2.9 Doxygen

Doxygen es un generador de documentación para una gran variedad de lenguajes, incluyendo C# (17).

Se ha utilizado esta herramienta para generar la documentación en html y latex de todas las funciones y clases del proyecto, tanto de la aplicación como del servidor.

6 ASPECTOS RELEVANTES

6.1 Planificación temporal

La primera reunión para decidir el tema del proyecto se realizó el 30 de noviembre de 2022.

Por motivos de exámenes, no se empezó a trabajar en este hasta el 3 de febrero del año siguiente, cuando se decidió la siguiente planificación que duraría hasta principios de junio:

- Inicio con la ingeniería del software más básica. Definición vaga de los objetivos y requisitos y planificación de las clases principales.
- Después, aproximadamente 3 horas por semana de implementación o pruebas y 3 horas de refinamiento de análisis y diseño.
- Reunión aproximadamente mensual con las tutoras para comprobar y corregir el progreso y la dirección que sigue el proyecto.

Esta planificación destaca por la poca dedicación que implica en el proyecto, sin embargo, esto se debe a la asistencia a las clases de otras asignaturas y a la falta de tiempo durante el periodo dicho, tras el cual se cambió la planificación a una con mayor empeño, desde junio hasta finales de agosto:

- Reunión semanal o bisemanal con las tutoras.
- Reunión bisemanal o mensual con la clienta, con la que se cambian o se refinan detalles sobre los requisitos y objetivos.
- Aproximadamente 10 horas semanales de implementación o pruebas y 4 horas semanales de análisis y diseño, incluyendo en esto último la redacción de este documento.

6.2 Análisis

En este apartado se revisan resumidamente los requisitos y los casos de uso.

Los requisitos funcionales y no funcionales se encuentran detallados ampliamente en el Anexo I, Plan del proyecto software.

Los casos de uso se encuentran detallados en el Anexo III, Especificación de diseño.

6.2.1 Especificación de requisitos

Entre los requisitos funcionales se encuentran:

Seleccionar juego, Modificar juegos, Juego: Rosco, Juego: Puzzle, Juego: Nombres, Juego: Cartas, Interfaz adecuada, Guardado de progreso, Inicio de sesión y Retroalimentación al usuario.

Los no funcionales son:

Entretenimiento, Desarrollo Mental, Seguridad, Usabilidad, Mantenibilidad y modularidad y Adaptabilidad.

Este es un ejemplo de una tabla de requisito funcional:

RF – 5	Juego: Nombres
Descripción	La aplicación debe proporcionar una forma en la que el usuario pueda relacionar nombres de personas con sus correspondientes imágenes.
Prioridad	Alta
Objetivos asociados	OBJ – 3 Plataforma de juegos.

6.2.2 Casos de uso

Un caso de uso es una representación de como un actor o usuario interactúa con un sistema para lograr un objetivo específico.

6.2.2.1 Diagrama de casos de uso

Para tener una visión clara de cómo interactúan los actores con el sistema, se ha creado un diagrama de casos de uso, una representación gráfica que muestra los casos de uso divididos en paquetes y da una visión general del sistema.

El diagrama está separado por paquetes, que representan las distintas pantallas o escenas de la aplicación.

De manera resumida, las líneas continuas son asociaciones, que representan interacciones entre los actores y los casos de uso.

Las líneas discontinuas son *include* o *extend*. Los *include* representan relaciones en las que un caso de uso forma parte de otro, o utiliza a otro en su ejecución principal.

Los *extend* son relaciones en las que un caso de uso opcionalmente puede utilizar o no al otro.

Para ver el diagrama con mayor resolución se puede usar el siguiente enlace: [Diagrama de casos de uso](#)

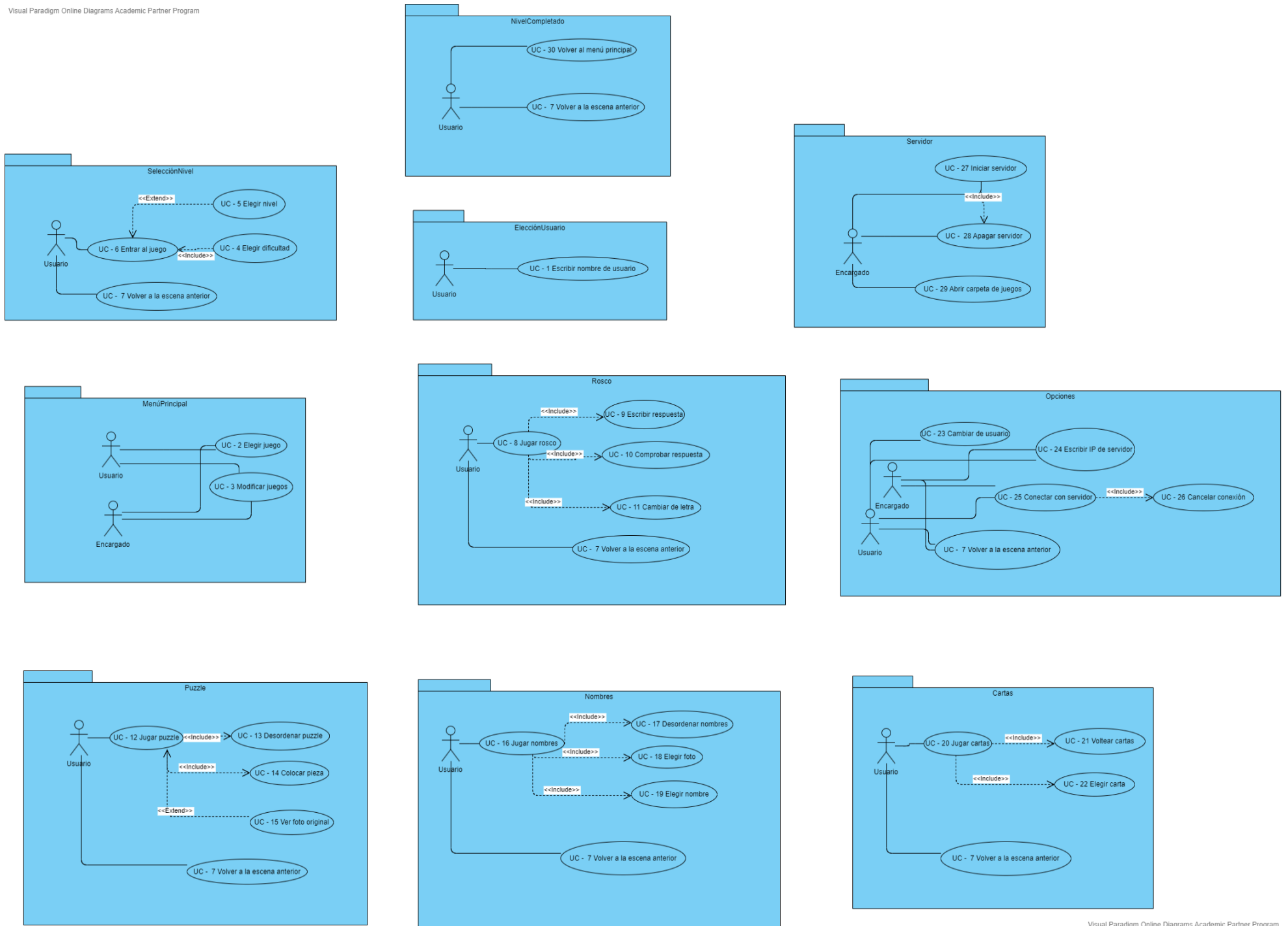


Figura 4. Diagrama de casos de uso

6.3 Diseño

En este apartado se comenta resumidamente el diagrama de clases. Para ver su evolución en detalle, consultar el Anexo III, Especificación de diseño, que también incluye el diagrama de despliegue.

6.3.1 Diagrama de clases

Un diagrama de clases es una representación de la estructura de un programa que muestra las clases, atributos, métodos y relaciones entre objetos.

El diagrama está dividido en paquetes que representan las distintas escenas o pantallas de la aplicación, que a su vez están divididos en clases, que contienen atributos y métodos.

El desarrollo de los diagramas de clases del proyecto se puede ver más en detalle en el anexo III, especificación de diseño. Aquí se puede ver el [Diagrama de clases final](#)

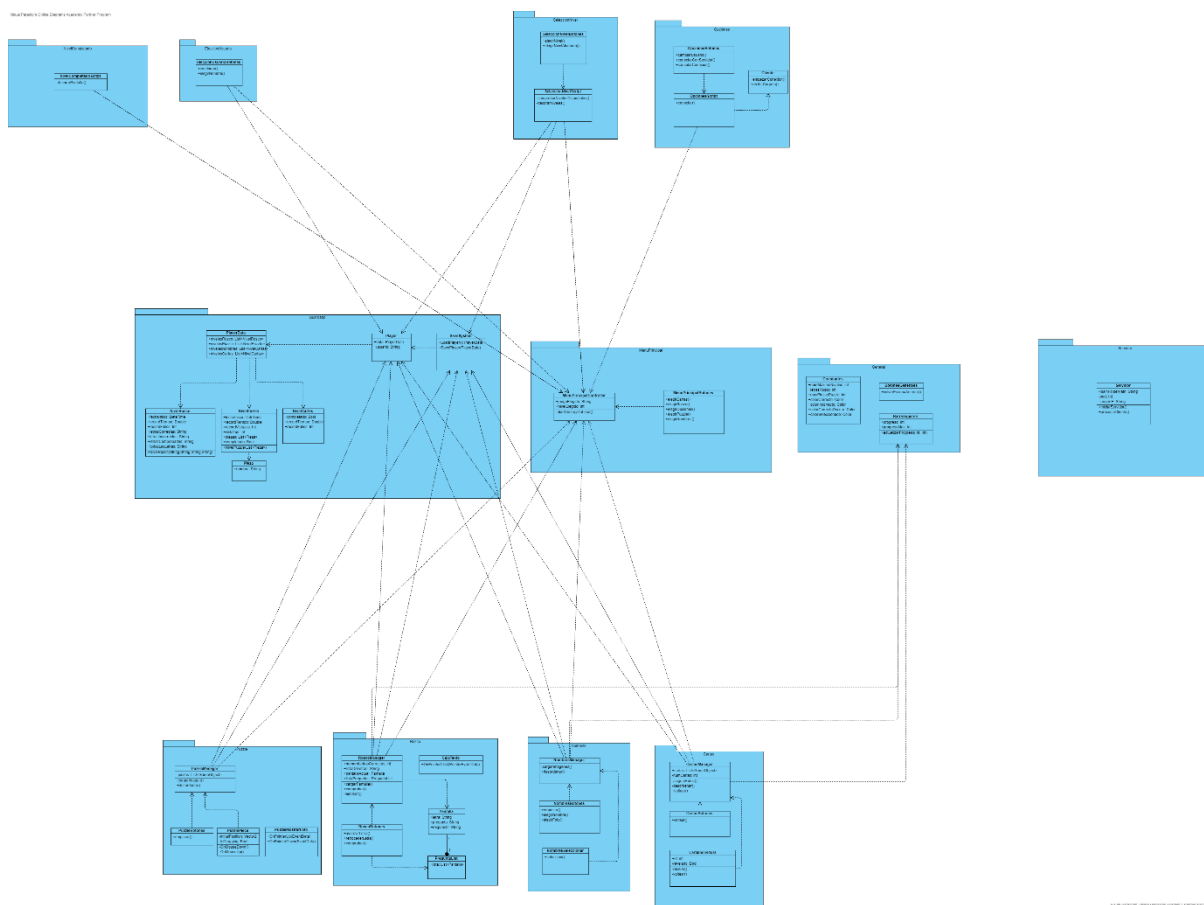


Figura 5. Diagrama de clases final

6.4 Implementación

Para detallar lo ocurrido en la fase de implementación se ha realizado un diario de desarrollo.

En este apartado se detallan los avances en el proyecto, incluyendo las versiones que han sido desechadas o mejoradas, para llegar al producto final.

Cabe destacar que la numeración de las versiones no ha sido planeada de manera temporal estrictamente, sino que se han separado las versiones al hacer copias de seguridad, ya sea tras implementar cambios importantes en el sistema, o antes de tener una reunión.

La numeración de las versiones se ha realizado mediante *Semantic Versioning*, un sistema en el que el primer número representa la versión principal del software, con un 0 significando que es una versión temprana en desarrollo y con un 1 para indicar que la versión ya es algo más o menos estable que se puede entregar o liberar al público. El segundo número indica cambios entre versiones más pequeños.

También merece la pena mencionar que para los juegos se han utilizado imágenes aleatorias que no tienen nada que ver con el proyecto para poder hacer pruebas correctamente hasta conseguir las imágenes proporcionadas por la encargada del centro en torno al final del desarrollo.

La documentación detallada de las clases y métodos del proyecto se puede encontrar en el Anexo IV, Documentación técnica de programación, o en la carpeta Documentación, que dentro de la subcarpeta correspondiente contiene el índice en el archivo index.html.

6.4.1 Versión 0.1

6.4.1.1 Menú principal

En la primera versión se plantea un menú principal que presenta:

- Un botón con símbolo de engranaje que lleve a un menú de opciones que se hará posteriormente.
- 4 botones con los iconos de los distintos juegos.
- 4 botones de elección de nivel.
- Un botón de empezar a jugar.



Figura 6. Menú principal, primera versión

En primer lugar, se ha elegido un fondo azul temporal intentando que parezca a una aplicación de juegos más familiar. La paleta de colores se cambiará en un futuro, posiblemente haciendo que siga los colores del centro CRMF o investigando más a fondo un tono que no provoque fatiga visual.

La interfaz y la ordenación de los elementos también están abiertas a cambios, ya que ahora mismo no se podrían añadir más juegos o niveles de forma que la UI quede limpia y estética.

6.4.1.2 Rosco

Esta es la primera iteración del juego Rosco, en el que se pueden apreciar:

- Un botón para volver al menú principal.
- La letra con la que se está jugando actualmente.
- La lista de letras que hay que completar.
- La pregunta.
- La caja de texto para introducir la respuesta.
- Un botón para comprobar.
- Dos botones para avanzar y retroceder entre letras.



Figura 7. Rosco, primera versión

En primer lugar, se ha elegido esta interfaz y no un Rosco en forma de círculo porque es más fácil de implementar y ocupa menos espacio en pantalla, sin embargo, se plantea la posibilidad de cambiarlo a uno circular en el futuro si se ve necesario.

En cuanto al botón de volver al menú principal, se ha colocado en la esquina superior izquierda porque es un sitio menos accesible y es menos probable que se le dé inintencionadamente. El icono está sujeto a cambios.

La pregunta se coloca justo después de la lista de letras y con un amplio espacio debajo, ya que no se sabe la longitud que tendrá. Se contemplará la posibilidad de añadir un límite de caracteres a las preguntas para que la interfaz no se comporte de manera indeseada.

Después se encuentra una caja de texto en la que no hará falta un límite de caracteres porque permite desplazar el texto para poder leer correctamente.

El botón de comprobar actualmente no realiza ninguna acción visible, solo imprime "Correcto" o "Incorrecto" en la consola.

En la esquina inferior izquierda se encuentra el botón de retroceder entre letras y en la esquina inferior derecha, el de avanzar, que iteran la lista de letras modificando la letra actual.

6.4.2 Versión 0.2

6.4.2.1 Guardado de archivos en JSON

El motivo principal de esta aplicación es que el contenido de los juegos se pueda cambiar fácilmente por un encargado del centro. Para ello, se ha hecho un sistema en el que se puedan guardar los niveles del juego Rosco en formato JSON, para después poder cargarse durante la ejecución del juego, en vez de que las preguntas y las respuestas estén fijadas ya en código por adelantado.

Para ello se crea una clase que permita ser serializable en formato JSON y guarde los elementos necesarios para cada pantalla:

```
[System.Serializable]
public class Pantalla
{
    public string letra;
    public string pregunta;
    public string respuesta;
}
[System.Serializable]
public class PreguntaList
{
    public Pantalla[] lista;
}
```

Después se crea una clase que contenga un array de Pantallas y esta será la que guardaremos en los archivos JSON, este será el estilo que tendrá:

```
{
  "lista": [
    {
      "letra": "a",
      "pregunta": "Color característico de pasamanos, estores y parte
de las paredes del CRMF",
      "respuesta": "AMARILLO"
    },
  ],
}
```

```
{
  "letra": "b",
  "pregunta": "Planta en la que se encuentra la recepción del
CRMF",
  "respuesta": "BAJA"
},
{
  "letra": "c",
  "pregunta": "Estancia del CRMF donde podemos tomar algo y
pedirnos un pincho",
  "respuesta": "CAFETERÍA"
},
{
  "letra": "d",
  "pregunta": "Supermercado con símbolo rojo y blanco del cual
hay uno en la calle del Centro de Salud.",
  "respuesta": "DÍA"
},
...
```

Cabe destacar que, actualmente, el juego es sensible a tildes y mayúsculas. Esto es algo que se debe consultar más adelante con la clienta.

6.4.2.2 Editor de Rosco

El botón de opciones del menú principal lleva a un editor en el que se presentan:

- 5 botones de selección de nivel a editar.
- Una caja de texto en la que escribir el archivo JSON que se quiere guardar.
- Un botón de volver al menú principal.
- Un botón de aceptar y guardar.

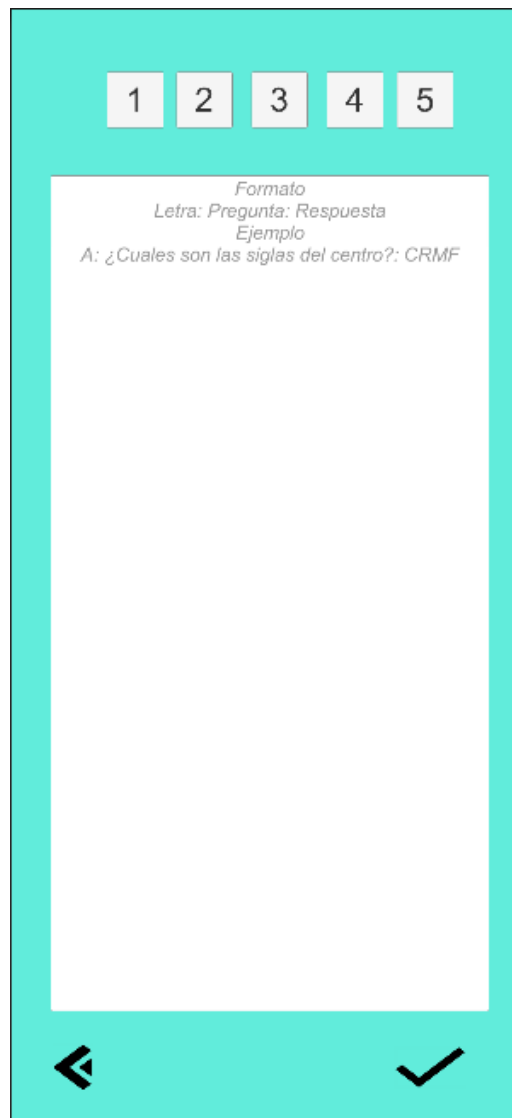


Figura 8. Editor de Rosco, primera versión

En primer lugar, los botones de selección de nivel cambian la caja de texto y cargan el JSON del nivel en cuestión, adaptado para el formato expuesto.

El botón de volver al menú principal está en la esquina inferior izquierda.

La caja de texto permite escribir y el botón de aceptar guarda lo que haya en la caja de texto en el nivel seleccionado.

Actualmente este diseño tiene unos cuantos problemas:

- Cómo gestionar el guardado de un nivel cuando está incompleto.
- Cómo gestionar el guardado de un nivel con un formato incorrecto o sin haber seleccionado un nivel anteriormente.

- Si se escribe en la caja de texto y después se selecciona un nivel, ¿se debería sobrescribir el texto? Esto podría provocar frustración al suceder por error.
- El botón de volver al menú principal está en la esquina inferior izquierda, no sigue mucha consistencia con el Rosco. Se planteará si esto es un problema menor o no.
- Para modificar los juegos de varios usuarios, habría que ir uno por uno cambiándolos desde esta interfaz, lo cual puede ser muy tedioso.

6.4.2.3 Puzzle

Esta es la primera versión del juego Puzzle, en la que se puede ver:

- Un conjunto de piezas.
- Un bloque de huecos para las piezas.
- Un botón de desordenar el puzzle.
- Un botón para mostrar la foto original.

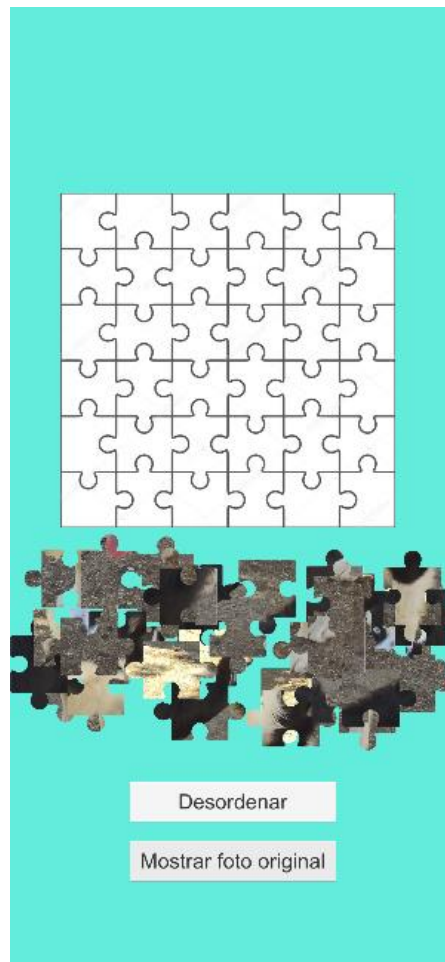


Figura 9. Puzzle, primera versión

Para empezar, el juego consiste en un típico puzzle en el que hay que colocar todas las piezas en su posición original. En primer lugar, el jugador puede ver la foto montada y cuando pulse

el botón de desordenar ya comenzará el juego. Después podrá mover las piezas una a una y al dejar una en la posición correcta, se quedará clavada y no se podrá mover más. En este momento, al completar todo el puzzle solo se imprime un mensaje en consola.

Cabe destacar también que la dificultad está muy abierta a cambios, tanto en lo que se refiere al número de piezas que hay que colocar, como a la opción de mostrar foto original, que podría permitirse solo un número limitado de veces.

Se ha tomado este vídeo como guía (3).

6.4.3 Versión 0.3

6.4.3.1 Reunión con las tutoras

Tras las dos primeras versiones, se realizó una reunión con las dos tutoras, en la cual, principalmente se sacaron las siguientes conclusiones:

- Sería conveniente usar una paleta de colores que corresponda a los colores corporativos del CRMF, para ello habrá que tener una reunión con la cliente.
- En un principio, 4 juegos serían suficientes para el proyecto, ya que, desde un punto de vista competitivo, este no se basa en la calidad o cantidad de los juegos individuales, sino en la capacidad que ofrece para cambiar el contenido de cada juego.

6.4.3.2 Reordenación de carpetas

Esto no tiene mucha relevancia para el producto final, pero puede ser significativo en cuanto a lo que se refiere a facilitar el desarrollo del proyecto.

Inicialmente, se organizaban las carpetas por tipo (*Scripts*, escenas, *sprites*, etc.), como se vio en clase en la asignatura Animación Digital, pero en esta versión se decidió cambiar a un sistema ordenado por juegos, ya que estos están aislados entre sí.

Esta metodología facilitaría en un futuro el desarrollo a nuevos juegos.

6.4.3.3 Servidor

El servidor se ha hecho en otro proyecto distinto de Unity, ya que se tiene que ejecutar en un ordenador en vez de un dispositivo móvil.

Esta es la primera versión de prueba:



Figura 10. Servidor, primera versión

El botón de conectar cliente desaparecerá en la versión final, ya que en este proyecto solo es necesario para hacer pruebas de forma más cómoda.

El botón de iniciar servidor no es necesario ya que este podría iniciarse al ejecutar el programa, pero se considerará si dejarlo o no por mejor claridad para el encargado.

El servidor espera a que se conecte un cliente y cuando ocurre, enviará recursivamente las carpetas y archivos que estén en *ServerFiles* al cliente conectado, mientras que este último lo guardará en Niveles.

6.4.3.4 Juego de Nombres

Nombres es un juego en el que el usuario tiene que memorizar el nombre de cada imagen y después emparejarlos de vuelta. Esta es la primera iteración:

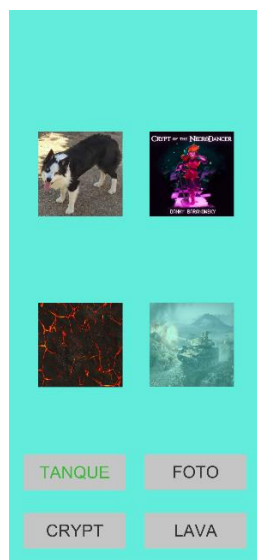


Figura 11. Nombres, primera iteración

Las imágenes y los textos son botones que seleccionan el objeto pulsado y espera a que se pulse el siguiente para comprobarlo. Actualmente, solo funciona si se pulsa primero en el nombre y después en la imagen.

6.4.3.5 Centralización de funciones

Este es otro cambio que tampoco afecta al producto final, pero que facilita el desarrollo. Tras hacer una revisión de los *scripts* de los juegos, consideré que hacía falta coherencia, ya que, algunos botones llaman al objeto mánager principal y este ejecutaba la función en cuestión, mientras que otros botones tenían en su clase funciones con código complejo.

Por ejemplo, en la primera versión la función que comprobaba si la respuesta era correcta:

```
public void botonComprobar()
{
    InputField cajaTexto = FindObjectOfType<InputField>();
    if
(cajaTexto.text.ToUpper() == roscopantallaActual.respuesta.ToUpper())
    {

        Debug.Log("correcto");
        string letraActual =
roscopantallaActual.letra.ToUpper();
        if (!roscopantallaActual.letrasComprobadas.Contains(letraActual))
        {
            texto.text = texto.text.Replace(letraActual,
$"<color=green>{letraActual}</color>");
            letraTitulo.text =
$"<color=green>{letraActual}</color>";
            roscopantallaActual.letrasComprobadas += letraActual;
            roscopantallaActual.letrasCorrectas += letraActual;
        }
    }
    else
    {
        Debug.Log("falso");
    }
}
```

```
        string letraActual =
rosco.pantallaActual.letra.ToUpper();
        if (!rosco.letrasComprobadas.Contains(letraActual))
        {
            texto.text = texto.text.Replace(letraActual,
$"<color=red>{letraActual}</color>");
            letraTitulo.text =
$"<color=red>{letraActual}</color>";
            rosco.letrasComprobadas += letraActual;
            rosco.letrasIncorrectas += letraActual;
        }
    }
    Debug.Log(rosco.letrasComprobadas);
    if(rosco.letrasComprobadas.Length==26)
        cambiarPantalla();
}
```

Esta función estaba localizada en RoscoBotones.cs y provocaba que esta clase tuviera 3 atributos, `rosco`, `texto` y `letraTitulo`, mientras que, de la segunda forma, esta clase solo tiene una referencia a `rosco`, que es el componente manager principal del juego y la clase `RoscoBotones` solo ejecuta esto:

```
public void botonComprobar()
{
    rosco.comprobar();
}
```

Así, el código complejo se encuentra centralizado en `RoscoScript.cs` y de esta forma todo queda mucho más limpio y organizado.

6.4.4 Versión 0.4

6.4.4.1 Cambio de selección de nivel

Tras un par de pruebas, se determinó que el menú no era suficientemente fácil de usar, por eso, se ha optado por utilizar otra pantalla, en la que, después de elegir el juego, se elegirá

el nivel que se quiera jugar. Esta pantalla mostrará el título y el icono del juego, además de los niveles disponibles (8 como máximo, de momento).

La aplicación leerá la carpeta correspondiente al rosco y buscará los ficheros *.json* correspondientes a cada nivel. Los niveles que no tengan un fichero y por lo tanto no se puedan jugar, serán transparentes y sus botones estarán deshabilitados. Además, el color de los números representará el progreso que el jugador lleva con cada nivel, siguiendo la leyenda:

- Botón transparente: El nivel no está disponible para jugar.
- Número negro: El nivel no se ha jugado todavía.
- Número rojo: De 1 a 9 aciertos.
- Número naranja: De 10 a 19 aciertos.
- Número azul: De 20 a 25 aciertos.
- Número verde: Nivel completado.

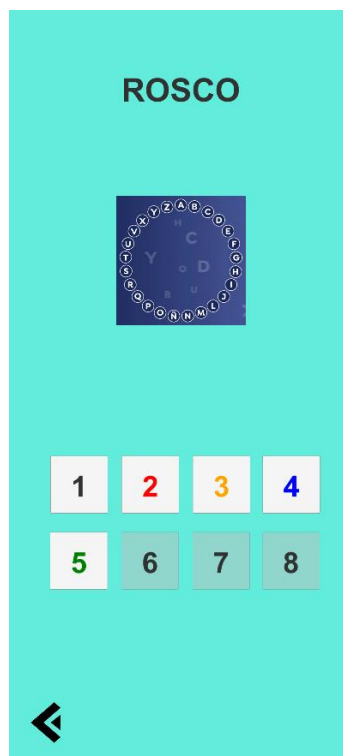


Figura 12. Selección de nivel, primera versión

6.4.4.2 Guardado de progreso en binario

Para mostrar el progreso del usuario de cada nivel y que se quede guardado aun volviendo al menú principal o habiendo cerrado la aplicación, es necesario guardar una serie de variables en memoria secundaria para después cargarlas cuando se necesiten.

Para ello se ha creado una clase *Player*, que sirve como modelo y almacena los datos de los juegos. Actualmente, solo se guarda el avance individual de cada nivel en el juego Rosco, ya que se necesita una cantidad considerable de tiempo para completar un nivel, mientras que en el resto de juegos solo se guarda si el nivel está completado o no, porque no se ve necesario implementar la lógica de carga para juegos que se completen en una cantidad de tiempo de unos pocos minutos.

También se ha creado la clase estática *SaveSystem*, que se puede usar desde cualquier otra clase y sirve para guardar en binario instancias de la clase *Player* en un archivo llamado "player.bin", que se encuentra en la carpeta proporcionada por Unity de *Application.persistentDataPath*, la cual estará en *Appdata/LocalLow* en Windows y en *Android/data* en dispositivos Android.

6.4.5 Versión 0.5

6.4.5.1 Reunión con las tutoras

El día 10/7 se tuvo una segunda reunión con las tutoras en las que se plantearon las siguientes mejoras:

- El progreso mostrado en la selección de nivel no es fácil de entender, ya que, por ejemplo, mostrando el nivel en rojo al tener un solo acierto, parece más un fallo que un acierto. Esto se podría corregir haciendo una leyenda al lado o cambiando la forma de representarlo con una barra de progreso incrustada en el botón del nivel.
- Se debería consultar con la clienta si quiere que el juego Rosco sea sensible a mayúsculas y tildes.
- Hay que consultar con la clienta sobre el color corporativo y la tipografía del CRMF para que se vea reflejado de alguna forma en la aplicación.
- Consultar si en el juego Puzzle sería conveniente mostrar la imagen en pequeño de forma constante.

6.4.5.2 Reunión con la clienta

Tras recibir el *feedback* de la clienta se dejaron en claro las siguientes características que el programa debería implementar:

- El logo del CRMF debería estar en algún lugar del programa.
- Los botones del menú deberían tener esquinas redondeadas para tener un aspecto ligeramente más moderno.
- Se especificó el color corporativo como un verde (con código #1ACA26) y se pidió que este fuera el color del fondo.

- Debería haber distintos niveles de dificultad para los juegos (menos Rosco).
- Debería haber una opción para mezclar niveles o escoger un nivel aleatorio.
- Los archivos de guardado deberían estar separados por usuarios.
- Se concluyó que las respuestas del Rosco no deberían ser sensibles a las mayúsculas, para evitar confusiones con nombres propios, pero sí a las tildes, para alentar una escritura adecuada.
- Se descartó la opción de que en el juego Puzzle se viera la imagen en pequeño porque, al fin y al cabo, se trata de un juego de practicar la memoria.
- El programa debería ofrecer la posibilidad de añadir hasta 20 niveles con contenidos diferentes en cada juego, permitiendo hacer *scroll* si no caben en la pantalla.

6.4.5.3 Cambio de diseño

Esta es segunda iteración del menú principal, tras recibir el *feedback* de la clienta. Se ha cambiado el color del fondo al color corporativo del CRMF, se ha puesto el logotipo del centro en el medio de la pantalla para rellenar un poco el espacio vacío y se han redondeado los botones:

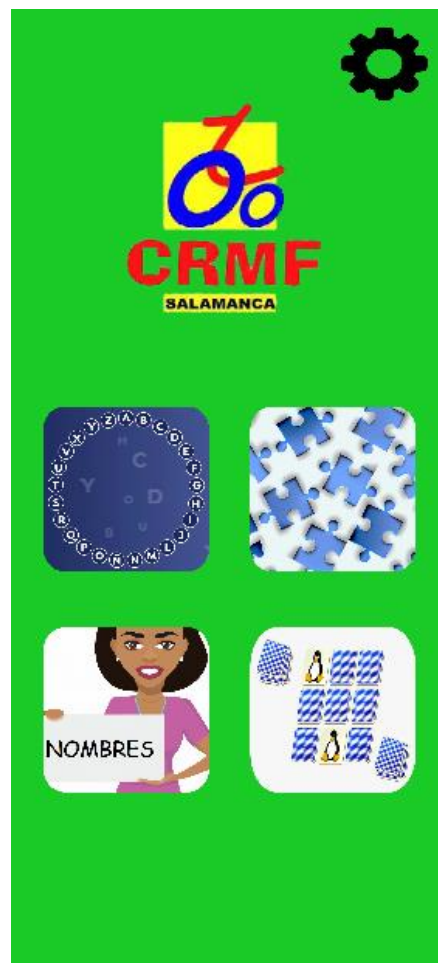


Figura 13. Cambio de colores

6.4.5.4 Pantalla de elección de usuario

Esta pantalla se mostrará siempre al principio al ejecutar la aplicación y su función es pedir el nombre al usuario. Este se usará para el guardado y carga de archivos de progreso de los juegos, cambiando el "player.bin" a "*NombreIntroducidoPorElUsuario.bin*".

Se ha decidido no poner un sistema de contraseñas, porque no es una aplicación que requiera alta seguridad y, además, es altamente posible que los usuarios a los que va dirigida la aplicación tengan problemas de memoria.

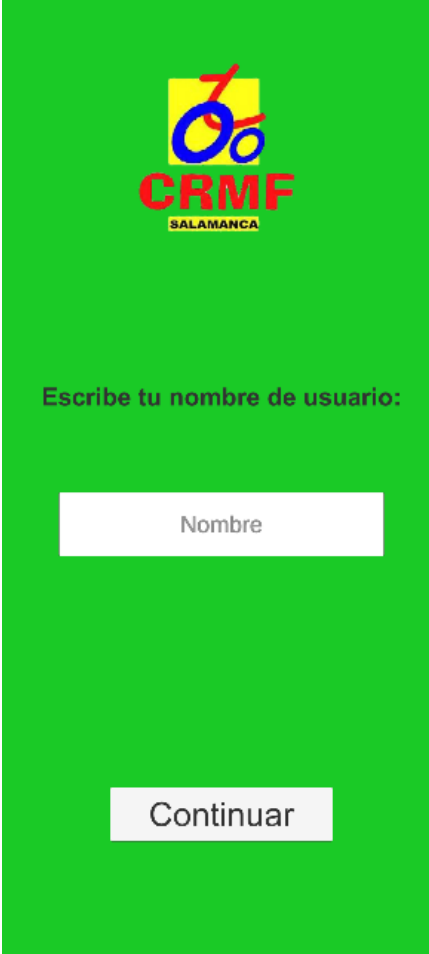


Figura 14. Elección de usuario

6.4.5.5 Pequeños cambios de QoL en Rosco

En esta nueva versión del juego Rosco se realizaron algunas mejoras, en concreto:

- La caja de las respuestas se limpia al cambiar entre pantallas para que no haya que borrar la respuesta anterior cada vez que se quiera escribir una nueva.
- Como las preguntas pueden ser de longitud variable, se muestran con un tamaño de letra adaptado, de manera que cuanto más larga es la pregunta, más pequeño es el texto. De esta forma se evita el *overflow* de texto en pantalla.

- Al pulsar el botón de comprobar en Rosco, ahora la caja de texto en la que se escribe la respuesta se volverá roja o verde durante 1 segundo para dar un *feedback* más claro al usuario de si ha acertado o fallado la pregunta:

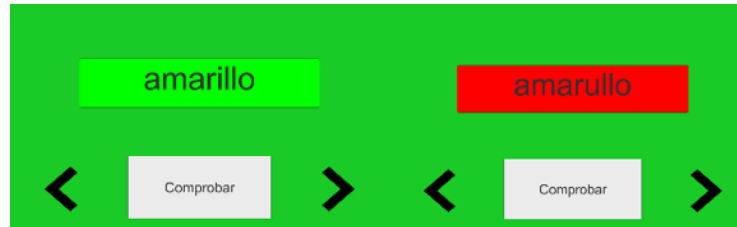


Figura 15. Respuesta correcta e incorrecta

6.4.6 Versión 0.6

6.4.6.1 Botón de nivel aleatorio

Se ha añadido un botón que permita elegir un nivel al azar. En el caso del juego Rosco, en vez de elegir un nivel al azar mezclará preguntas de todos los Roscos existentes creando un nuevo nivel.

Este botón muestra una caja de signos de interrogación.



Figura 16. Botón de nivel aleatorio

6.4.6.2 Selector de dificultad

Para poder adaptar la dificultad de los usuarios a sus capacidades se ha añadido un selector de dificultad, independiente del selector de número de nivel. Existen 3 grados de dificultad: Fácil, normal y difícil.

No tiene sentido que en Rosco haya distintos grados de dificultad porque esta depende intrínsecamente de los niveles, así que el selector se ocultará en su pantalla de selección.

En cuanto a los demás juegos, se han creado 3 escenas por juego para las distintas dificultades. Las dificultades establecidas para cada juego son:

Puzzle, fácil: 3x3 piezas cuadradas

Puzzle, normal: 6x6 piezas con distintas formas

Puzzle, difícil: 10x10 piezas cuadradas

Cartas, fácil: 3x2 cartas

Cartas, normal: 3x4 cartas

Cartas, difícil: 4x4 cartas

Nombres, fácil: 2x2 personas

Nombres, normal: 3x3 personas

Nombres, difícil: 4x4 personas

6.4.6.3 Cartas

Esta es la primera versión del juego Cartas, que muestra una serie de cartas que se ocultarán al pulsar el botón “Voltear” y el jugador deberá encontrar las parejas, pulsando de dos en dos.

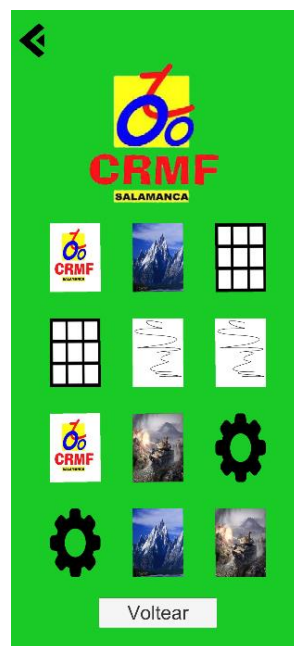


Figura 17. Cartas, primera iteración, antes de jugar



Figura 18. Cartas, primera iteración, durante el juego

6.4.7 Versión 0.7

6.4.7.1 Soporte para Android

Hasta ahora la aplicación solo se estaba probando en Windows por simplicidad, pero en algún momento había que dar el salto y comprobar el funcionamiento del programa en un móvil Android no emulado.

Para ello ha habido que mover la carpeta en la que se guardan los niveles a la misma en la que se guarda el progreso de cada usuario e implementar la funcionalidad del botón de retroceso en Android.

6.4.8 Versión 0.8

6.4.8.1 Reunión con clientea

Para analizar y probar las modificaciones de interfaz mencionadas se realizó una nueva reunión con la responsable del centro en la que se plantearon nuevas mejoras a realizar:

- El fondo fatiga la vista así que debería cambiarse, si es posible por uno con un degradado o un patrón en vez de una imagen plana.
- El botón de volver hacia atrás debería devolver al jugador a la escena anterior y no siempre al menú.
- Algunos textos no son bien legibles así que deberían tener un borde más grueso o un fondo distinto.

- La aplicación debe estar adaptada para tablet, en vez de móvil. Se tomará como referencia la tablet del centro, con una resolución de 1824x2736.
- Si es posible, al hacer *click* en cajas de texto debería aparecer el teclado en pantalla, porque la tablet del centro tiene un sistema operativo Windows.
- La forma de presentar todas las piezas de puzzle desordenadas puede ser muy agobiante, se ha de contemplar una forma de arreglar este aspecto.

6.4.8.2 Adaptación a tablet

Se han reordenado todas las interfaces para distribuir mejor el espacio de forma que se vea correctamente en dispositivos con pantalla más ancha que un móvil.

De ahora en adelante no se tendrá como una prioridad que la aplicación se muestre correctamente en móviles, pero si es posible y no requiere mucho tiempo de desarrollo se intentará.

6.4.8.3 Cambio de fondo

Tras el *feedback* de la reunión se enviaron a la clienta distintos fondos y se decidió que el que se muestra en la siguiente figura es el más adecuado para la aplicación, porque inspira tranquilidad y no sobrecarga la pantalla:



Figura 19. Fondo nuevo

6.4.8.4 Corrección de legibilidad de textos

Como se puede ver en la figura anterior y en la siguiente, se ha colocado un fondo blanco detrás de algunos textos para mejorar su legibilidad. El color es todavía temporal y probablemente se juegue con la transparencia para mejorar su estética.



Figura 20. Rosco, versión más legible

6.4.8.5 Scroll de niveles

En la pantalla de selección de nivel se ha añadido la zona de *scroll* solicitada para que se puedan añadir hasta 20 niveles. Las flechas laterales están para dar pistas al usuario de que hay más de 5 niveles, porque sin ellas sería altamente probable que los usuarios no se dieran cuenta de que es posible hacer *scroll*.

El fondo blanco transparente indica al usuario la zona en la que se pueden arrastrar los niveles.



Figura 21. Selección de nivel con slider

Se ha usado el siguiente vídeo como guía (18).

6.4.8.6 Cambios en pantalla de opciones

Como es incómodo tener que salir de la aplicación para poder cambiar de usuario, ahora la pantalla de opciones presenta una opción que lleva a la escena de cambio de usuario.

Además, se ha intentado mejorar la información que se le da al usuario cuando se intenta conectar al servidor para que sepa qué está pasando en cada momento, mostrando mensajes en pantalla con relación al estado de la conexión, como: “Conectando...”, “No se ha podido conectar con el servidor” o “Conexión realizada con éxito”.

También se oscurece la pantalla como indicador de que el usuario no debe tocar nada mientras se conecta y se ha añadido un botón que permite cancelar la conexión.

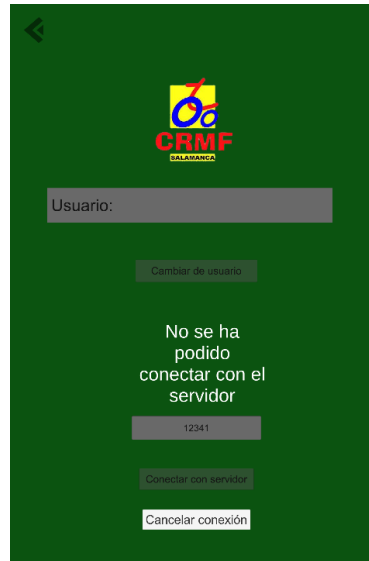


Figura 22. Texto de estado de conexión

6.4.9 Versión 0.9

6.4.9.1 Funcionalidad de flechas en selección de nivel

Las flechas en selección de nivel han servido para dar la indicación de que hay más de 5 niveles, pero también daban la falsa sensación de que se pueden pulsar para mostrar el resto de niveles, por eso se ha hecho que esas flechas no sean decorativas y se pueda pulsar en ellas para avanzar o retroceder 5 niveles.

6.4.9.2 Ayudas y Consejos

Se han añadido pequeños mensajes que indican lo que hay que hacer en los juegos en los que se ha visto necesario, además de en la pantalla de selección de nivel y menú principal. Aunque se pudiera ver como información redundante o sobrante, esto puede ayudar a los usuarios a no sentirse desorientados usando la aplicación y, sobre todo, es importante recordar que dichos usuarios tendrán en su mayoría problemas de memoria de cierto grado.

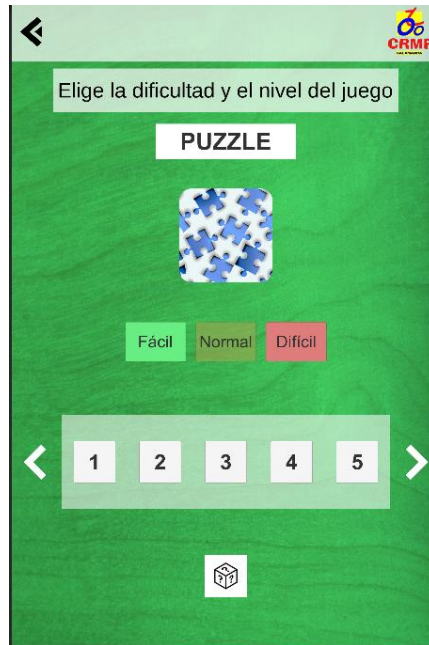


Figura 23. Consejo en selección de nivel

6.4.9.3 Barra de navegación

Se ha añadido una barra de navegación en la parte superior de la pantalla, la cual ayuda a uniformizar todas las pantallas ligeramente y muestra información relevante cuando sea necesaria. En esta barra se encuentran por orden de izquierda a derecha: El botón de retroceso, el icono y el nombre de la pantalla actual si esta es un juego, el progreso de la partida actual y el icono del CRMF.

Cuando el progreso es actualizado, se muestra una animación en la que el texto brilla en verde durante dos segundos.

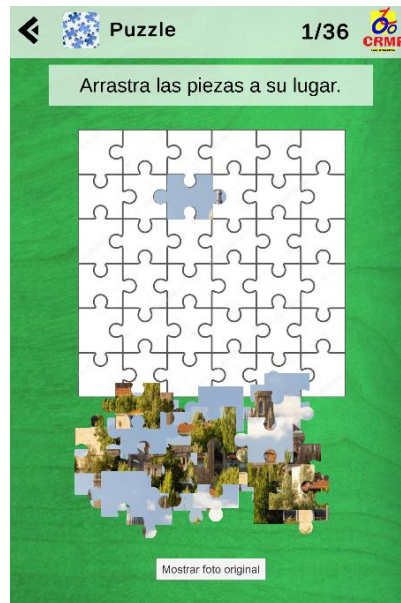


Figura 24. Barra de navegación en Puzzle

6.4.9.4 Botón de retroceso

Ahora el botón de retroceder lleva a la última pantalla y no siempre al menú. Además, ahora también funciona el botón de retroceder propio de los dispositivos Android.

Se ha usado el siguiente hilo como ayuda (19).

6.4.10 Versión 0.10

6.4.10.1 Cambios en Puzzle

Previamente no se vio necesario implementar un sistema de guardado para el estado de la partida del juego Puzzle porque una partida no debería durar más de 5 minutos, pero actualmente, con la nueva dificultad de 100 piezas, sí se ve necesario, por lo que se ha implementado.

Adicionalmente, tomando la aplicación “Jigsaw Puzzles - puzzle games” como referencia, ahora las piezas en lugar de colocarse aleatoriamente, esparcidas por la pantalla, se muestran en una barra de *scroll*, de forma que se presentan de modo mucho más uniforme. Aun así, todavía se permite al jugador sacar las piezas de la barra y desordenarlas para que este pueda organizarlas a su manera.

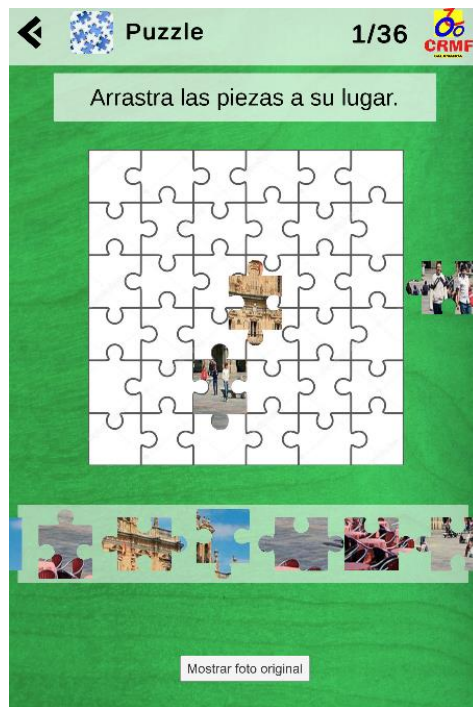


Figura 25. Puzzle con slider de piezas

6.4.10.2 Sonidos

En esta fase del desarrollo se han añadido sonidos para mejorar la calidad general de la aplicación, pero, sobre todo, para aumentar la satisfacción al progresar en los juegos y para mejorar el *feedback* al usuario después de que este realice cualquier acción.

6.4.10.3 Cambios en Servidor

Se ha aplicado el cambio de diseño también a la pantalla del servidor, junto con un par de cambios:

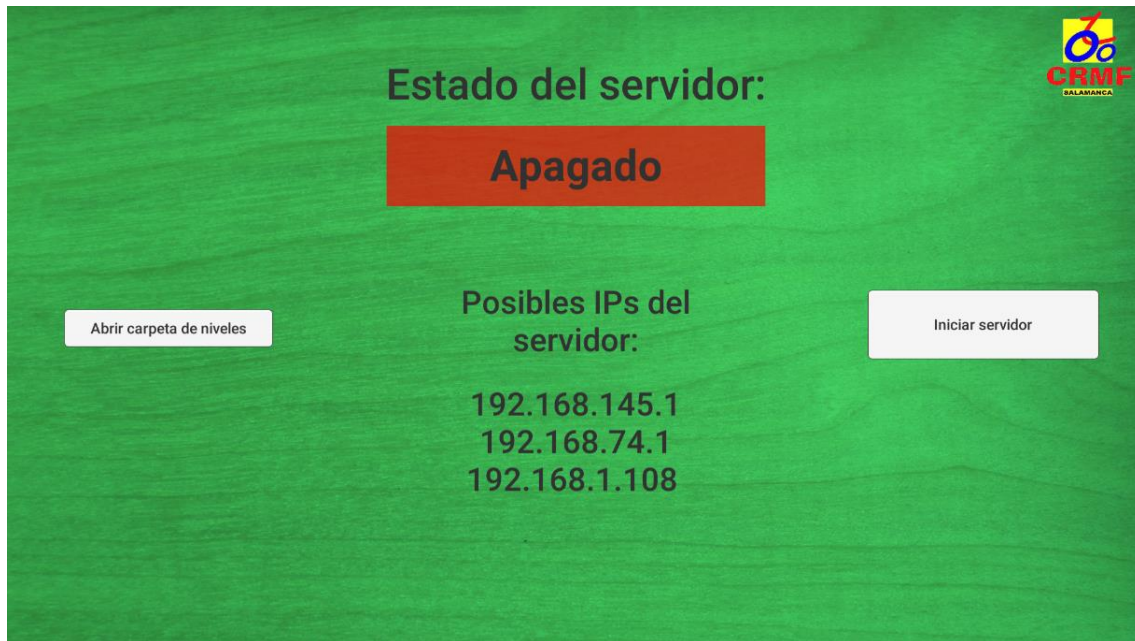


Figura 26. Servidor con cambio de diseño

Además del fondo, se ha puesto un indicador de si el servidor se encuentra apagado, encendido o conectando, con sus correspondientes colores rojo, verde y amarillo.

También se muestran las IPs de las interfaces del ordenador. El texto se llama posibles IPs porque el ordenador puede tener más de una interfaz, estando conectado por wifi y ethernet.

Asimismo, se ha añadido un botón que abre la carpeta que contiene los archivos modificables, en el explorador de archivos de Windows.

6.4.11 Versión 0.11

6.4.11.1 Pantalla de nivel completado

Se ha creado la siguiente escena para felicitar al jugador al completar al nivel y mostrar de forma resumida sus resultados. Se muestra el icono del juego de forma muy visible, en el centro y a la izquierda de la pantalla, a su derecha aparece el nivel y la dificultad elegidos y abajo se detallan el número de fallos que se han cometido en la partida y el tiempo que el jugador ha tardado en pasarse el nivel. Si la nueva puntuación es un récord, se muestra en rojo para llamar la atención del usuario.



Figura 27. Pantalla de nivel completado

6.4.11.2 Cambio de tipografía

Se ha escogido la fuente Roboto para todos los textos de la aplicación, alternando entre Roboto-medium para textos cortos y Roboto-light para textos largos. El motivo de haber elegido esta fuente y no otra es porque es ampliamente usada por Google y muy común en aplicaciones de Android, además es gratis y Google proporciona los *assets* necesarios para implementarla fácilmente.

También se ha cambiado el *sprite* del botón de retroceso a uno que representa mejor su función.

Se ha usado el siguiente tutorial para el cambio de fuente (20) y los consejos mencionados en la siguiente guía (21) .

6.4.11.3 Cambios en Rosco

Para mejorar el aspecto de la interfaz de este juego se ha puesto un borde negro grueso al listado de letras en el Rosco para que cuando estas cambien de color a verde o rojo, sigan siendo fácilmente distinguibles del fondo. En la caja de texto de la respuesta, en vez de ponerse en rojo o verde al responder una pregunta, ahora se muestra una animación en la que se torna del color correspondiente y vuelve al blanco a lo largo de dos segundos.

El texto de la pregunta se ha enmarcado dentro de un rectángulo con una ligera transparencia para facilitar su lectura y se ha puesto el tamaño de letra variable, dependiendo de la longitud de la pregunta.



Figura 28. Rosco, con fondo transparente y letras con borde

6.4.12 Versión 0.12

6.4.12.1 Pequeños arreglos

Ahora todos los nombres de usuario se guardan en mayúsculas para evitar confusiones en las que un usuario podría no acordarse de qué nombre puso exactamente.

El tiempo que se muestra al completar un nivel de Rosco o Puzzle depende de cuando se inició el nivel, de forma que ya no se puede dejar una pieza o una pregunta para el final y salir y entrar de la aplicación para completar el nivel con un tiempo récord.

6.4.13 Versión 0.13

6.4.13.1 Pequeños arreglos y cambios de QoL

(QoL o Quality of Life se refiere a cambios de menor magnitud que se hacen en un programa para mejorar la experiencia de usuario).

Se ha cambiado un parámetro del canvas para que la aplicación se muestre correctamente en móvil y en distintas resoluciones.

Al conectar con el servidor, ahora muestra en cada momento el número de archivos recibidos para mostrar un mejor *feedback* de si la conexión se está realizando con éxito o no.

Cuando no hay niveles cargados porque no se han descargado del servidor, ahora se muestra un consejo en el que se pide al usuario que vaya a opciones y hable con un encargado para descargar los niveles.

Se ha corregido un error en el que, en dispositivos con menor rendimiento, las piezas de puzzle no guardaban su lugar correctamente por un error de concurrencia.

6.4.14 Versión 1.0

6.4.14.1 Cambios en Nombres y Cartas

Después de hacer un par de pruebas, se concluyó que hacía falta poner énfasis en el elemento seleccionado actual para evitar confusiones. Por eso se ha puesto un marco dorado alrededor de la imagen o nombre seleccionado en Nombres y se ha hecho lo mismo en Cartas.

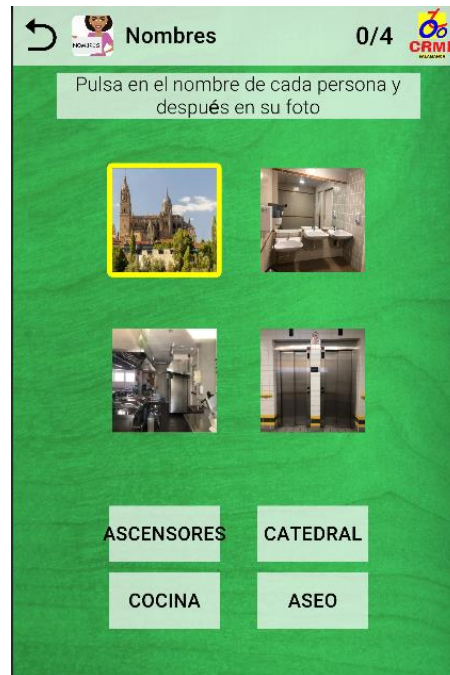


Figura 29. Nombres, con imagen seleccionada enmarcada

Adicionalmente, anteriormente en el juego Nombres solo se permitía relacionar las parejas haciendo click primero en el texto y luego en la imagen, mientras que ahora se puede hacer de las dos maneras.

También, por si no fuera obvio, se ha puesto un consejo en el caso de que no haya suficientes imágenes para la dificultad elegida.



Figura 30. Nombres, sin suficientes archivos

Se ha decidido permitir jugar el nivel aún con insuficientes fotos por casos en los que solo pudiera faltar una o dos fotos para llegar al número máximo.

Por último, en el juego Cartas, al elegir correctamente una pareja, esta se vuelve transparente para que no se confunda con las que hay todavía disponibles en pantalla.

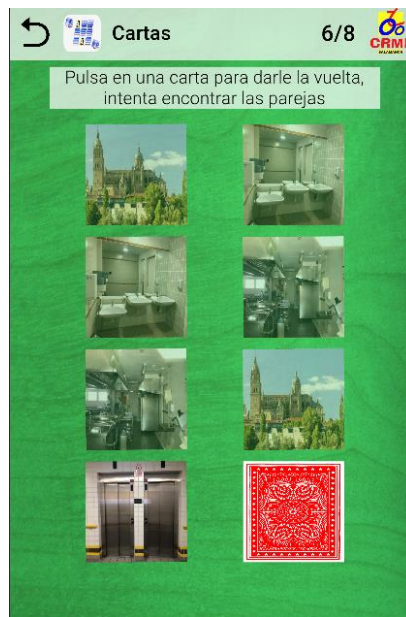


Figura 31. Cartas con transparencia

6.4.14.2 Log de eventos

No se había especificado como requisito pero, por si fuese necesario, se ha hecho un log en el que se almacenan sucesos como los siguientes en el siguiente formato:

2023-08-22 02:03:07 ROBER: Inicia sesion

2023-08-22 02:03:10 ROBER: Entra al juego rosco, nivel 3, dificultad Normal

2023-08-22 02:03:19 ROBER: Entra al juego puzzle, nivel 1, dificultad Facil

2023-08-22 02:03:31 ROBER: Completa el juego puzzle, nivel 1, dificultad Facil, con 0 fallos y tardando 11,0 segundos

6.4.14.3 Cambio de guardado de tiempo en Rosco y Puzzle

Como en estos dos juegos se puede salir y volver a entrar, manteniendo el progreso del nivel, se decidió en un principio almacenar la fecha en la que se inició el nivel y mostrar el tiempo que se tardó en completar, incluyendo el tiempo transcurrido fuera del juego.

Ya que esto no resulta muy justo a la hora de comparar puntuaciones, se ha decidido guardar solamente el tiempo transcurrido con el juego abierto en la pantalla. Para ello se guarda el progreso automáticamente cada cierto tiempo (10 segundos en un principio).

6.4.14.4 Cambios de QoL

Como es fácil que un usuario pueda equivocarse en una letra de su nombre y después preguntarse por qué no se ha cargado el progreso que había hecho, se ha puesto un mensaje al dejar de escribir en Elección Usuario que diga si el nombre escrito es nuevo o ya está registrado.



Figura 32. Usuario registrado y usuario no registrado

Además, en la pantalla de Selección Nivel, se ha cambiado la palabra “nivel” por otras dependiendo del juego elegido. Esto se ha hecho porque dicha palabra puede llevar a la equivocación de que distintos niveles conllevan distintas dificultades, cuando no es así.



Figura 33. Consejos de los distintos juegos en selección de nivel

Por último, en Rosco, aunque se decidió que el juego debía ser sensible a tildes, se ha decidido que, sin embargo, no sea sensible a espacios ni símbolos como comas, puntos o guiones, ya que un pequeño error humano a la hora de escribir las respuestas podría conllevar a preguntas irresolubles.

6.4.15 Versión 1.1

Se han arreglado los aspectos comentados en el apartado de pruebas. De forma resumida, los archivos se ordenan correctamente según los números que contengan en el nombre y en el nombre de usuario se eliminan caracteres prohibidos en nombres de archivos (<, >, :, ;, ", /, \, |, ?, *)

6.4.16 Versión 1.2 (Final)

En esta última versión se ha limpiado el código y se han añadido los comentarios necesarios para generar la documentación correctamente mediante Doxygen.

Los comentarios siguen este formato:

```
/// <summary>
/// Asigna fotos aleatorias a las cartas en el juego.
/// </summary>
/// <param name="info">Información del directorio de las
fotos.</param>
/// <returns>Lista de archivos de fotos asignados a las
cartas.</returns>
private List<FileInfo> asignarFotosAleatorias(DirectoryInfo info){}
```

6.5 Pruebas

Desafortunadamente, durante el desarrollo del proyecto, los usuarios para los que está dirigida la aplicación no están en el centro (vendrían en septiembre, cuando finaliza la entrega), así que no se han podido realizar pruebas de uso con usuarios reales.

Por ello, se pueden tomar como pruebas las reuniones comentadas en el apartado anterior de implementación, en las que se enseña el prototipo actual y se discuten fallos, aciertos y posibles mejoras.

Además de las comentadas en el anterior apartado se pueden añadir las siguientes pruebas realizadas en torno al final del desarrollo, enfocadas a cómo se comportaría la aplicación después de no seguir el procedimiento indicado en las instrucciones de usuario, como puede ser, por ejemplo, usando formatos inadecuados o nombres distintos a los indicados.

6.5.1 Con muchos niveles

Se ha probado a jugar al Puzzle con 11 niveles y se ha descubierto un error en el que los parámetros de los últimos 10 niveles estaban mal puestos y, por lo tanto, eran inaccesibles.

Además de este error, se ha encontrado otro en el que, al usar orden alfabético, el nivel Puzzle10 iría antes de Puzzle2, lo que desordenaría los niveles. Por ello se ha cambiado la forma de ordenación a una dependiente solo de los números de los archivos.

Con esto arreglado, se ha probado también con más archivos que el máximo número de niveles (20) y estos archivos simplemente se ignorarán, por lo que no hay problemas.

6.5.2 Nombres incorrectos

Se ha testado el puzzle con los siguientes archivos:

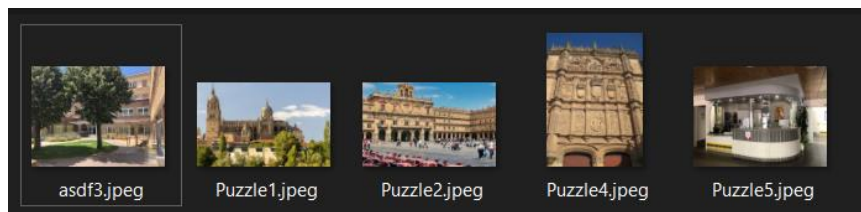


Figura 34. Archivos con nombres incorrectos

La conclusión obtenida es que el orden de los niveles solo depende del número que tengan, así que no afecta al resultado.

Sin embargo, usando el siguiente conjunto de archivos:

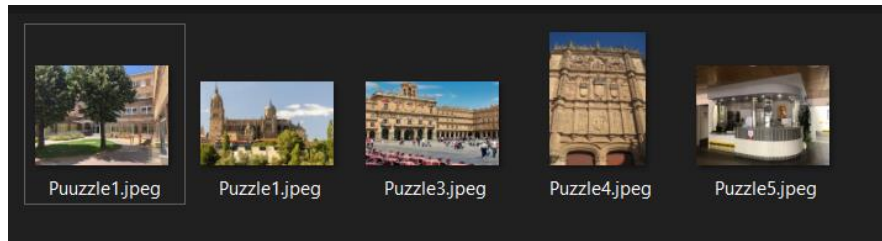


Figura 35. Archivos con nombres incorrectos, repitiendo número

En este caso lo que ocurre es que, al usar dos niveles con el mismo número, todos los demás se desordenarán aumentando su posición en 1, por ello habrá especial énfasis en el manual de usuario con respecto a la nomenclatura.

6.5.3 Nombres de usuario extraños

Se ha probado a utilizar los siguientes nombres:

(vacío), “ñ”, “#”, “ ”, “¢”, “ll=”.

El carácter “ ha dado fallos debido a que no se pueden crear archivos con dicho carácter en Windows, así que, para evitar posibles problemas, se ha decidido eliminar de los nombres de usuario todos los caracteres que no soporta Windows en los nombres de archivo, que son:

<, >, :, ;, /, \, |, ?, *

6.6 Descripción de la aplicación

En este apartado se detalla la totalidad del proyecto en su versión final.

Se divide en dos partes, un programa para Windows, pensado para ordenador y otro para Android o Windows, pensado para tablet.

6.6.1 Servidor



Figura 36. Servidor, versión final

Se trata de un programa que, principalmente, permite encender un servidor que espera a que se conecte el cliente desde la otra aplicación.

A la izquierda hay un botón que abre en el explorador de archivos la carpeta que contiene los archivos necesarios para los juegos.

A la derecha hay un botón que enciende el servidor y que, cuando este está encendido, permite volver a apagarlo.

En la parte superior se muestra el estado del servidor: Apagado, Encendido o Conectando, respectivamente en rojo, verde y amarillo.

En la parte inferior se muestran las posibles IPs del servidor para que se pueda conectar el cliente de la aplicación. Puede haber una o más dependiendo de si el ordenador tiene conexión ethernet, wifi, etc.

6.6.2 Aplicación móvil

Se trata del programa principal del proyecto, se divide en las siguientes escenas:

6.6.3 Elección de usuario



Figura 37. Elección de usuario, versión final

Esta escena es la primera que aparece cuando el usuario inicia la aplicación. Aquí podrá poner su nombre de usuario para que su progreso se almacene de manera independiente al resto de usuarios.

En caso de que ya haya un usuario registrado con ese nombre, se mostrará en la parte superior "Usuario registrado" y en caso contrario, "Usuario nuevo", para minimizar errores humanos.

Al pulsar en "Continuar", se avanza al menú principal.

6.6.4 Menú principal

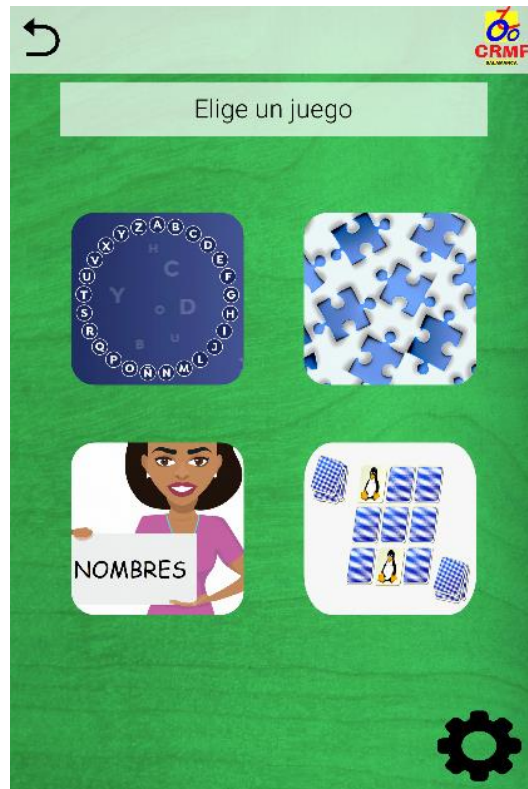


Figura 38. Menú principal, versión final

En esta escena el usuario puede elegir uno de los 4 juegos disponibles situados en el centro de la pantalla, o la escena de opciones, situada en la esquina inferior derecha.

Hacer *click* en uno de los juegos llevará a la pantalla de selección de nivel.

Cabe destacar que en todas las pantallas a partir de la actual se muestra un botón de volver a la escena anterior en la esquina superior izquierda.

6.6.5 Opciones

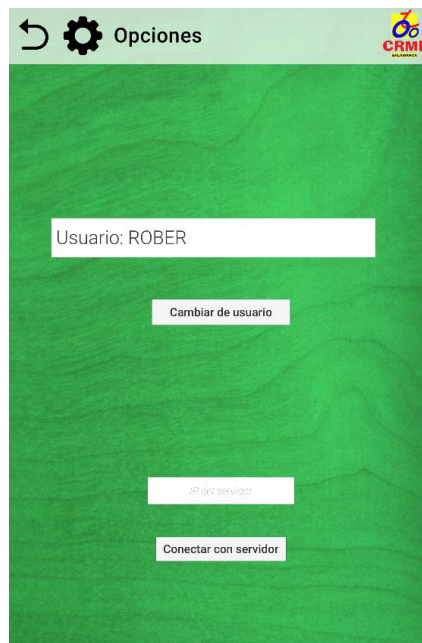


Figura 39. Opciones, versión final

Esta es la pantalla en la que el usuario o un encargado puede conectarse con el servidor, poniendo la IP de este en la caja de texto en la mitad inferior de la pantalla y pulsando en “Conectar con servidor”.

Mientras se esté realizando la conexión, la pantalla se oscurece y se deshabilitan los botones, excepto uno que aparece en la parte inferior de la pantalla que cancela la conexión.

Adicionalmente se muestra un mensaje informando si se ha encontrado el servidor o no y el estado del envío de los archivos.

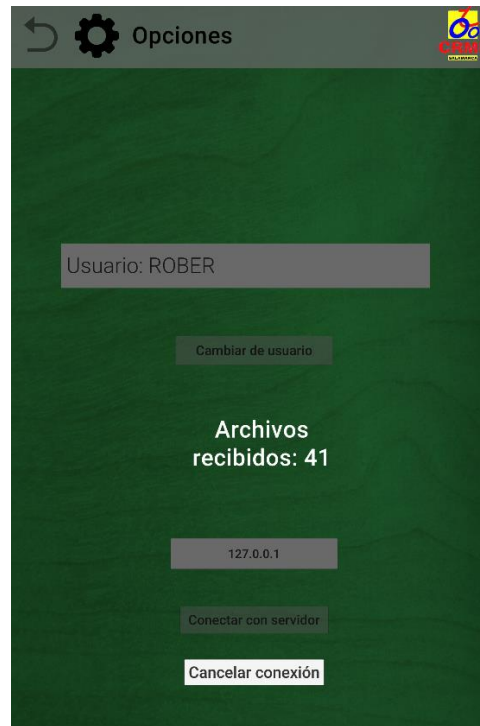


Figura 40. Opciones, durante una conexión con el servidor

Además, también se puede cambiar de usuario pulsando el botón central “Cambiar de usuario”, volviendo a la pantalla de elección de usuario.

6.6.6 Selección de nivel

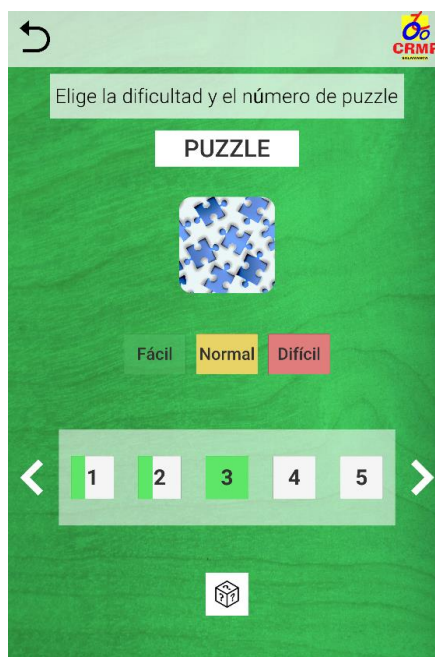


Figura 41. Selección de nivel, versión final

En esta pantalla el usuario puede elegir la dificultad y el nivel del juego.

Entre las dificultades se encuentran “Fácil”, “Normal” y “Difícil”, cuyos cambios se detallan más adelante en cada juego.

Los niveles representan distinto contenido jugable que no implica cambios de dificultad, es decir, en Puzzle cambian la imagen, en Rosco cambian el conjunto de preguntas y en Cartas y Nombres el conjunto de imágenes.

En la parte superior se muestra el nombre y el icono del juego, elegidos anteriormente en el menú principal.

El usuario puede elegir uno de los niveles disponibles (20 como máximo) haciendo click en uno de los botones numerados en la mitad inferior de la pantalla. Para acceder al resto de niveles a partir del quinto, es posible deslizar o hacer click en las flechas laterales.

En la parte inferior central hay un botón que escoge un nivel aleatorio de los que estén disponibles.

6.6.7 Nivel completado



Figura 42. Nivel completado, versión final

Esta pantalla se muestra cuando el usuario ha finalizado de forma exitosa un nivel.

En la mitad superior de la pantalla se muestra el juego, nivel y dificultad elegidos, mientras que en la mitad inferior se muestra un resumen con el tiempo que ha necesitado para completar el nivel y los fallos cometidos. En caso de que el tiempo o el número de fallos sean menores a los anteriormente registrados, también se mostrará “(Récord)”, en rojo.

Por último, en la parte inferior central hay un botón que permite volver al menú principal.

6.6.8 Rosco



Figura 43. Rosco, versión final

Rosco es un juego en el que el usuario tiene que responder preguntas, cuyas respuestas empiezan por letras específicas, siguiendo el orden del abecedario.

En la parte superior de la pantalla se encuentra la barra de navegación, con el botón de retroceso, el nombre e icono del juego y el progreso, en este caso, representando el número de letras acertadas del total.

Debajo se muestra la letra actual y el listado total de letras. Estas estarán coloreadas en verde si se ha respondido correctamente, o en rojo, en caso de que la respuesta haya sido incorrecta.

En la mitad se encuentra la pregunta que el usuario debe leer y responder en la caja de texto que está justo debajo. Una vez escrita la respuesta, el usuario debe pulsar el botón "Comprobar" para verificarla.

Adicionalmente, el usuario puede avanzar o retroceder entre letras usando las flechas laterales en la parte inferior.

Cabe mencionar que este juego tiene una única dificultad porque esta es inherente a las preguntas y no tendría sentido modificarla de manera externa.

El nivel se completa cuando se responden correctamente todas las preguntas. Cabe mencionar que una pregunta incorrecta se puede volver a responder.

6.6.9 Puzzle

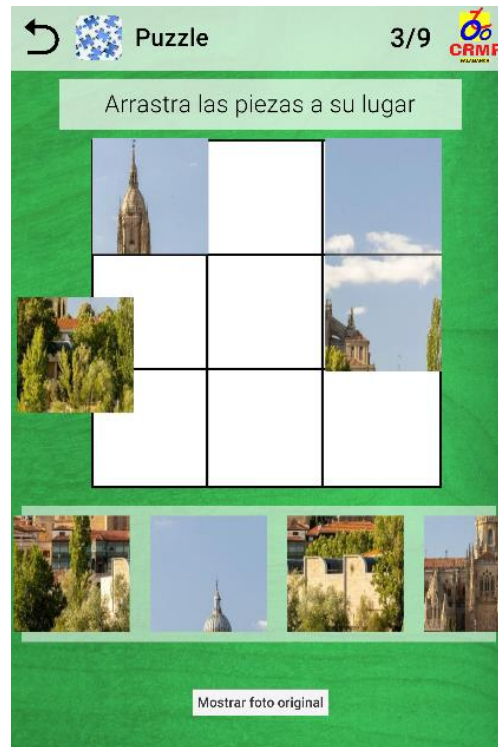


Figura 44. Puzzle, versión final

Puzzle es un juego en el que el usuario debe arrastrar las piezas a su posición original.

En primer lugar, se presenta la imagen en el centro de la pantalla, para que el usuario intente recordarla y cuando esté listo, pulse el botón “Desordenar”, que comienza el juego.

Posteriormente, en el centro quedará una cuadrícula blanca que el usuario puede tomar como referencia para colocar las piezas que se encuentran en la mitad inferior de la pantalla, en un rectángulo que se puede deslizar para acceder al resto de piezas.

Cuando una pieza se coloque a una posición muy cercana a la original, esta se fijará y no se podrá mover más, si no, la pieza se quedará en el punto en el que sea colocada y podrá ser movida a otra posición o junto al resto de piezas de la zona inferior.

Asimismo, en la parte inferior central se encuentra un botón “Mostrar foto original”, que colocará la imagen con transparencia en la cuadrícula hasta que el usuario suelte dicho botón.

La dificultad cambia el número y tamaño de las piezas, siendo 9 piezas en fácil, 36 en normal y 100 en difícil.

El nivel se completa cuando se colocan correctamente todas las piezas.

6.6.10 Juego Nombres



Figura 45. Nombres, versión final

Nombres es un juego que consiste en memorizar el nombre de cada imagen y emparejarlos después.

En primer lugar, se presentan todas las imágenes junto a su nombre y cuando el usuario pulse el botón “Empezar” comenzará el juego.

En la mitad superior de la pantalla se encuentran las imágenes y en la inferior los nombres. Cuando el usuario pulse en uno de estos, se enmarcará con un borde dorado y si después pulsa su correspondiente nombre o imagen, se volverán transparentes y ya no se podrán volver a pulsar. En caso de que sea incorrecto, el texto se volverá rojo durante un par de segundos.

La dificultad afecta al número de nombres que hay que emparejar, siendo 6 para fácil, 9 para normal y 20 para difícil.

El nivel se completa cuando se emparejan todas las imágenes con sus nombres.

6.6.11 Juego Cartas



Figura 46. Cartas, versión final

Cartas es un juego que consiste en emparejar imágenes iguales.

En primer lugar, se muestran todas las cartas bocarriba para que el usuario intente recordar la posición de cada pareja hasta que pulse el botón “Voltear”, cuando empezará el juego.

Tras esto, todas las cartas estarán dadas la vuelta y cuando el usuario pulse en una, esta se mostrará hasta que pulse la siguiente. Si resultan ser iguales, se volverán transparentes y ya no se podrán volver a pulsar. En caso contrario, se mostrarán ambas durante un segundo y se volverán a dar la vuelta.

La dificultad afecta al número de cartas, siendo 8 cartas en fácil, 16 en normal y 30 en difícil.

El nivel se completa cuando todas las parejas hayan sido encontradas.

7 CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS

Al final del proyecto se ha conseguido desarrollar una aplicación correctamente adaptada a los usuarios del centro y configurable por los encargados, mediante la división en dos programas, servidor y aplicación.

Se han conseguido cumplir todos los requisitos y objetivos propuestos en un principio y los añadidos a lo largo de las iteraciones de implementación y diseño.

En cuanto a lo referido a la aplicación de conocimientos adquiridos a lo largo de la carrera, puedo destacar las siguientes asignaturas:

Ingeniería del Software: Por lo que respecta a las fases de análisis y diseño en las que especifican objetivos, requisitos y casos de uso detallados al inicio del proyecto.

Mi opinión personal con respecto a esto es que este procedimiento es algo altamente necesario para proyectos de gran envergadura. Sin embargo, al ser este proyecto uno de menor categoría, los procesos de ingeniería del software acaban ocupando la mayor parte del tiempo dedicado al proyecto, dejando mucho menos tiempo para el desarrollo.

Aún con esta desventaja, está claro que este proceso será de gran utilidad en caso de que haya que mantener o ampliar el proyecto en un futuro.

Animación Digital: Por lo relacionado con el funcionamiento y la programación en Unity.

Tras haber realizado el proyecto, sigo pensando que Unity es una gran herramienta para desarrollar aplicaciones por su facilidad de uso y su potencia.

Interacción-Persona-Ordenador: Por lo relacionado a crear una interfaz fácil de entender para el usuario.

En un principio no me di cuenta de la importancia que tenía este apartado del desarrollo, pero a lo largo de la implementación he visto que es muy fácil hacer una mala interfaz que no entienda nadie, salvo el desarrollador. Por ello he aplicado más en serio los conocimientos de la asignatura y mediante pruebas, inspiración en otras aplicaciones y herramientas como el Contrast Finder, he conseguido obtener un modelo de interfaz más adecuado.

En un futuro se podrían realizar una serie de mejoras para la aplicación, como:

- Añadir más juegos. Unos de los que se pensaron al principio, pero que no se llegaron a implementar fueron: Sopa de letras, para trabajar la atención y la semántica; ordenación de oraciones, para trabajar la semántica y asociación de trabajadores con lugares del centro, para trabajar la memoria.
- Opciones de accesibilidad, tales como un filtro de colores, distintos tamaños de fuente o compatibilidad con algunas máquinas del CRMF, como pueden ser el rastreador ocular.
- Hacer una tabla de récords general, en la que los usuarios puedan compararse con el resto para competir e incitar el progreso y la mejora de sus capacidades cerebrales.
- Compatibilidad con otros sistemas operativos, como IOS.

- Creación de interfaz responsiva para adaptarse a dispositivos móviles con un mayor rango de distintas resoluciones.
- Mejora de la arquitectura cliente-servidor para que los usuarios puedan utilizarlo desde casa y no necesariamente desde la misma red local.

Aún con todas estas mejoras que podrían realizarse, mi conclusión final es que estoy satisfecho con el resultado de la aplicación y que he aprendido mucho sobre los procedimientos ajenos a la programación que conlleva el desarrollo de un programa.

REFERENCIAS Y BIBLIOGRAFÍA

Se ha utilizado el estilo de citación Vancouver.

1. Jiménez-Echenique J. La terapia ocupacional, fundamentos de la disciplina. Cuad Hosp Clínicas. 2018;59(ESPECIAL):82–7.
2. Carrillo-Mora P. Sistemas de memoria: reseña histórica, clasificación y conceptos actuales. Primera parte: Historia, taxonomía de la memoria, sistemas de memoria de largo plazo: la memoria semántica. Salud Ment. 2010 Feb;33(1):85–93.
3. Jigsaw Puzzles - puzzle games - Apps on Google Play [Internet]. [cited 2023 Sep 3]. Available from: <https://play.google.com/store/apps/details?id=com.easybrain.jigsaw.puzzles&hl=en>
4. Pasapalabra (España). In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: [https://es.wikipedia.org/w/index.php?title=Pasapalabra_\(Espa%C3%B1a\)&oldid=153463157](https://es.wikipedia.org/w/index.php?title=Pasapalabra_(Espa%C3%B1a)&oldid=153463157)
5. Desarrollo ágil de software. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 4]. Available from: https://es.wikipedia.org/w/index.php?title=Desarrollo_%C3%A1gil_de_software&oldid=153436524
6. Scrum (desarrollo de software). In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 4]. Available from: [https://es.wikipedia.org/w/index.php?title=Scrum_\(desarrollo_de_software\)&oldid=152581461](https://es.wikipedia.org/w/index.php?title=Scrum_(desarrollo_de_software)&oldid=152581461)
7. Unity (motor de videojuego). In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: [https://es.wikipedia.org/w/index.php?title=Unity_\(motor_de_videojuego\)&oldid=151999035](https://es.wikipedia.org/w/index.php?title=Unity_(motor_de_videojuego)&oldid=151999035)
8. Microsoft Visual Studio. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: https://es.wikipedia.org/w/index.php?title=Microsoft_Visual_Studio&oldid=153493795
9. Microsoft Word. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: https://es.wikipedia.org/w/index.php?title=Microsoft_Word&oldid=153369879
10. ChatGPT. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: <https://es.wikipedia.org/w/index.php?title=ChatGPT&oldid=153492207>
11. Adobe Photoshop. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: https://es.wikipedia.org/w/index.php?title=Adobe_Photoshop&oldid=152974480

12. Golden [Internet]. [cited 2023 Sep 3]. Dall-E 2 - Wiki. Available from: https://golden.com/wiki/Dall-E_2-EKMN8GY
13. Microsoft Paint. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 3]. Available from: https://es.wikipedia.org/w/index.php?title=Microsoft_Paint&oldid=153367563
14. Contrast-finder.org. Contrast-Finder. [cited 2023 Sep 3]. Contrast Finder, encuentra los contrastes adecuados de color para crear webs accesibles (WCAG). Available from: <https://app.contrast-finder.org/?lang=es>
15. Microsoft Project. In: Wikipedia, la enciclopedia libre [Internet]. 2023 [cited 2023 Sep 5]. Available from: https://es.wikipedia.org/w/index.php?title=Microsoft_Project&oldid=152413673
16. Visual Paradigm - Online Productivity Suite [Internet]. [cited 2023 Sep 5]. Available from: <https://online.visual-paradigm.com/>
17. Doxygen: Doxygen [Internet]. [cited 2023 Sep 5]. Available from: <https://www.doxygen.nl/>
18. Scroll View in Unity - 2021 Tutorial - YouTube [Internet]. [cited 2023 Sep 3]. Available from: <https://www.youtube.com/watch?v=Q-G-W93jhYc>
19. Unity Forum [Internet]. [cited 2023 Sep 3]. How Can I Open Previous Scene? Available from: <https://forum.unity.com/threads/how-can-i-open-previous-scene.652507/>
20. How to use a Custom Font with Text Mesh Pro in Unity [Internet]. 2021 [cited 2023 Sep 3]. Available from: <https://www.youtube.com/watch?v=A33HoKdOhTQ>
21. Do's and Don'ts of Using Light Typefaces [Internet]. 2014 [cited 2023 Sep 3]. Available from: <https://uxmovement.com/content/dos-and-donts-of-using-light-typefaces/>