



ESCUELA SUPERIOR DE INGENIERÍA

INGENIERO TÉCNICO INFORMÁTICA DE SISTEMAS

**DIDACTICGAME: PLATAFORMA DIDÁCTICA PARA
ANDROID Y ESCRITORIO.**

Israel Alías Perdigones

Cádiz, Mayo 2016



ESCUELA SUPERIOR DE INGENIERÍA

INGENIERO TÉCNICO EN INFORMÁTICA DE SISTEMAS

DIDACTICGAME: PLATAFORMA DIDÁCTICA PARA ANDROID Y ESCRITORIO

DEPARTAMENTO: INGENIERÍA INFORMÁTICA

DIRECTOR DEL PROYECTO:..... GUADALUPE ORTIZ BELLOT

AUTOR DEL PROYECTO:..... ISRAEL ALÍAS PERDIGONES

Cádiz, Mayo 2016

Fdo.: Israel Alías Perdigones

Índice general

1. INTRODUCCIÓN	13
1.1. MOTIVACIÓN	14
1.1.1. <i>Motivación docente</i>	14
1.1.2. <i>Motivación personal</i>	15
1.2. OBJETIVOS	16
1.3. ALCANCE	16
2. DESCRIPCIÓN GENERAL DEL PROYECTO.....	17
2.1. PERSPECTIVA DEL PRODUCTO	17
2.1.1. <i>Dependencias del producto</i>	17
2.1.2. <i>Interfaces de usuario</i>	17
2.1.3. <i>Interfaces software</i>	18
2.2. FUNCIONES DEL PRODUCTO	19
2.3. CARACTERÍSTICAS DE LOS USUARIOS	19
3. PLANIFICACIÓN	19
3.1. FASES	19
3.1.1. <i>Primera fase (toma de contacto con las tecnologías)</i>	20
3.1.2. <i>Segunda fase (primeras pruebas)</i>	20
3.1.3. <i>Tercera fase (elección del proyecto final)</i>	20
3.1.4. <i>Cuarta fase (planteamiento de la interfaz gráfica)</i>	20
3.1.5. <i>Quinta fase (administración de la información)</i>	21
3.1.6. <i>Sexta fase (obtención de los ejecutables)</i>	21
3.1.7. <i>Séptima fase (realización de pruebas)</i>	22
3.1.8. <i>Octava fase (realización de la memoria)</i>	22
3.1.9. <i>Novena fase (desarrollo de la presentación)</i>	22
3.2. DIAGRAMA DE GANTT	22
4. ANÁLISIS.....	23
4.1. TOMA DE REQUISITOS	23
4.1.1. <i>Requisitos funcionales del sistema</i>	23
4.1.2. <i>Requisitos no funcionales del sistema</i>	24
4.2. MODELO DE INTERACCIÓN DEL SISTEMA	24
4.3. MODELO DE CASOS DE USO	25
4.3.1. <i>Modelo general de casos de uso</i>	25

4.4.	MODELO DE COMPORTAMIENTO DEL SISTEMA	34
4.4.1.	Caso de uso: Jugar Rosco	34
4.4.2.	Caso de uso: Jugar QQSM	35
4.4.3.	Caso de uso: Tiempo Finalizado	36
4.4.4.	Caso de uso: Usar Comodín.....	37
4.4.5.	Caso de uso: Elegir Respuesta	37
4.4.6.	Caso de uso: Ver Ranking.....	38
4.4.7.	Caso de uso: Ver Puntuaciones.....	40
4.4.8.	Caso de uso: Añadir Jugador.....	40
4.4.9.	Caso de uso: Eliminar Jugador.....	41
4.4.10.	Caso de uso: Modificar Edad.....	42
4.4.11.	Caso de uso: Modificar Color.....	43
4.4.12.	Caso de uso: Eliminar Palabra.....	43
4.4.13.	Caso de uso: Añadir Palabra	44
4.4.14.	Caso de uso: Añadir Palabras.....	44
4.4.15.	Caso de uso: Seleccionar Letra	45
4.4.16.	Caso de uso: Editar Descripción.....	46
4.4.17.	Caso de uso: Eliminar Pregunta.....	47
4.4.18.	Caso de uso: Añadir Pregunta	47
4.4.19.	Caso de uso: Añadir Preguntas.....	48
4.4.20.	Caso de uso: Seleccionar Correcta	49
4.4.21.	Caso de uso: Editar Respuesta.....	49
4.4.22.	Caso de uso: Salir.....	50
4.5.	MODELO CONCEPTUAL DE DATOS	51
5.	DISEÑO	51
5.1.	INTERFAZ.....	51
5.1.1.	Adobe Photoshop CC	51
5.1.2.	Vector Magic.....	51
5.2.	MENÚS.....	52
5.3.	BASE DE DATOS	52
6.	IMPLEMENTACIÓN	53
6.1.	TECNOLOGÍAS.....	54
6.2.	CÓDIGO	55
7.	PRUEBAS	59
7.1.	PRUEBAS UNITARIAS	60

7.2.	PRUEBAS DE INTEGRACIÓN	60
7.2.1.	Menú Inicial:.....	60
7.2.2.	Menú Datos:.....	60
7.2.3.	Menú Jugadores:	61
7.2.4.	Menú Puntuaciones:.....	61
7.2.5.	Menú Ranking:	61
7.2.6.	Menú Jugar:	61
7.2.7.	Juego Rosco:.....	61
7.2.8.	Juego QQSM:.....	62
7.2.9.	Clasificación:	62
7.3.	PRUEBAS FUNCIONALES	62
7.4.	PRUEBAS NO FUNCIONALES	62
7.5.	PRUEBAS DE ACEPTACIÓN	63
8.	CONCLUSIONES	63
8.1.	SÍNTESIS.....	63
8.2.	TRABAJO FUTURO	63
8.3.	OBJETIVOS ALCANZADOS	65
8.4.	EXPERIENCIA PERSONAL.....	66
9.	BIBLIOGRAFÍA.....	66
ANEXOS	67	
ANEXO 1.	MANUAL DEL USUARIO	67
A1.1.	Requisitos del sistema.....	67
A1.2.	Instalación.....	67
A1.3.	Juego.....	67
A1.3.1.	Controles	67
A1.3.2.	Menús.....	68
A1.3.3.	Introducción de datos.....	76
ANEXO 2.	MANUAL DEL DESARROLLADOR	79
A2.1.	Herramientas utilizadas	79
A2.2.	Conexión con la base de datos	79
A2.3.	Manejo de la base de datos	79
A2.4.	Clases y objetos comunes.....	80
A2.5.	Creación de textos	82
A2.6.	Redimensión de imágenes.....	83

Índice de figuras

FIGURA 1: DIAGRAMA DE GANTT	23
FIGURA 2: DIAGRAMA INTERACCIÓN DEL SISTEMA	25
FIGURA 3: DIAGRAMA GENERAL DE CASOS DE USO.....	26
FIGURA 4: DIAGRAMA DE SECUENCIA (JUGAR ROSCO).....	35
FIGURA 5: DIAGRAMA DE SECUENCIA (JUGAR QQSM)	35
FIGURA 6: DIAGRAMA DE SECUENCIA (INICIO ROSCO)	36
FIGURA 7: DIAGRAMA DE SECUENCIA (USAR COMODÍN)	37
FIGURA 8: DIAGRAMA DE SECUENCIA (ELEGIR RESPUESTA)	37
FIGURA 9: DIAGRAMA DE SECUENCIA (VER RANKING)	39
FIGURA 10: DIAGRAMA DE SECUENCIA (VER PUNTUACIONES)	40
FIGURA 11: DIAGRAMA DE SECUENCIA (AÑADIR JUGADOR)	41
FIGURA 12: DIAGRAMA DE SECUENCIA (ELIMINAR JUGADOR).....	41
FIGURA 13: DIAGRAMA DE SECUENCIA (MODIFICAR EDAD)	42
FIGURA 14: DIAGRAMA DE SECUENCIA (MODIFICAR COLOR).....	43
FIGURA 15: DIAGRAMA DE SECUENCIA (ELIMINAR PALABRA)	43
FIGURA 16: DIAGRAMA DE SECUENCIA (AÑADIR PALABRA).....	44
FIGURA 17: DIAGRAMA DE SECUENCIA (AÑADIR PALABRAS)	45
FIGURA 18: DIAGRAMA DE SECUENCIA (SELECCIONAR LETRA)	45
FIGURA 19: DIAGRAMA DE SECUENCIA (EDITAR DESCRIPCIÓN)	46
FIGURA 20: DIAGRAMA DE SECUENCIA (ELIMINAR PREGUNTA).....	47
FIGURA 21: DIAGRAMA DE SECUENCIA (AÑADIR PREGUNTA)	47
FIGURA 22: DIAGRAMA DE SECUENCIA (AÑADIR PREGUNTAS)	48
FIGURA 23: DIAGRAMA DE SECUENCIA (SELECCIONAR CORRECTA).....	49
FIGURA 24: DIAGRAMA DE SECUENCIA (EDITAR RESPUESTA).....	49
FIGURA 25: DIAGRAMA DE SECUENCIA (SALIR)	50
FIGURA 26: DIAGRAMA CONCEPTUAL DE CLASES	51
FIGURA 27: BASES DE DATOS	53
FIGURA 28: MENÚ INICIAL	68
FIGURA 29: MENÚ DATOS	69
FIGURA 30: MENÚ DATOS (INSERTAR FICHERO)	70
FIGURA 31: MENÚ DATOS (INSERTAR PALABRA)	70
FIGURA 32: MENÚ JUGADORES	71
FIGURA 33: MENÚ JUGADORES (SELECCIONAR COLOR)	71
FIGURA 34: MENÚ PUNTUACIONES.....	72

FIGURA 35: MENÚ RANKING	72
FIGURA 36: MENÚ JUGAR.....	73
FIGURA 37: JUEGO ROSCO (ESTADO INICIAL)	73
FIGURA 38: JUEGO ROSCO (JUGANDO)	74
FIGURA 39: JUEGO QQSM (ESTADO INICIAL)	75
FIGURA 40: JUEGO QQSM (JUGANDO)	76
FIGURA 41: CLASIFICACIÓN	76
FIGURA 42: MENÚ DATOS (DESCRIPCIÓN)	78
FIGURA 43: DIAGRAMA DE CLASES	81

Índice de tablas

TABLA 1: CASO DE USO (JUGAR <i>ROSCO</i>)	26
TABLA 2: CASO DE USO (JUGAR <i>QQSM</i>)	27
TABLA 3: CASO DE USO (VER RANKING).....	27
TABLA 4: CASO DE USO (VER PUNTUACIONES)	27
TABLA 5: CASO DE USO (ADMINISTRAR JUGADORES).....	27
TABLA 6: CASO DE USO (SALIR)	28
TABLA 7: CASO DE USO (ADMINISTRAR DATOS)	28
TABLA 8: CASO DE USO (FALLAR)	29
TABLA 9: CASO DE USO (TIEMPO FINALIZADO)	29
TABLA 10: CASO DE USO (USAR COMODÍN).....	29
TABLA 11: CASO DE USO (ELEGIR RESPUESTA).....	30
TABLA 12: CASO DE USO (AÑADIR JUGADOR)	30
TABLA 13: CASO DE USO (ELIMINAR JUGADOR)	30
TABLA 14: CASO DE USO (MODIFICAR EDAD).....	30
TABLA 15: CASO DE USO (MODIFICAR COLOR)	31
TABLA 16: CASO DE USO (ELIMINAR PALABRA).....	31
TABLA 17: CASO DE USO (AÑADIR PALABRA)	31
TABLA 18: CASO DE USO (AÑADIR PALABRAS).....	32
TABLA 19: CASO DE USO (SELECCIONAR LETRA).....	32
TABLA 20: CASO DE USO (EDITAR DESCRIPCIÓN)	32
TABLA 21: CASO DE USO (ELIMINAR PREGUNTA)	33
TABLA 22: CASO DE USO (AÑADIR PREGUNTA).....	33
TABLA 23: CASO DE USO (AÑADIR PREGUNTAS)	33
TABLA 24: CASO DE USO (SELECCIONAR CORRECTA)	34
TABLA 25: CASO DE USO (EDITAR RESPUESTA)	34
TABLA 26: DIAGRAMA DE SECUENCIA (JUGAR <i>ROSCO</i>).....	35
TABLA 27: DIAGRAMA DE SECUENCIA (JUGAR <i>QQSM</i>).....	36
TABLA 28: DIAGRAMA DE SECUENCIA (TIEMPO FINALIZADO)	36
TABLA 29: DIAGRAMA DE SECUENCIA (USAR COMODÍN).....	37
TABLA 30: DIAGRAMA DE SECUENCIA (ELEGIR RESPUESTA).....	38
TABLA 31: DIAGRAMA DE SECUENCIA (VER RANKING).....	40
TABLA 32: DIAGRAMA DE SECUENCIA (VER PUNTUACIONES)	40
TABLA 33: DIAGRAMA DE SECUENCIA (AÑADIR JUGADOR)	41
TABLA 34: DIAGRAMA DE SECUENCIA (ELIMINAR JUGADOR)	42

TABLA 35: DIAGRAMA DE SECUENCIA (MODIFICAR EDAD).....	42
TABLA 36: DIAGRAMA DE SECUENCIA (MODIFICAR COLOR)	43
TABLA 37: DIAGRAMA DE SECUENCIA (ELIMINAR PALABRA).....	44
TABLA 38: DIAGRAMA DE SECUENCIA (AÑADIR PALABRA)	44
TABLA 39: DIAGRAMA DE SECUENCIA (AÑADIR PALABRAS).....	45
TABLA 40: DIAGRAMA DE SECUENCIA (SELECCIONAR LETRA)	46
TABLA 41: DIAGRAMA DE SECUENCIA (EDITAR DESCRIPCIÓN).....	46
TABLA 42: DIAGRAMA DE SECUENCIA (ELIMINAR PREGUNTA)	47
TABLA 43: DIAGRAMA DE SECUENCIA (AÑADIR PREGUNTA).....	48
TABLA 44: DIAGRAMA DE SECUENCIA (AÑADIR PREGUNTAS)	48
TABLA 45: DIAGRAMA DE SECUENCIA (SELECCIONAR CORRECTA)	49
TABLA 46: DIAGRAMA DE SECUENCIA (EDITAR RESPUESTA)	50
TABLA 47: DIAGRAMA DE SECUENCIA (SALIR)	50

1. Introducción

En los últimos años, diversos estudios han demostrado que las actividades lúdicas mejoran el aprendizaje de los estudiantes. Por ello, es necesario desarrollar dichas actividades con la finalidad de potenciar el conocimiento de las diferentes áreas de estudio.

Actividades como las propuestas en este proyecto son un ejemplo de que la tecnología sigue estando al servicio de aquel que la precisa solucionando nuestros problemas del día a día. En el presente proyecto se aborda este objetivo desarrollando un cliente Android y otro para escritorio que nos permite, como ya se ha comentado, de una forma lúdica adquirir conocimientos de cualquier materia mediante el aprendizaje de los principales conceptos que nos ocupen a través de un juego. Esto se lleva a cabo mediante la simulación de varios concursos televisivos, como por ejemplo “Pasapalabra” o “¿Quién quiere ser millonario?”, con los que el usuario puede centrarse principalmente en el aprendizaje de los conceptos y no tanto en la mecánica del juego.

A través de estos juegos y mediante las plataformas para los que están disponibles, se puede llevar a cabo este tipo de aprendizaje, de forma individual (cliente Android) o de forma grupal (con un ordenador personal o mediante su proyección en un aula)

Una de las principales características del proyecto es que permite a sus usuarios elegir el ámbito de estudio, al poder introducir cualquier batería de conceptos y definiciones que sean oportunos.

Además de esto, futuras ampliaciones, como la introducción de otros juegos, la posibilidad de elegir los conceptos según la temática a estudiar en cada momento o el uso de la aplicación en otros entornos de ejecución, hacen de este proyecto el comienzo de una herramienta con la que muchos puedan hacer del estudio un hobby.

1.1. Motivación

1.1.1. Motivación docente

La educación se caracteriza por la interacción de múltiples procesos de aprendizaje puesto que cada persona posee una forma diferente de pensar y aprender. Es por ello que algunos investigadores afirman que no existe un método de aprendizaje puro, al igual que no existen estilos homogéneos de personalidad.

Así, las estrategias didácticas de los docentes deberían tener en cuenta los estilos de aprendizaje de los estudiantes, para que potenciaran más sus habilidades cognitivas logrando un aprendizaje significativo y útil.

Diversos autores¹ establecen cuatro tipos de aprendizaje, entre los que se encuentra el estilo activo. Este estilo se caracteriza por la implicación plena del alumno en nuevas tareas que signifiquen retos en la realización y consecución de objetivos. Las actividades cuyo proceso dura mucho tiempo terminan por cansar al alumno que ve mermada su capacidad de aprendizaje.

Un ejemplo de actividades que favorecen y potencian el aprendizaje para este grupo son las siguientes:

- Coordinación del trabajo en pequeños grupos, evitando actividades en grupos grandes
- Participación en clases en las que primen las actividades prácticas
- Asistencia a clases con un enfoque lúdico
- Empleo de las TIC (vídeo, audio, fotografía, internet, aplicaciones informáticas, etc.) para la realización de tareas
- Elaboración de mapas conceptuales con palabras clave

¹ Libro: Alonso, C.; Gallego, D.; Honey, P. (1994). Los Estilos de Aprendizaje. Procedimientos de diagnóstico y Mejora.

- Puesta en práctica de lo aprendido en clase
- Interacción profesor-alumno en clases dinámicas

Herramientas como la propuesta en este proyecto, intentan satisfacer las necesidades de personas que presentan un estilo activo de aprendizaje y apoyar así, la labor docente de una forma más eficaz.

1.1.2. Motivación personal

No quería comenzar el desarrollo de este documento sin explicar a groso modo la situación en la que me encuentro en el momento de desarrollar este proyecto.

Todo se remonta al año 2009 en el que finalizo la educación secundaria obligatoria y me dispongo a la elección del que probablemente sea mi ámbito laboral por el resto de mi vida, cosa que nunca es sencilla ni definitiva.

Mis influencias familiares me llevan a la elección de las fuerzas armadas como futuro laboral, y la escala de oficiales como meta. Por lo tanto las opciones que se me plantean son la realización de una academia de cinco años o la realización de una diplomatura y la consiguiente academia de dos años. Esta última opción es la elegida y por lo tanto solo restaba la elección de la diplomatura a realizar. Una vez más las influencias familiares me llevan a la elección de la ingeniería técnica informática.

Después de todos los cursos académicos, unos con más fortuna que otros, me llevan a la obtención de todos los créditos necesarios para poder plantearme el siguiente paso a seguir, y no, no era la realización de este proyecto, sino el ingreso en la academia de infantería de marina.

Al término del periodo de formación y la posterior incorporación al destino se me plantea la necesidad de la realización de este proyecto para la posterior realización del concurso/oposición para el acceso a la escala de oficiales del ejército.

Y este es el punto en el que nos encontramos, en la obtención del último requisito para alcanzar la mayor meta en mi vida hasta ahora.

1.2. Objetivos

El principal objetivo de este proyecto es el empleo del apoyo informático para el aprendizaje, a nivel personal o a nivel grupal, de los conceptos más genéricos o más específicos de cualquier ámbito en el que se emplee.

Los objetivos específicos a alcanzar en la realización del proyecto son:

- Creación de una aplicación multiplataforma orientada al aprendizaje de cualquier temática. Dicha aplicación constará de las siguientes características generales:
 - Elección de cualquier temática de estudio mediante la introducción libre de la batería de conceptos y preguntas.
 - Carga automática de las preguntas y respuestas especificadas en un formato concreto.
 - Personalización de cada sesión según la funcionalidad que se le quiera dar a la aplicación (un único jugador, varios, con/sin temporizador, etc.)
 - Administración del perfil de cada usuario.
 - Seguimiento del progreso en el aprendizaje mediante las puntuaciones almacenadas de cada sesión.
- La aplicación estará disponible tanto para un entorno Android como para un entorno de escritorio.

1.3. Alcance

El uso de esta aplicación está orientado al aprendizaje de los distintos conceptos que sean precisos en cualquier ámbito de estudio, es por ello que el usuario no ha de ser encasillado en edad o tema de estudio y de ahí su versatilidad en lo referente al qué y al cómo se estudia dicha terminología.

Es por ello que se precisa del desarrollo de dos clientes, para dispositivos móviles y de escritorio con los cuales se puede abarcar cualquier edad y estilo de vida.

En primer lugar se desarrollará un cliente para dispositivos móviles que permita a cualquier usuario con un dispositivo Android el uso de este soporte para su aprendizaje personal. En segundo lugar un cliente para escritorio que permita su uso tanto a nivel personal como grupal y su consiguiente beneficio al interaccionar con otros usuarios.

2. Descripción general del proyecto

A continuación se realiza una breve descripción de los rasgos principales del producto.

2.1. Perspectiva del producto

2.1.1. Dependencias del producto

Este proyecto se crea como una herramienta de aprendizaje independiente, no forma parte de un sistema ya creado ni pretende serlo, sin embargo, la condición de software libre lo hace idóneo para ser continuado por quien lo estime oportuno. De hecho, una de las características del proyecto es su capacidad de ser ampliado, añadiendo nuevos juegos y características que más adelante se detallan.

2.1.2. Interfaces de usuario

La interfaz gráfica se plantea mediante menús, a los que se puede acceder mediante la pulsación de los botones que aparecen en ellos o

mediante los clics de ratón si nos encontramos en el cliente de escritorio o mediante la utilización de los botones del sistema si nos encontramos en Android.

Estos menús a su vez, nos llevarán a los juegos implementados en el sistema, y estos, a las pantallas de puntuaciones correspondientes.

Cabe destacar el menú de introducción de datos, mediante el cual podemos modificar la base de datos [11] a través de la interfaz gráfica de forma muy intuitiva.

Por otro lado, las pantallas correspondientes a los juegos implementados, buscan, a través de la ambientación de los programas televisivos que simulan, que la experiencia del usuario sea amena e intuitiva, sin tener la necesidad de mirar siquiera el manual.

Una de las características principales de la interfaz gráfica es que carece de ninguna palabra en menús o botones más allá de los nombres de los juegos ya implementados. Esto permite que cualquiera sea el idioma del usuario, pueda hacer uso de la aplicación sin que su experiencia se vea perjudicada.

Así mismo, la interfaz es capaz de adaptarse a cualquier resolución de pantalla, siendo 1024x576 la resolución óptima para su uso.

2.1.3. Interfaces software

El producto se relaciona con el sistema operativo en el que se ejecuta, de tal forma que adapta su funcionamiento a él. Los controles son adaptados al entorno gracias al framework LibGDX [6] que facilita esta labor, añadiendo al producto la característica de ser multiplataforma.

Las librerías empleadas en el proyecto garantizan sus continuas actualizaciones, lo que facilita el mantenimiento e incluso la mejora del mismo.

Así mismo, el sistema interactúa con la base de datos SQLite [12] mediante la conexión con esta a través de un driver específico.

2.2. Funciones del producto

La función principal del producto es la de apoyar y potenciar el aprendizaje en cualquier ámbito de estudio. Para ello es necesario que el programa sea intuitivo, capaz de adaptarse a las necesidades del usuario y permitir la interacción de varios de ellos mejorando así la experiencia de uso. A grandes rasgos las principales funciones que debe realizar el producto son las siguientes:

- Almacenamiento del perfil de cada jugador.
- Almacenamiento de la batería de preguntas y palabras a usar en cada juego.
- Capacidad de modificar los datos almacenados para el aprendizaje.
- Posibilidad de jugar a varios juegos que enriquezcan la experiencia de aprendizaje.
- Generar una retroalimentación en forma de puntuaciones para cada partida.
- Generar listas con estas puntuaciones que puedan ser comparadas entre jugadores.

2.3. Características de los usuarios

La aplicación creada presenta una interfaz intuitiva y fácil de usar, lo que permite a cualquier tipo de usuario, independientemente de los conocimientos que posea, el manejo de dicha aplicación sin dificultad. Esto es esencial si lo que se persigue es fomentar el aprendizaje al no verse limitado por el uso de la aplicación.

3. Planificación

3.1. Fases

A continuación se presentan los distintos periodos de desarrollo que se han llevado a cabo para la realización del proyecto

3.1.1. Primera fase (toma de contacto con las tecnologías)

Mis primeros pasos en el desarrollo del proyecto se dieron cuando contrastaba información acerca de las diferentes posibles elecciones de herramientas para el desarrollo de un videojuego, ya que en la carrera no pude cursar la asignatura destinada a dicho fin y la curiosidad me llevaba a aprender sobre el tema.

Fue entonces cuando un compañero me comenta que su proyecto se basa precisamente en el aprendizaje de LibGDX como framework para el desarrollo de videojuegos multiplataforma y es cuando empiezo a investigar sobre el tema.

3.1.2. Segunda fase (primeras pruebas)

Es entonces cuando realizo varias pruebas que consisten en la reproducción de varios juegos clásicos y otros no tanto, con los que puedo empezar a familiarizarme con el entorno y puedo empezar a integrar diferentes herramientas que enriquecen la experiencia. Juegos como Súper Mario Bros o 1942 sirven como inspiración para entender el reto al que me enfrente sin más ayuda que un buscador y muchas dudas.

3.1.3. Tercera fase (elección del proyecto final)

Una vez que confirme que LibGDX era el framework que quería usar y que por lo tanto Java [5] era el lenguaje a emplear, el siguiente paso era la búsqueda del proyecto en concreto a desarrollar.

Después de unas semanas de correos con varios posibles tutores, escogí el proyecto en cuestión por su gran versatilidad y su enfoque en el aprendizaje didáctico.

3.1.4. Cuarta fase (planteamiento de la interfaz gráfica)

El proyecto se basa en el uso de una interfaz gráfica intuitiva para el aprendizaje mediante juegos de cualquier temática, por lo tanto mis primeros pasos fueron referentes al desarrollo de esta interfaz a nivel gráfico y por ultimo implementación.

Lo primero fue el desarrollo de todas las pantallas de menús de administración de jugadores y conceptos que luego se mostrarían en los juegos en cuestión.

Posteriormente se desarrolló el primer juego ambientado en el programa de televisión "Pasapalabra", en el que el jugador debe encontrar la palabra que corresponde a la descripción que se le plantea.

Más tarde se desarrolló una simulación del programa de televisión "¿Quién quiere ser millonario?" en el que el usuario debe contestar una serie de preguntas en las que dispone de cuatro posibles respuestas, siendo una de ellas la correcta.

3.1.5. Quinta fase (administración de la información)

Luego de realizar el diseño y desarrollo de la interfaz gráfica, se plantea la necesidad de elegir la correcta organización y almacenamiento de la información a emplear. Al ser la información a almacenar de no muy gran tamaño, se decide el uso de SQLite como herramienta de administración de la base de datos, lo que nos permite un menor coste en las búsquedas así como en el almacenamiento de la información. La información se almacenara en una serie de tablas con las que la información puede ser fácilmente consultada y modificada.

3.1.6. Sexta fase (obtención de los ejecutables)

Para poder emplear este proyecto como una herramienta de aprendizaje es necesario disponer de dos ejecutables, el concerniente a Android y al escritorio, para poder ser empleados en los distintos escenarios que se presenten según el ámbito de aprendizaje.

3.1.7. Séptima fase (realización de pruebas)

Para comprobar el correcto funcionamiento del programa, se han de realizar diferentes pruebas que abarquen todos los posibles puntos en los que el programa pueda generar un mal funcionamiento. Considerando que cada módulo añadido ha sido probado por separado antes de su integración, solo restará probar el programa al completo como suma de sus partes. Se ha de tener en cuenta que el programa se enfoca al uso educativo y por lo tanto el control de la información a mostrar dependerá exclusivamente del usuario final. Es interesante considerar que la comprobación de su correcto funcionamiento se ha llevado a cabo por usuarios en distintos ámbitos de estudios, lo que permite contrastar sus diferencias de manejo e introducción de información.

3.1.8. Octava fase (realización de la memoria)

Esta fase, que abarca la mayoría del proyecto, es quizás la más tediosa ya que no se trata más que de documentar todos los pasos que se van dando en el desarrollo de este proyecto.

3.1.9. Novena fase (desarrollo de la presentación)

Todo proyecto necesita ser expuesto por su autor y aquí es donde se le dedica tiempo a la realización de una presentación que, una vez expuesta, da a conocer a grandes rasgos, los pasos que se han seguido para el desarrollo de este proyecto.

3.2. Diagrama de Gantt

En la Figura 1 podemos ver el diagrama de Gantt desarrollado con GanttProject [2] correspondiente al desarrollo del proyecto.



Figura 1: Diagrama de Gantt

4. Análisis

4.1. Toma de requisitos

En este apartado se plantearán los requisitos funcionales y no funcionales para el sistema que se nos presenta.

4.1.1. Requisitos funcionales del sistema

El uso de LibGDX hace posible que las funcionalidades ofrecidas a ambos clientes sean idénticas. Ofreciendo la misma experiencia al usuario ya sea desde un PC o un dispositivo móvil.

- Registrar jugadores en el sistema. Se podrán añadir o eliminar tantos jugadores como sea preciso.
- Personalizar cada jugador. El sistema permitirá modificar las preferencias de cada jugador.
- Almacenar palabras y preguntas. Se podrán añadir y eliminar tantas palabras y preguntas como se desee, ya sea una por una o leyéndolas de un fichero de texto.
- Modificar las palabras y preguntas almacenadas. Las palabras y preguntas podrán ser modificadas desde la aplicación campo por campo.
- Mostrar las puntuaciones de cada jugador. El sistema mostrará las puntuaciones de cada juego para cada jugador.

- Mostrar un ranking absoluto de jugadores para cada juego. El sistema creará un ranking global de las diez mejores puntuaciones para cada juego.
- Elegir los jugadores con los que jugar. Antes de empezar una partida, el usuario podrá escoger qué jugadores participarán en ella.
- Finalizar una partida. El usuario podrá finalizar una partida en cualquier momento del juego.
- Establecer el tiempo restante. El juego *Rosco* permitirá establecer en cualquier momento el tiempo restante para cada jugador.
- Mostrar la clasificación de la última partida. Tras cada partida, se mostrará un ranking con las puntuaciones de los jugadores que han participado en ella.

4.1.2. Requisitos no funcionales del sistema

Al emplearse la misma implementación para ambos sistemas, los requisitos no funcionales deben ser tenidos en cuenta para el que menos prestaciones tenga de ellos, permitiendo así que la experiencia del usuario nunca se vea perjudicada.

- Eficiencia. Los recursos del sistema son usados de la mejor forma posible.
- Facilidad de uso. La interfaz es intuitiva y no presenta ninguna complejidad en su uso.
- Velocidad. El tratamiento de la información y cómo es mostrada se tiene en cuenta para favorecer la velocidad del sistema.
- Transferencia limitada. El acceso a memoria se limita al máximo para mejorar la experiencia del usuario.

4.2. Modelo de interacción del sistema

El siguiente diagrama representado en la Figura 2, muestra la interacción que lleva a cabo el cliente, ya sea Android o de escritorio, con el sistema, y este a su vez, con la interfaz y la base de datos.

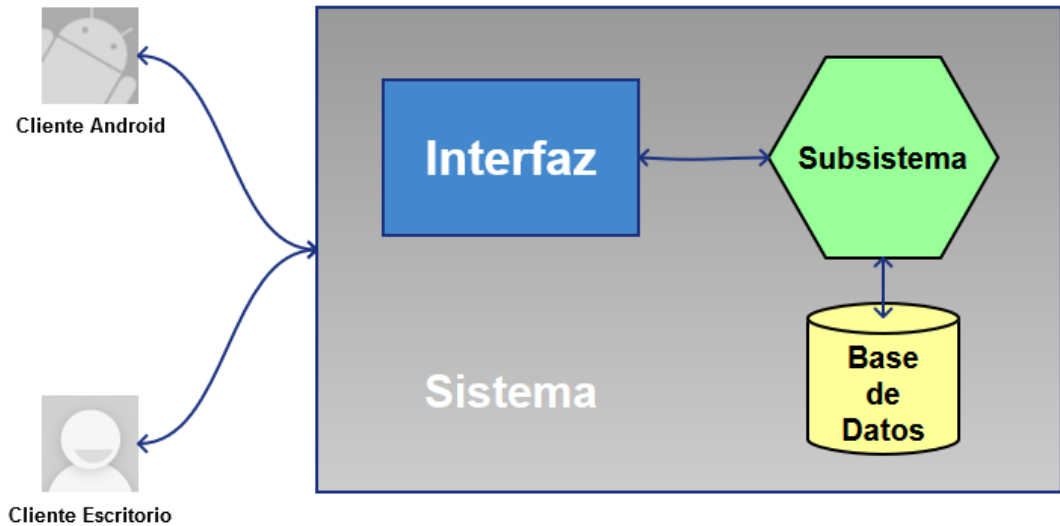


Figura 2: Diagrama interacción del sistema

4.3. Modelo de casos de uso

4.3.1. Modelo general de casos de uso

Primero se muestra el modelo general Figura 3, desarrollado con Pencil Project [9], y posteriormente se analizan individualmente cada uno de los casos de uso de la Tabla 1 a la Tabla 25:

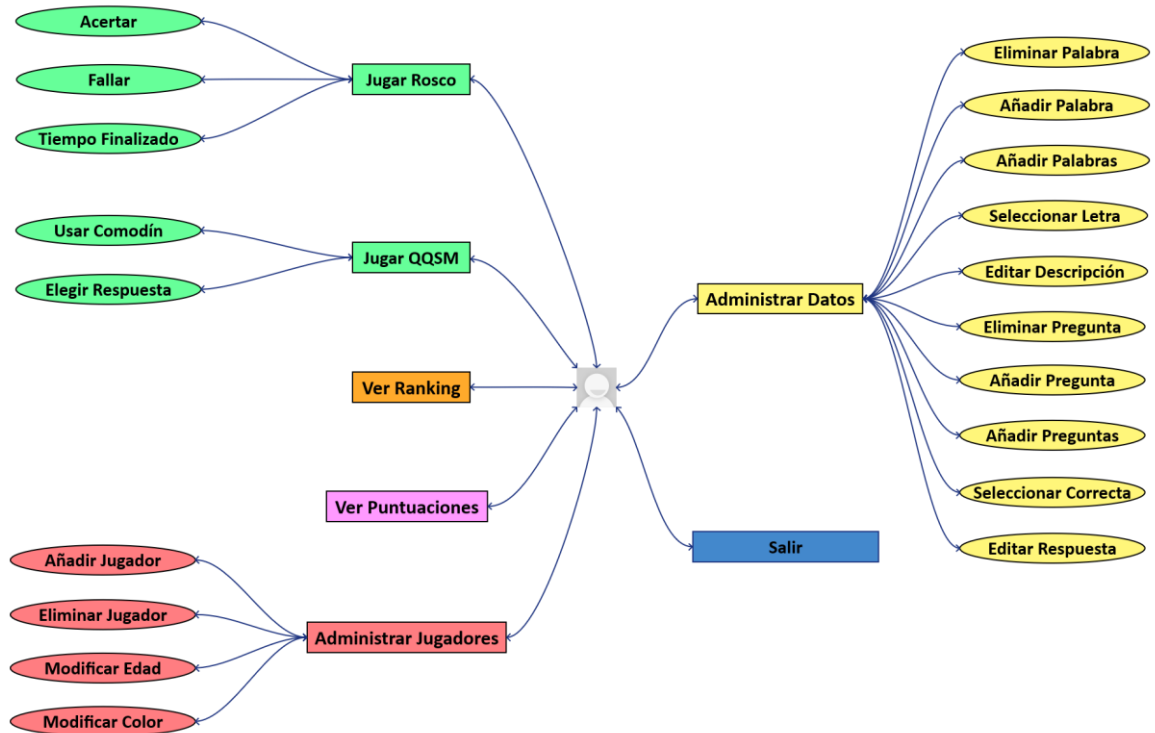


Figura 3: Diagrama general de casos de uso

Caso de uso	Jugar Rosco
Descripción	Jugar al juego Rosco
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú jugar 3. Se muestra el menú jugar 4. El usuario pulsa el botón del juego Rosco 5. Se muestra la pantalla del juego Rosco si hay algún jugador seleccionado

Tabla 1: Caso de uso (Jugar Rosco)

Caso de uso	Jugar <i>QQSM</i>
Descripción	Jugar al juego <i>QQSM</i>
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú jugar 3. Se muestra el menú jugar 4. El usuario pulsa el botón del juego <i>QQSM</i> 5. Se muestra la pantalla del juego <i>QQSM</i> si hay algún jugador seleccionado

*Tabla 2: Caso de uso (Jugar *QQSM*)*

Caso de uso	Ver Ranking
Descripción	Mostrar el ranking
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú ranking 3. Se muestra el menú ranking

Tabla 3: Caso de uso (Ver Ranking)

Caso de uso	Ver Puntuaciones
Descripción	Mostrar ver las puntuaciones
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú puntuaciones 3. Se muestra el menú puntuaciones 4. El usuario selecciona el jugador del que quiera ver las puntuaciones

Tabla 4: Caso de uso (Ver Puntuaciones)

Caso de uso	Administrar Jugadores
Descripción	Administrar los jugadores de la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú jugadores 3. Se muestra el menú jugadores

Tabla 5: Caso de uso (Administrar Jugadores)

Caso de uso	Salir
Descripción	Mostrar pantalla anterior o cerrar la aplicación
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra la pantalla en cuestión 2. El jugador selecciona el botón atrás o pulsa el botón secundario si se encuentra en la aplicación para escritorio, o pulsa el botón de retroceso o el botón home si se encuentra en la aplicación para Android 3. Se cierra la aplicación si se encuentra en el menú inicial
Escenario alternativo	<ol style="list-style-type: none"> 3.1. La pantalla actual es el menú jugar <ol style="list-style-type: none"> 3.1.1. Se muestra el menú inicio 3.2. La pantalla actual es el menú datos <ol style="list-style-type: none"> 3.2.1. Se actualizan en la base de datos los valores que hayan sido modificados y no hayan sido actualizados todavía 3.2.2. Se muestra el menú inicio 3.3. La pantalla actual es el menú ranking <ol style="list-style-type: none"> 3.3.1. Se muestra el menú inicio 3.4. La pantalla actual es el menú puntuaciones <ol style="list-style-type: none"> 3.4.1. Se muestra el menú inicio 3.5. El menú actual es el menú jugadores <ol style="list-style-type: none"> 3.5.1. Se muestra el menú inicio 3.6. La pantalla actual es el juego Rosco <ol style="list-style-type: none"> 3.6.1. Se muestra el menú jugar 3.7. La pantalla actual es el juego QQSM <ol style="list-style-type: none"> 3.7.1. Se muestra el menú jugar 3.8. La pantalla actual es clasificación <ol style="list-style-type: none"> 3.8.1. Se muestra el menú jugar

Tabla 6: Caso de uso (Salir)

Caso de uso	Administrar Datos
Descripción	Administrar la información de la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú de inicio 2. El jugador selecciona el botón de menú datos 3. Se muestra el menú datos 4. El usuario administra las palabras y preguntas de la base de datos

Tabla 7: Caso de uso (Administrar Datos)

Caso de uso	Acertar
Descripción	Marcar la letra actual como correcta
Escenario principal	<ol style="list-style-type: none"> 1. Se pulsa el botón Inicio / Pausa 2. El usuario pulsa el botón acierto 3. Se marca la letra actual como correcta 4. Se incrementa el contador de aciertos 5. Se muestra la siguiente palabra

Tabla 8: Caso de uso (Acertar)

Caso de uso	Fallar
Descripción	Marcar la letra actual como fallida
Escenario principal	<ol style="list-style-type: none"> 1. Se pulsa el botón Inicio / Pausa 2. El usuario pulsa el botón fallo 3. Se marca la letra actual como fallida 4. Se muestra la siguiente palabra

Tabla 9: Caso de uso (Fallar)

Caso de uso	Tiempo Finalizado
Descripción	El tiempo restante se acaba
Escenario principal	<ol style="list-style-type: none"> 1. El tiempo restante llega a cero 2. La partida finaliza para ese jugador

Tabla 10: Caso de uso (Tiempo Finalizado)

Caso de uso	Usar Comodín
Descripción	El usuario consume el comodín del que dispone
Escenario principal	<ol style="list-style-type: none"> 1. El usuario elige usar el comodín 2. El comodín es consumido y desaparece 3. Las respuestas posibles se reducen a la mitad

Tabla 11: Caso de uso (Usar Comodín)

Caso de uso	Elegir Respuesta
Descripción	Elegir una respuesta posible

Escenario principal	<ol style="list-style-type: none"> 1. El usuario elige una de las cuatro posibles respuestas 2. Se avanza a la siguiente pregunta si es correcta o finaliza la partida para ese jugador si es incorrecta
---------------------	--

Tabla 12: Caso de uso (Elegir Respuesta)

Caso de uso	Añadir Jugador
Descripción	Añadir un nuevo jugador
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú jugadores 2. El usuario el botón de añadir un nuevo jugador 3. El usuario escribe el nombre del nuevo jugador 4. Se añade el nuevo jugador a la base de datos con los valores por defecto y se muestran en el menú jugadores
Escenario alternativo	<ol style="list-style-type: none"> 3.1. El usuario cancela la introducción del nombre 3.1.1. Se muestra el menú jugadores sin modificar

Tabla 13: Caso de uso (Añadir Jugador)

Caso de uso	Eliminar Jugador
Descripción	Eliminar un jugador de la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú jugadores 2. El usuario pulsa el botón de eliminar un jugador 3. Se elimina el jugador mostrado de la base de datos incluidas sus puntuaciones

Tabla 14: Caso de uso (Eliminar Jugador)

Caso de uso	Modificar Edad
Descripción	Modificar edad de un jugador
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú jugadores 2. El usuario pulsa el botón modificar la edad 3. Se introduce la nueva edad del jugador 4. Se actualiza la edad del jugador seleccionado de la base de datos y se muestra en el menú jugadores
Escenario alternativo	<ol style="list-style-type: none"> 3.1. El usuario cancela la introducción de la edad 3.1.1. Se muestra el menú jugadores sin modificar

Tabla 15: Caso de uso (Modificar Edad)

Caso de uso	Modificar Color
Descripción	Modificar color de un jugador
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú jugadores 2. El usuario pulsa el botón de modificar el color 3. Se muestra la paleta de colores 4. Se selecciona un color de la paleta 5. Se actualiza el color de fondo del menú jugadores y de la base de datos del jugador seleccionado

Tabla 16: Caso de uso (Modificar Color)

Caso de uso	Eliminar Palabra
Descripción	Se elimina la palabra de la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de eliminar la palabra actual 3. Se elimina la palabra mostrada de la base de datos 4. Se muestra la siguiente palabra almacenada

Tabla 17: Caso de uso (Eliminar Palabra)

Caso de uso	Añadir Palabra
Descripción	Añadir una nueva palabra a la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de añadir una nueva palabra 3. Se escribe la nueva palabra 4. Se escribe la descripción de la nueva palabra 5. Se muestra la nueva palabra 6. Se muestra la primera letra de la nueva palabra 7. Se muestra la descripción de la nueva palabra
Escenario alternativo	<ol style="list-style-type: none"> 3.1. Se cancela la introducción de la nueva palabra <ol style="list-style-type: none"> 3.1.1. Se muestra el menú datos sin modificar 4.1. Se cancela la introducción de la nueva descripción <ol style="list-style-type: none"> 4.1.1. Se muestra el menú datos sin modificar

Tabla 18: Caso de uso (Añadir Palabra)

Caso de uso	Añadir Palabras
-------------	-----------------

Descripción	Añadir las palabra de un fichero a la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de añadir palabras de un fichero 3. Se selecciona el fichero 4. Se añaden las nuevas palabras a la base de datos 5. Se muestra la nueva palabra 6. Se muestra la letra añadida de la nueva palabra 7. Se muestra la descripción de la nueva palabra

Tabla 19: Caso de uso (Añadir Palabras)

Caso de uso	Seleccionar Letra
Descripción	Modificar la letra de la palabra
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario selecciona otra letra a mostrar 3. Se actualiza en la base de datos la nueva letra seleccionada de la palabra mostrada

Tabla 20: Caso de uso (Seleccionar Letra)

Caso de uso	Editar Descripción
Descripción	Modificar la descripción de la palabra
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario edita la descripción de la palabra mostrada 3. El usuario pulsa el botón “intro” y se actualiza la descripción en la base de datos
Escenario alternativo	3.1. El usuario no pulsa el botón “intro” y no se actualiza la descripción en la base de datos

Tabla 21: Caso de uso (Editar Descripción)

Caso de uso	Eliminar Pregunta
Descripción	Se elimina la pregunta de la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de eliminar la pregunta actual 3. Se elimina la pregunta mostrada de la base de datos 4. Se muestra la siguiente pregunta almacenada

Tabla 22: Caso de uso (Eliminar Pregunta)

Caso de uso	Añadir Pregunta
Descripción	Añadir una nueva pregunta a la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de añadir una nueva pregunta 3. Se escribe la nueva pregunta 4. Se escribe la respuesta correcta 5. Se escribe la primera respuesta incorrecta 6. Se escribe la segunda respuesta incorrecta 7. Se escribe la tercera respuesta incorrecta 8. Se muestra la nueva pregunta 9. Se muestra la respuesta correcta 10. Se muestran las respuestas incorrectas
Escenario alternativo	<ol style="list-style-type: none"> 3.2. Se cancela la introducción de la nueva palabra <ol style="list-style-type: none"> 3.1.2. Se muestra el menú datos sin modificar 4.2. Se cancela la introducción de la respuesta correcta <ol style="list-style-type: none"> 4.1.2. Se muestra el menú datos sin modificar 5.1. Se cancela la introducción de la primera respuesta incorrecta <ol style="list-style-type: none"> 5.1.1. Se muestra el menú datos sin modificar 6.1. Se cancela la introducción de la segunda respuesta incorrecta <ol style="list-style-type: none"> 6.1.1. Se muestra el menú datos sin modificar 7.1. Se cancela la introducción de la tercera respuesta incorrecta <ol style="list-style-type: none"> 7.1.1. Se muestra el menú datos sin modificar

Tabla 23: Caso de uso (Añadir Pregunta)

Caso de uso	Añadir Preguntas
Descripción	Añadir las preguntas de un fichero a la base de datos
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario pulsa el botón de añadir preguntas de un fichero 3. Se selecciona el fichero 4. Se añaden las nuevas preguntas a la base de datos 5. Se muestra la nueva pregunta 6. Se muestra la respuesta correcta de la nueva pregunta 7. Se muestran las respuestas de la nueva pregunta

Tabla 24: Caso de uso (Añadir Preguntas)

Caso de uso	Seleccionar Correcta
Descripción	Modificar la respuesta correcta de la pregunta
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario selecciona otra respuesta correcta 3. Se actualiza en la base de datos la nueva respuesta correcta seleccionada de la pregunta mostrada

Tabla 25: Caso de uso (Seleccionar Correcta)

Caso de uso	Editar respuesta
Descripción	Modificar la respuesta de la pregunta
Escenario principal	<ol style="list-style-type: none"> 1. Se muestra el menú datos 2. El usuario edita una respuesta de la pregunta mostrada 3. El usuario pulsa el botón “intro” y se actualiza la respuesta modificada en la base de datos
Escenario alternativo	<ol style="list-style-type: none"> 3.2. El usuario no pulsa el botón “intro” y no se actualiza la respuesta en la base de datos

Tabla 26: Caso de uso (Editar Respuesta)

4.4. Modelo de comportamiento del sistema

A continuación se detalla el comportamiento del sistema mediante diagramas de secuencia, de la Figura 4 a la Figura 25, desarrollados con Argo UML [1] que detallan los eventos que tienen lugar entre el usuario y el sistema, además de las condiciones y efectos que tienen estos eventos mostrados en tablas, desde la Tabla 26 a la Tabla 47:

4.4.1. Caso de uso: Jugar Rosco

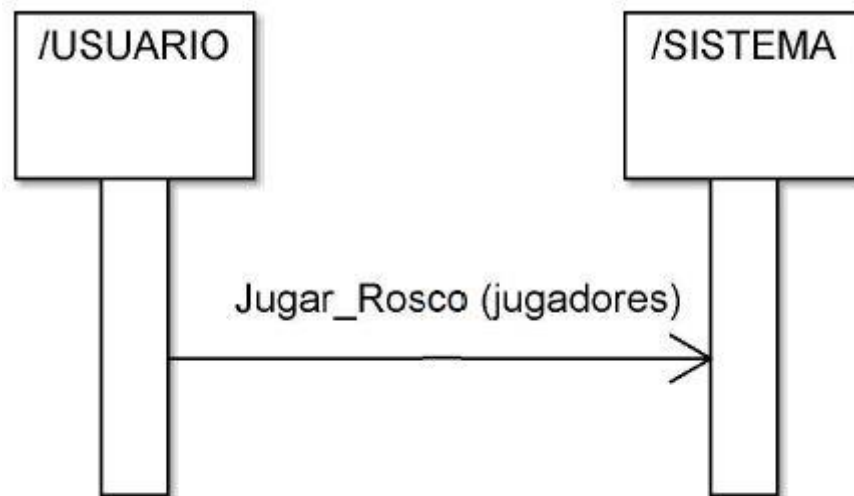


Figura 4: Diagrama de secuencia (Jugar Rosco)

Operación	Jugar_Rosco(jugadores)
Objetivo	Iniciar el juego Rosco.
Precondiciones	Jugadores no está vacío.
Postcondiciones	Se muestra la pantalla del juego Rosco y el primer jugador de jugadores.

Tabla 27: Diagrama de secuencia (Jugar Rosco)

4.4.2. Caso de uso: Jugar QQSM

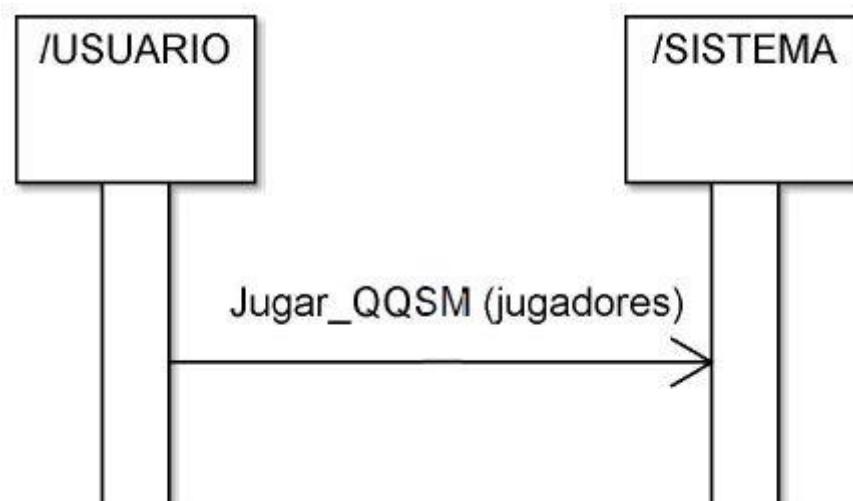


Figura 5: Diagrama de secuencia (Jugar QQSM)

Operación	Jugar_QQSM(jugadores)
Objetivo	Iniciar el juego Rosco.
Precondiciones	Jugadores no está vacío.
Postcondiciones	Se muestra la pantalla del juego QQSM y el primer jugador de jugadores.

Tabla 28: Diagrama de secuencia (Jugar QQSM)

4.4.3. Caso de uso: Tiempo Finalizado

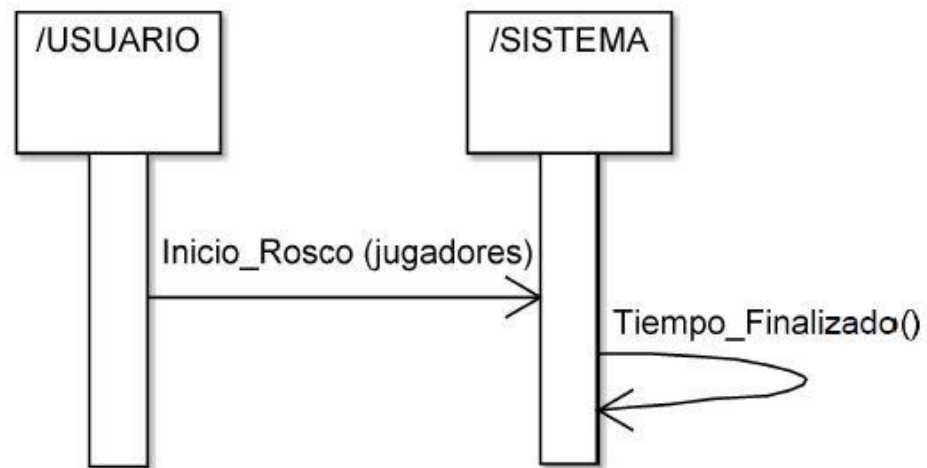


Figura 6: Diagrama de secuencia (Inicio Rosco)

Operación	Tiempo_Finalizado()
Objetivo	Finalizar partida.
Precondiciones	Ninguna.
Postcondiciones	Se pasa al siguiente jugador si queda alguno o se muestra la clasificación.

Tabla 29: Diagrama de secuencia (Tiempo Finalizado)

4.4.4. Caso de uso: Usar Comodín

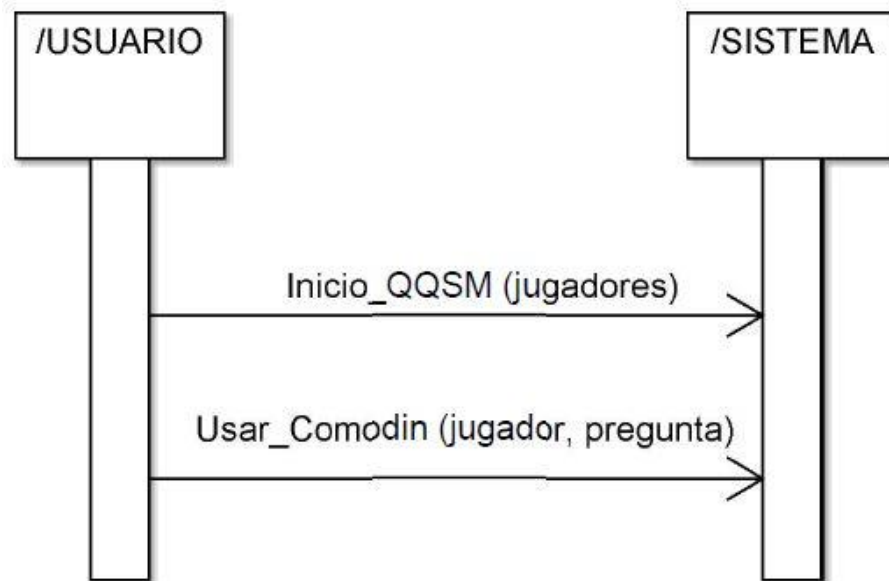


Figura 7: Diagrama de secuencia (Usar Comodín)

Operación	Usar_Comodin(jugador, pregunta)
Objetivo	Consumir comodín.
Precondiciones	El jugador actual ha iniciado la partida y no ha consumido el comodín en esta partida.
Postcondiciones	Se consume el comodín y se muestran la mitad de respuestas posibles.

Tabla 30: Diagrama de secuencia (Usar Comodín)

4.4.5. Caso de uso: Elegir Respuesta

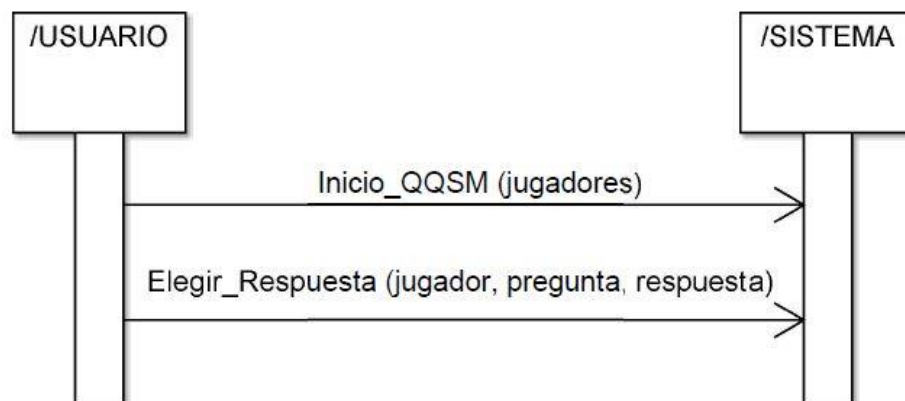


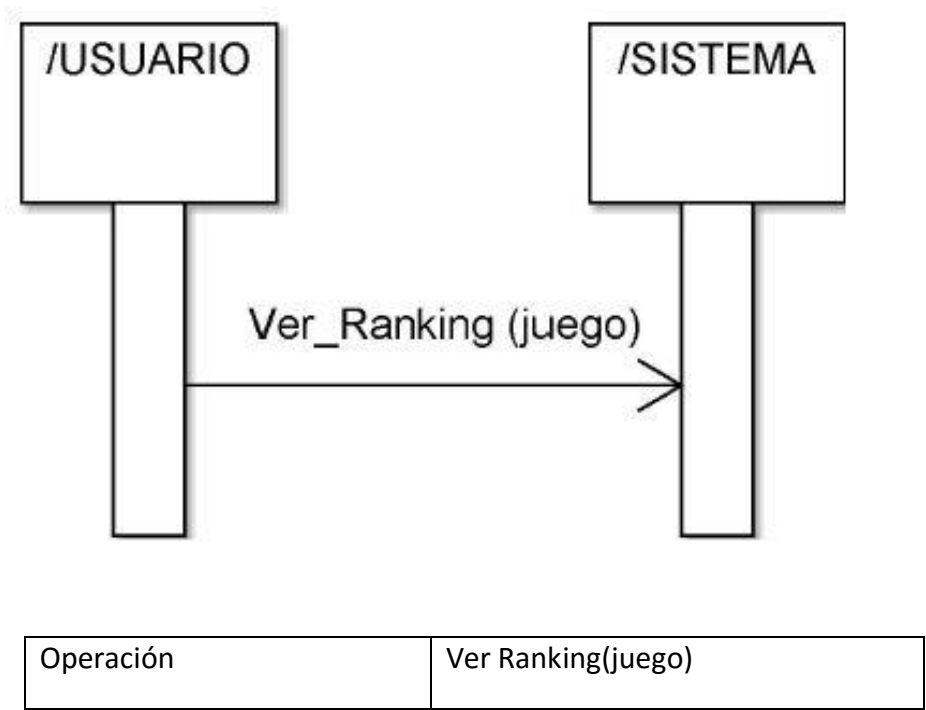
Figura 8: Diagrama de secuencia (Elegir Respuesta)

Operación	Elegir_Respuesta(jugador, pregunta, respuesta)
Objetivo	Responder a la pregunta actual.
Precondiciones	El jugador actual ha iniciado la partida.
Postcondiciones	Se incrementa el contador de aciertos y se muestra la siguiente pregunta o se incrementa el contador de fallos y se termina la partida del jugador actual.

Tabla 31: Diagrama de secuencia (Elegir Respuesta)

4.4.6. Caso de uso: Ver Ranking

Figura 9: Diagrama de secuencia (Ver Ranking)



Objetivo	Muestra las puntuaciones del juego.
Precondiciones	Ninguna.
Postcondiciones	Se muestra las 10 mejores puntuaciones almacenadas en la base de datos para el juego seleccionado.

Tabla 32: Diagrama de secuencia (Ver Ranking)

4.4.7. Caso de uso: Ver Puntuaciones

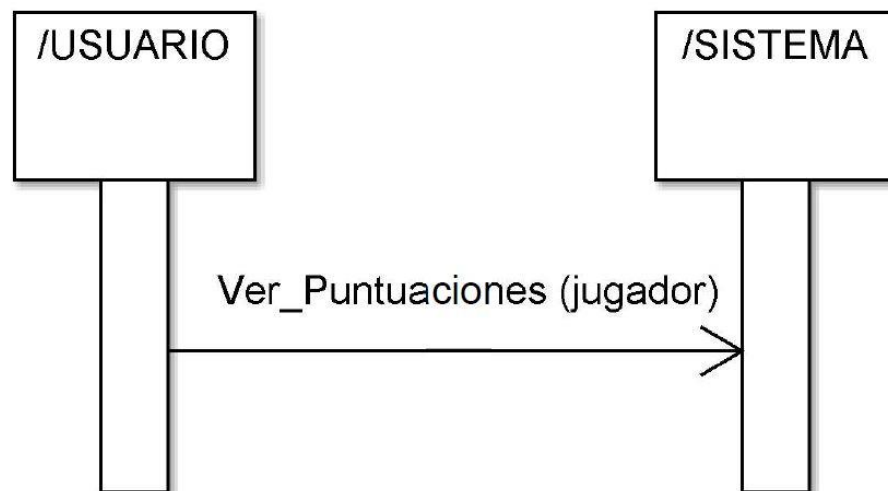


Figura 10: Diagrama de secuencia (Ver Puntuaciones)

Operación	Ver Puntuaciones(jugador)
Objetivo	Muestra las puntuaciones del jugador.
Precondiciones	Ninguna.
Postcondiciones	Se muestra las puntuaciones almacenadas en la base de datos para el jugador seleccionado.

Tabla 33: Diagrama de secuencia (Ver Puntuaciones)

4.4.8. Caso de uso: Añadir Jugador

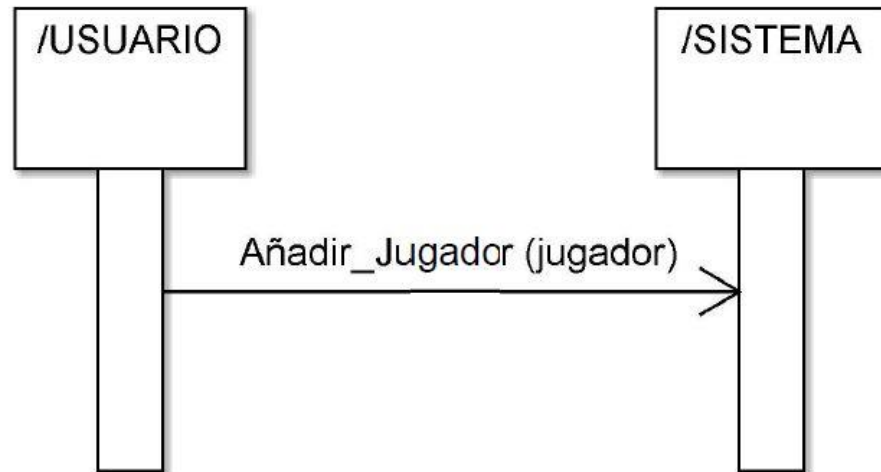


Figura 11: Diagrama de secuencia (Añadir Jugador)

Operación	Añadir_Jugador(jugador)
Objetivo	Añadir un nuevo jugador.
Precondiciones	Ninguna.
Postcondiciones	Se añade un nuevo jugador a la base de datos con la edad y el color por defecto.

Tabla 34: Diagrama de secuencia (Añadir Jugador)

4.4.9. Caso de uso: Eliminar Jugador

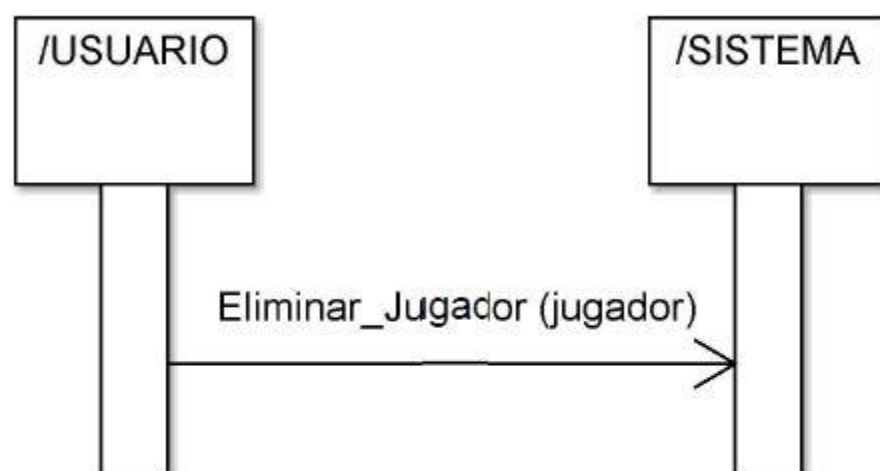


Figura 12: Diagrama de secuencia (Eliminar Jugador)

Operación	Eliminar_Jugador(jugador)
Objetivo	Eliminar el jugador actual.
Precondiciones	Existe al menos un jugador en la base de datos.
Postcondiciones	Se elimina el jugador mostrado de la base de datos junto con sus puntuaciones.

Tabla 35: Diagrama de secuencia (Eliminar Jugador)

4.4.10. Caso de uso: Modificar Edad

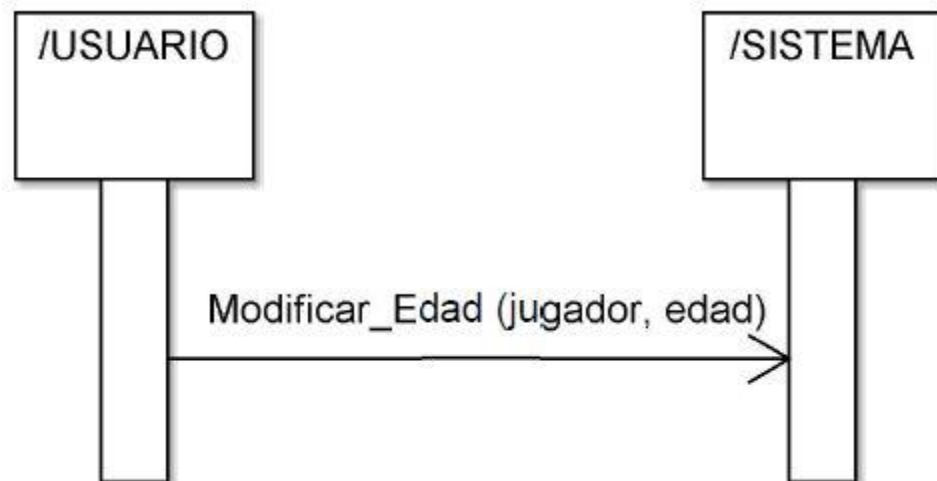


Figura 13: Diagrama de secuencia (Modificar Edad)

Operación	Modificar_Edad(jugador, edad)
Objetivo	Modificar la edad del jugador actual.
Precondiciones	Existe al menos un jugador en la base de datos.
Postcondiciones	Se modifica la edad del jugador actual.

Tabla 36: Diagrama de secuencia (Modificar Edad)

4.4.11. Caso de uso: Modificar Color

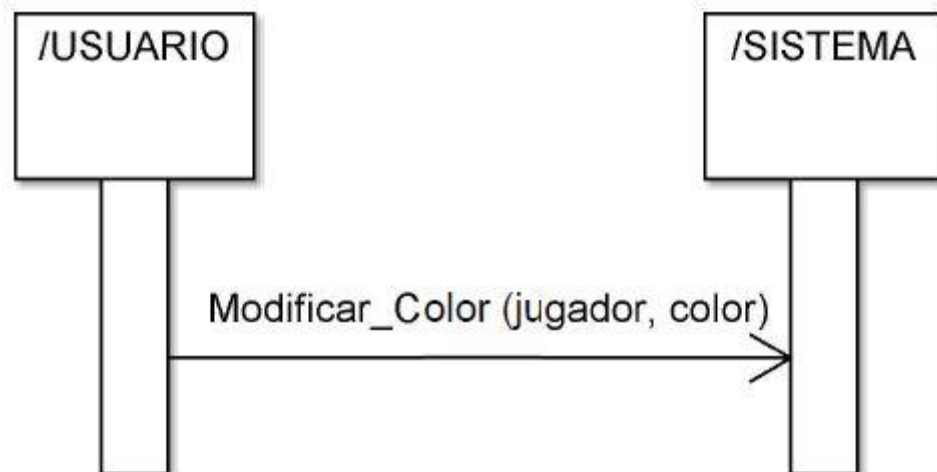


Figura 14: Diagrama de secuencia (Modificar Color)

Operación	Modificar_Color(jugador, color)
Objetivo	Modificar el color del jugador actual.
Precondiciones	Existe al menos un jugador en la base de datos.
Postcondiciones	Se modifica el color del jugador actual.

Tabla 37: Diagrama de secuencia (Modificar Color)

4.4.12. Caso de uso: Eliminar Palabra

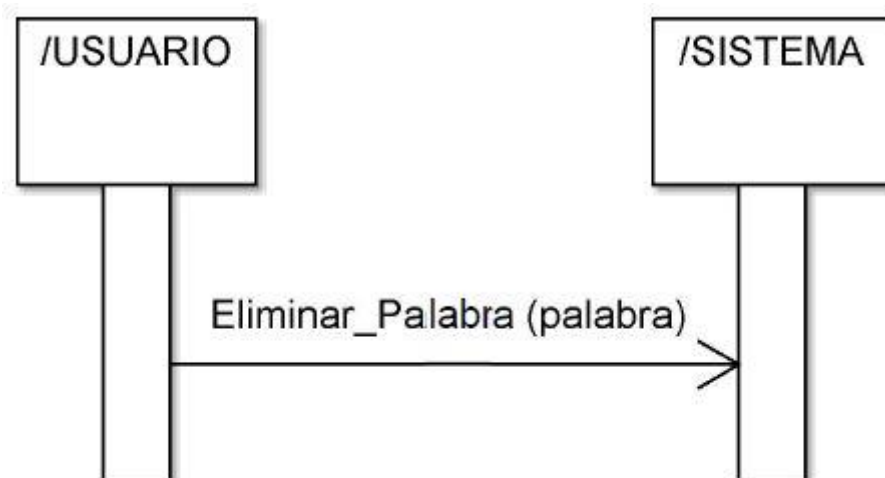


Figura 15: Diagrama de secuencia (Eliminar Palabra)

Operación	Eliminar_Palabra(palabra)
Objetivo	Eliminar palabra actual de la base de datos.
Precondiciones	Existe al menos una palabra en la base de datos.
Postcondiciones	Se elimina la palabra de la base de datos.

Tabla 38: Diagrama de secuencia (Eliminar Palabra)

4.4.13. Caso de uso: Añadir Palabra

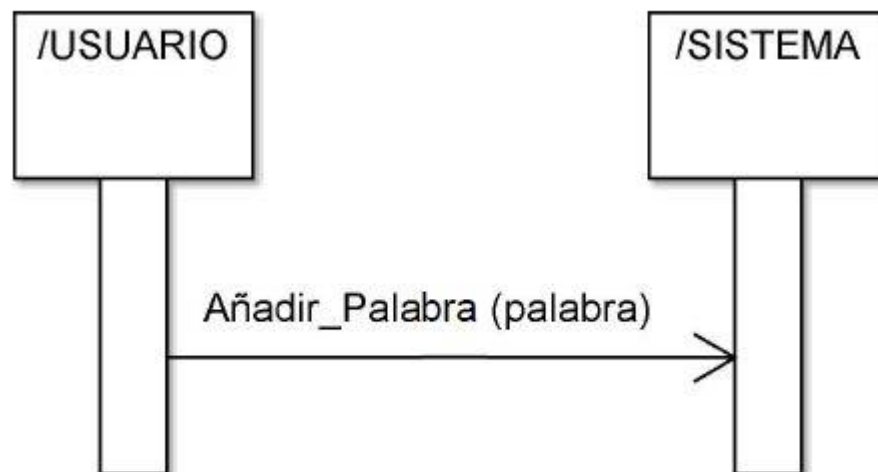


Figura 16: Diagrama de secuencia (Añadir Palabra)

Operación	Añadir_Palabra(palabra)
Objetivo	Añadir una nueva palabra a la base de datos.
Precondiciones	Ninguna.
Postcondiciones	Se añade una nueva palabra a la base de datos.

Tabla 39: Diagrama de secuencia (Añadir Palabra)

4.4.14. Caso de uso: Añadir Palabras

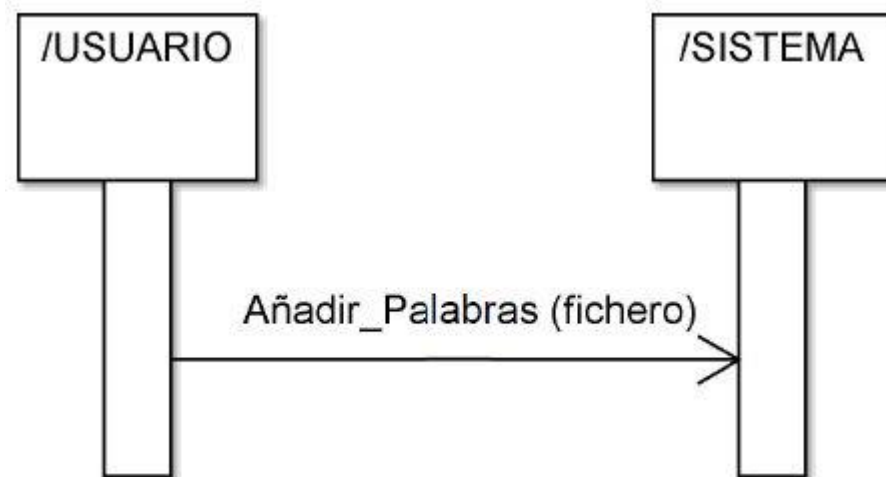


Figura 17: Diagrama de secuencia (Añadir Palabras)

Operación	Añadir_Palabras(fichero)
Objetivo	Añadir las palabras de un fichero a la base de datos.
Precondiciones	El fichero cumple los requisitos necesarios.
Postcondiciones	Se añaden las palabras contenidas en el fichero a la base de datos.

Tabla 40: Diagrama de secuencia (Añadir Palabras)

4.4.15. Caso de uso: Seleccionar Letra



Figura 18: Diagrama de secuencia (Seleccionar Letra)

Operación	Seleccionar_Letra(palabra, letra)
-----------	-----------------------------------

Objetivo	Seleccionar la letra de la palabra para ser mostrada en el Rosco.
Precondiciones	Existe al menos una palabra en la base de datos.
Postcondiciones	Se actualiza la letra de la palabra en la base de datos.

Tabla 41: Diagrama de secuencia (Seleccionar Letra)

4.4.16. Caso de uso: Editar Descripción

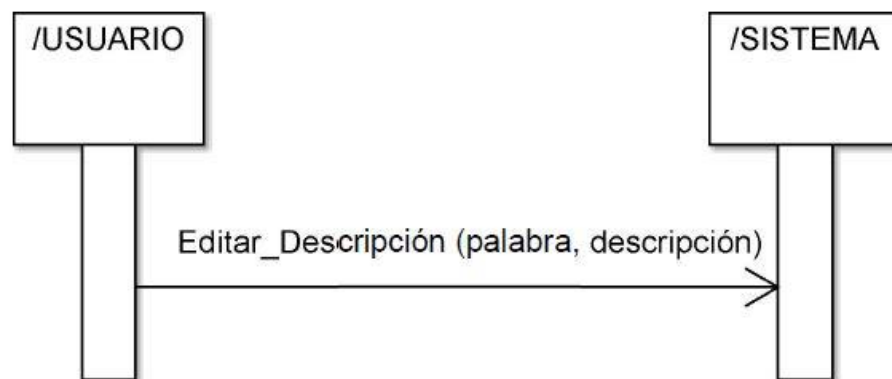


Figura 19: Diagrama de secuencia (Editar Descripción)

Operación	Editar_Descripción(palabra, descripción)
Objetivo	Actualizar la descripción de la palabra actual.
Precondiciones	Existe al menos una palabra en la base de datos.
Postcondiciones	Se actualiza la descripción de la palabra en la base de datos.

Tabla 42: Diagrama de secuencia (Editar Descripción)

4.4.17. Caso de uso: Eliminar Pregunta

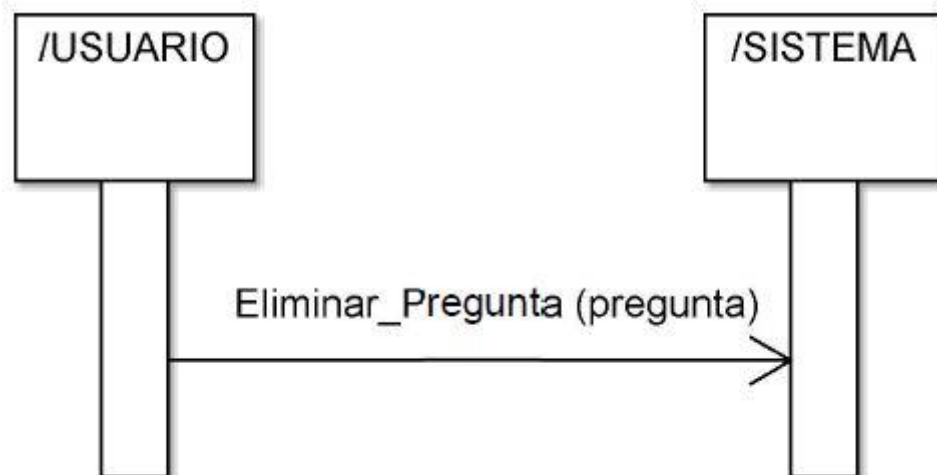


Figura 20: Diagrama de secuencia (Eliminar Pregunta)

Operación	Eliminar_Pregunta(pregunta)
Objetivo	Eliminar pregunta actual de la base de datos.
Precondiciones	Existe al menos una pregunta en la base de datos.
Postcondiciones	Se elimina la pregunta de la base de datos.

Tabla 43: Diagrama de secuencia (Eliminar Pregunta)

4.4.18. Caso de uso: Añadir Pregunta

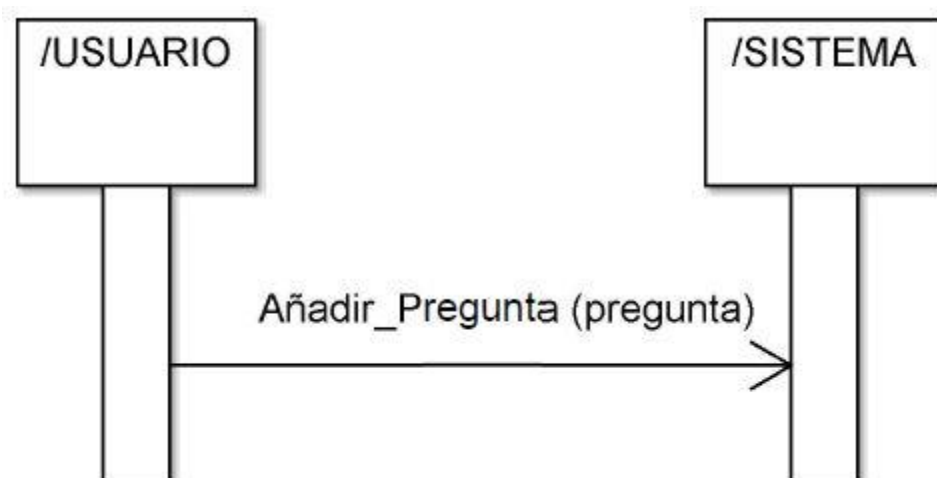


Figura 21: Diagrama de secuencia (Añadir Pregunta)

Operación	Añadir_Pregunta(pregunta)
Objetivo	Añadir una nueva pregunta a la base de datos.
Precondiciones	Ninguna.
Postcondiciones	Se añade una nueva pregunta a la base de datos.

Tabla 44: Diagrama de secuencia (Añadir Pregunta)

4.4.19. Caso de uso: Añadir Preguntas

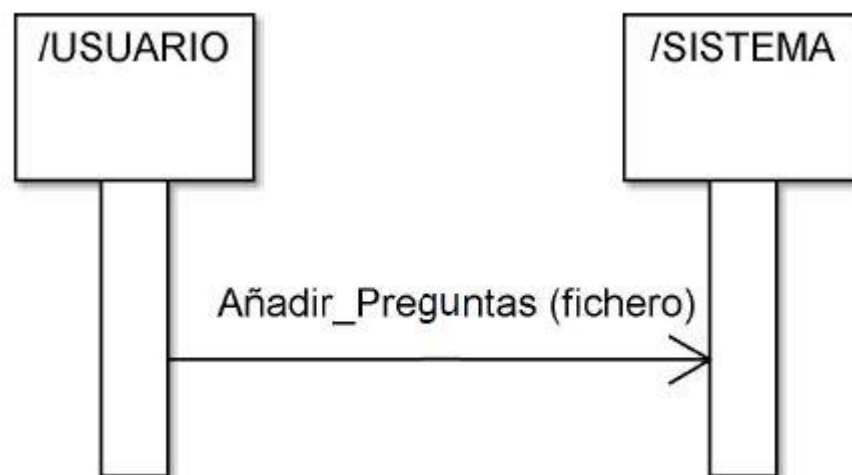


Figura 22: Diagrama de secuencia (Añadir Preguntas)

Operación	Añadir_Preguntas(fichero)
Objetivo	Añadir las preguntas de un fichero a la base de datos.
Precondiciones	El fichero cumple los requisitos necesarios.
Postcondiciones	Se añaden las preguntas contenidas en el fichero a la base de datos.

Tabla 45: Diagrama de secuencia (Añadir Preguntas)

4.4.20. Caso de uso: Seleccionar Correcta

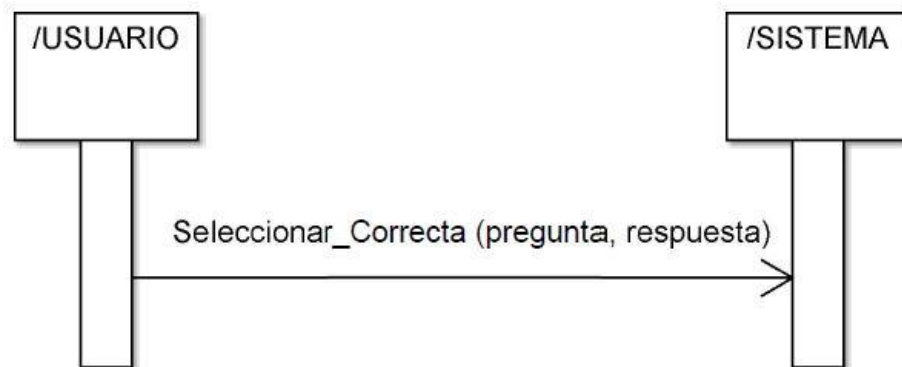


Figura 23: Diagrama de secuencia (Seleccionar Correcta)

Operación	Seleccionar_Correcta(pregunta, respuesta)
Objetivo	Seleccionar la respuesta correcta de la pregunta actual.
Precondiciones	Existe al menos una pregunta en la base de datos.
Postcondiciones	Se actualiza la respuesta correcta de la pregunta en la base de datos.

Tabla 46: Diagrama de secuencia (Seleccionar Correcta)

4.4.21. Caso de uso: Editar Respuesta

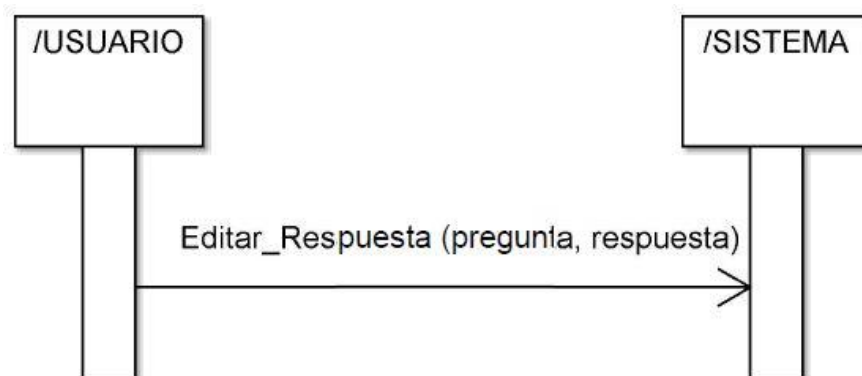


Figura 24: Diagrama de secuencia (Editar Respuesta)

Operación	Editar_Respuesta(pregunta, respuesta)
Objetivo	Actualizar la respuesta de la pregunta actual.
Precondiciones	Existe al menos una pregunta en la base de datos.
Postcondiciones	Se actualiza la respuesta de la pregunta en la base de datos.

Tabla 47: Diagrama de secuencia (Editar Respuesta)

4.4.22. Caso de uso: Salir

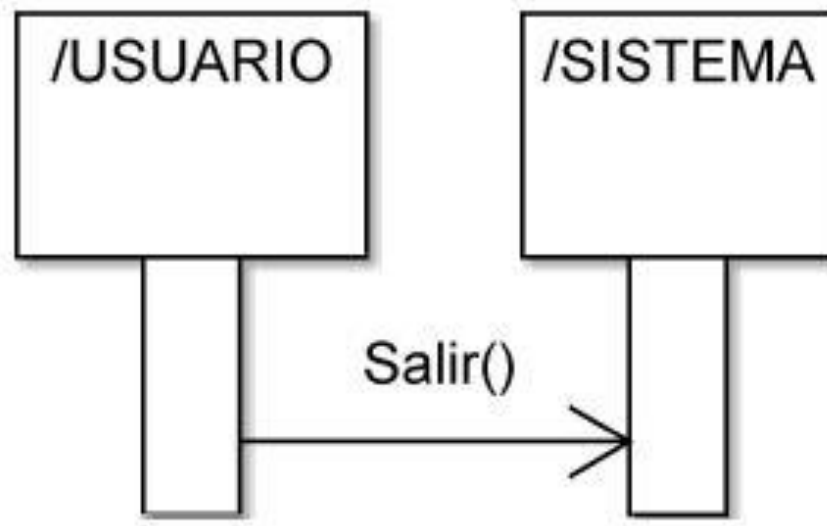


Figura 25: Diagrama de secuencia (Salir)

Operación	Salir()
Objetivo	Salir del menú actual.
Precondiciones	Ninguna.
Postcondiciones	Se muestra la pantalla anterior. Se muestra el menú jugar si la pantalla actual es algún juego o la pantalla de clasificación. Se sale del programa si la pantalla actual es el menú inicial.

Tabla 48: Diagrama de secuencia (Salir)

4.5. Modelo conceptual de datos

En la Figura 26 podemos ver el diagrama conceptual de clases.

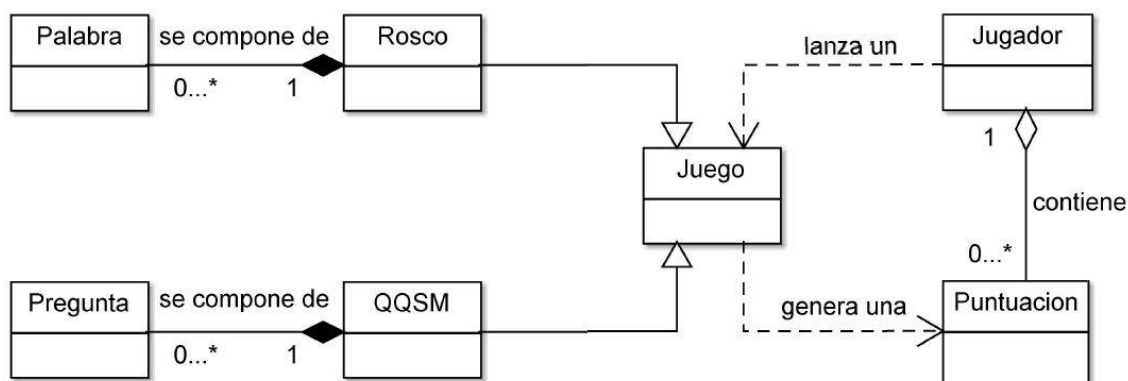


Figura 26: Diagrama conceptual de clases

5. Diseño

En este capítulo recorreremos todas las posibles pantallas del juego y veremos todos los elementos gráficos que se han utilizado, así como una breve explicación de las herramientas y los métodos utilizados para desarrollarlos.

Así mismo se presenta el diseño de las bases de datos y su organización.

5.1. Interfaz

Para el desarrollo de la interfaz se han utilizado las versiones gratuitas de las siguientes herramientas, que aunque incluyen menos características, son suficientes para el proyecto que nos ocupa:

5.1.1. Adobe Photoshop CC

Photoshop [10] es un editor de gráficos. Se ha usado para el diseño de todas las pantallas del juego al ser bastante versátil para el nivel de diseño que requiere el juego.

5.1.2. Vector Magic

Vector Magic [13] es un editor de gráficos vectoriales. Se ha usado para vectorizar todas las imágenes que se muestran en el juego. Las imágenes eran diseñadas con Photoshop y posteriormente se vectorizaban y redimensionaban con Vector Magic para aumentar su resolución y que la imagen estuviera más definida.

5.2. Menús

Existen cinco menús principales por los que el usuario puede desplazarse y visualizar todo el contenido de la aplicación.

- El primero es el menú de introducción de datos, en el que se pueden visualizar los jugadores, las palabras y las preguntas almacenadas en la base de datos, siendo posible modificar, añadir o eliminar palabras y preguntas desde este menú.
- El segundo es el menú de administración del perfil de los jugadores, en él se pueden añadir o eliminar jugadores, así como modificar las preferencias de cada uno de ellos.
- El tercero es el menú que permite la visualización de las puntuaciones registradas para cada jugador, en este menú se puede seleccionar cualquier jugador y ver todas las puntuaciones almacenadas de cada uno de los juegos existentes.
- El cuarto es el menú en el que se puede visualizar un ranking de las diez mejores puntuaciones almacenadas para el juego seleccionado.
- Y por último el menú que contiene una lista de los jugadores del sistema para poder ser añadidos para la siguiente partida en cualquiera de los dos juegos.

5.3. Base de datos

La aplicación debe almacenar distintos tipos de datos que deben estar accesibles por el usuario cada vez que inicie una nueva sesión. Para ello es necesaria la utilización de una base de datos que sea capaz de almacenar,

consultar y modificar estos datos de la forma más eficiente posible. El caso que nos ocupa presenta la necesidad de usar un sistema de gestión relacional ya que los vínculos existentes entre entidades así lo requieren.

Por lo tanto se decidió el uso del sistema de gestión SQLite. Este sistema presenta la ventaja de su pequeño tamaño y su eficiencia, lo que lo hace idóneo para su uso en dispositivos móviles.

Por otro lado, se decide crear una base de datos independiente para cada entidad, esto es necesario si se pretende alcanzar el objetivo de conseguir que el programa sea escalable en cuanto a añadir nuevos juegos se refiere. Es por ello que existirá una base de datos para almacenar los jugadores y otra base de datos por cada juego.

Teniendo esto en cuenta, la organización de las bases de datos es la que se muestra en la Figura 27:

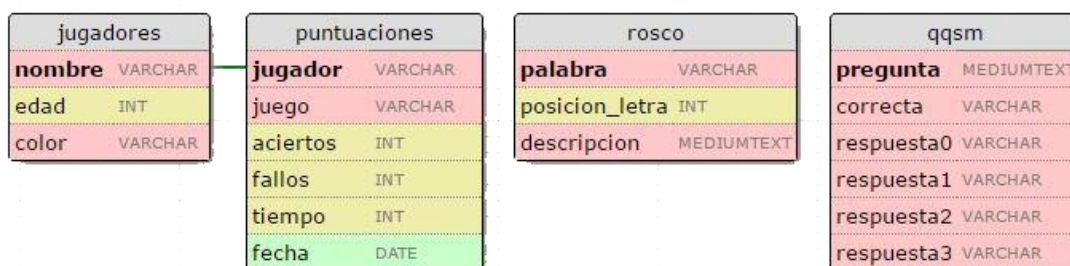


Figura 27: Bases de datos

En primer lugar la base de datos que contiene los jugadores estaría formada por las tablas “jugadores” y “puntuaciones”, siendo “nombre” la clave primaria de la tabla “jugadores” y a su vez clave foránea para la clave primaria “jugador” de la tabla “puntuaciones”.

En segundo lugar tenemos la base de datos del juego *Rosco*, que contiene la tabla “rosco” cuya clave primaria es el campo “palabra”.

Por último tenemos la base de datos del juego *QQSM*, que contiene la tabla “qqsm” cuya clave primaria es el campo “pregunta”.

6. Implementación

6.1. Tecnologías

A continuación se enumeran las tecnologías a utilizar en la realización del proyecto:

- El entorno de desarrollo utilizado es IntelliJ IDEA 2016. [8]
IntelliJ IDEA es un entorno de desarrollo muy versátil, lo que lo hace bastante atractivo. Además tiene disponible la versión Community Edition que es gratuita y contiene todas las herramientas necesarias para desarrollar proyectos de cualquier magnitud.
Este entorno permite la integración de Git [3] para el control de versiones ya que el proyecto ha sido desarrollado en diferentes computadoras.
Por otro lado la adhesión de plugins como el que permite la visualización de las base de datos, nos facilita la depuración del código además de la administración de la propia base de datos.
- El framework LibGDX para el desarrollo de videojuegos multiplataforma. Este framework posee una biblioteca que favorece la simplicidad en la implementación del proyecto, además su principal atractivo es la posibilidad de exportar el proyecto a diferentes plataformas sin tener que adaptar el proyecto a todas ellas, todo ello gracias al uso de Java como lenguaje de programación.
- Gradle [4] como herramienta de automatización de la construcción del código.
La utilización de LibGDX como framework para el desarrollo del proyecto plantea la necesidad de integrar diferentes librerías y herramientas que hacen del ensamblado del código algo no muy simple. Es entonces necesaria la utilización de Gradle como herramienta que favorece esta labor de construcción del código.
- Para la base de datos se empleará SQLite.
Para la gestión de la base de datos se ha decidido usar SQLite ya que su simplicidad favorece la reducción de la latencia en la administración y consulta.

El uso del proyecto en distintas plataformas como son las referentes a dispositivos móviles hacen de SQLite la herramienta perfecta por su pequeño tamaño.

6.2. Código

A continuación se muestran ejemplos de los principales fragmentos de código que emplea el programa para su funcionamiento.

- Jugador:

Un jugador será una clase que se implementará de la siguiente forma²:

```
public class Jugador {

    public String nombre = "Jugador";
    public int edad = 99;
    public Color color = Color.BLUE;

    public boolean seleccionado = true, jugando = false;

    public Integer letra_actual = -1;
    public int letras_visitadas[] = new int[26];

    public Integer n_aciertos = 0, n_fallos = 0;
    public int tiempo_qqsm = 0, tiempo_rosco = 150;
    public float contador = 0;
    public boolean comodin50_usado = false;

    public Jugador (String nombre, int edad, Color color) {
        this.nombre = nombre;
        this.edad = edad;
        this.color = color;
        BD.anadir_jugador(nombre, edad, color.toString());
    }

}
```

Y sus métodos modificadores serán:

```
public void edad (int edad) {
    this.edad = edad;
    BD.actualizar_edad(nombre, edad);
}

public void color (Color color) {
    this.color = color;
}
```

² El código se muestra mediante capturas de pantalla para facilitar su comprensión por el uso de colores del editor

```
BD.actualizar_color(nombre, color.toString());
}
```

- Jugadores:

Esta es la clase que se encarga de administrar los jugadores que existen en el sistema y que son los que participaran en cada uno de los juegos.

```
public class Jugadores {

    ArrayList<Jugador> coleccion = new ArrayList<>();
    private int jugador_actual = -1;

    public Jugadores() {

        BD.leer_jugadores(coleccion);
        if (!vacio())
            jugador_actual = 0;
    }
}
```

Y sus principales métodos de consulta:

```
//Se encarga de encontrar al siguiente jugador que este seleccionado
public boolean siguiente() {
    jugadores.jugador_actual().jugando = false;

    for (int i = jugador_actual; i < coleccion.size(); ++i) {
        if (i + 1 >= coleccion.size()) //Caso dar la vuelta hasta el
primer jugador
            i = -2;
        else
            if (coleccion.get(i + 1).seleccionado) { //Caso siguiente
jugador
                jugador_actual = i + 1;
                return true;
            }
        else
            if (i == jugador_actual - 1) //Caso ningun jugador restante
despues de dar la vuelta
                return false;
    }

    return true;
}
```

- Pantalla:

Esta clase permite abstraernos de la resolución de pantalla actual mediante el uso de funciones que transforman proporciones en píxeles:

```
public class Pantalla implements Screen {

    //Medidas
    public float anchura_juego = Gdx.graphics.getWidth();
    public float altura_juego = Gdx.graphics.getHeight();

    public float proporcion_x (double valor) {
        return (float) (anchura_juego * valor);
    }

    public float proporcion_y (double valor) {
        return (float) (altura_juego * valor);
    }
}
```

- **Menú Datos:**

A continuación se muestra el método de actualización de los datos mostrados para las palabras empleadas en el juego *Rosco*:

```
public void actualizar_palabras() {

    palabra_selector.setItems(BD.leer_columna_rosco("palabra").split("\n"));
    palabra_selector.pack();
    Palabra palabra =
    BD.leer_palabra((String)palabra_selector.getSelected());
    letra_selector.setItems(palabra.palabra.split(""));
    letra_selector.setSelectedIndex(palabra.posicion_letra);
    letra_selector.pack();
    descripcion_editor.setText(palabra.descripcion);
}
```

- **Menú Jugadores:**

En este ejemplo se puede apreciar como la mayoría de botones empleados en el programa, son rectángulos creados en tiempo de ejecución, que detectan si la pulsación se produce en el área que delimitan y es entonces cuando ejecutan la acción pertinente:

```
final Rectangle selector_edad = new Rectangle(anchura_juego
- proporcion_y(0.3), proporcion_y(0.7), proporcion_y(0.2),
proporcion_y(0.2));

if (selector_edad.contains(x, y))
//Selector de la edad
Gdx.input.getTextInput (new Input.TextInputListener() {
    public void input(String edad) {
```

```

        if (edad.isEmpty())
            edad = "Edad: 99 años";

jugadores.jugador(jugador_selector.getSelected()).edad(Integer.valueOf(edad));
jugador_edad.setText("Edad: " + edad + " años");
    }

    public void canceled() {}

}, "Introduzca la edad", "", "99");

```

- **Menú Ranking:**

Para crear el listado de puntuaciones se emplea el siguiente método:

```

public void crear_ranking() {
    rosco_ranking = "";
    qqsm_ranking = "";
    ArrayList<Puntuacion> puntuaciones_rosco =
BD.leer_puntuaciones("Rosco");
    ArrayList<Puntuacion> puntuaciones_qqsm =
BD.leer_puntuaciones("QQSM");
    for (int i = 0; i < puntuaciones_rosco.size(); ++i)
        rosco_ranking += (i + 1) + " posición -> " +
puntuaciones_rosco.get(i).jugador + ":\n" +
puntuaciones_rosco.get(i).puntuacion_mostrar;

    for (int i = 0; i < puntuaciones_qqsm.size(); ++i)
        qqsm_ranking += (i + 1) + " posición -> " +
puntuaciones_qqsm.get(i).jugador + ":\n" +
puntuaciones_qqsm.get(i).puntuacion_mostrar;
}

```

- **Juego QQSM:**

Para el juego *QQSM* se utiliza el siguiente método que nos permite mostrar una pregunta al azar de todas las que existen en la base de datos:

```

public void siguiente_pregunta() {
    if (preguntas.size() == 0)
        siguiente_jugador();
    else {
        int pregunta_aleatoria = MathUtils.random(preguntas.size() - 1);
        pregunta_actual = preguntas.get(pregunta_aleatoria);
        preguntas.remove(pregunta_actual);
    }
}

```

- **Juego Rosco:**

Este juego es el más complejo en lo referente a la implementación ya que se deben tener varios factores en cuenta a medida que se va

completando el *Rosco*. Como ejemplo se muestra la implementación del botón acierto, cuya labor es avanzar de letra o de jugador:

```
if (boton_acierto.contains(x, y)) {
    if (!jugadores.vacio())
        if (jugadores.jugador_actual().jugando) {

jugadores.jugador_actual().letras_visitadas[jugadores.jugador_actual()
.letra_actual] = 1; //letra actual visitada
        ++jugadores.jugador_actual().n_aciertos;

            if ((jugadores.jugador_actual().n_aciertos +
jugadores.jugador_actual().n_fallos) < 26) //no se ha contestado todo
                letra_siguiente();
            else //se ha completado el rosco
                jugador_siguiente();
        }
    }
}
```

También es interesante mostrar la implementación del método que se encarga de pasar al jugador siguiente o terminar la partida:

```
public void jugador_siguiente() {
    letra_descripcion_actual[0] = "";
    //Mostrar el texto de la letra actual en blanco
    letra_descripcion_etiqueta.setText(letra_descripcion_actual[0]);

    //si se ha completado el rosco o se ha acabado el tiempo se añade la
    puntuacion a la clasificacion
    if ((jugadores.jugador_actual().n_aciertos +
jugadores.jugador_actual().n_fallos) >= 26 ||
jugadores.jugador_actual().tiempo_rosco < 1) {
        jugadores.jugador_actual().seleccionado = false;
        clasificacion.puntuacion_anadir(jugadores.jugador_actual().nombre,
jugadores.jugador_actual().guardar_puntuacion("Rosco"));
    }

    //Si no hay mas jugadores se muestra la clasificacion
    if (!jugadores.siguiente())
        juego.setScreen(clasificacion);

    //Mostrar el nombre del jugador actual
    if (!jugadores.vacio()) {
nombre_jugador_etiqueta.setText(jugadores.jugador_actual().nombre); }
}
```

7. Pruebas

Como todos sabemos, la comprobación del correcto funcionamiento de todas las partes que forman el proyecto, es esencial para conocer que nuestro programa se

ejecuta justo como está previsto que lo haga, esto es sin fallos. Un pequeño error en una línea de código o una modificación no esperada del contenido de una variable puede ocasionar un mal funcionamiento del programa.

Es por ello necesario ir realizando constantemente pruebas al sistema con el objetivo de encontrar estos errores y corregirlos.

Las diferentes tipos de pruebas que se han ido realizando a lo largo del desarrollo del proyecto son las siguientes:

7.1. Pruebas unitarias

A medida que se desarrolla el proyecto, se van creando nuevos elementos que han de ser añadidos al mismo. Estos elementos han de ser probados antes de ser integrados al proyecto para poder identificar con mayor rapidez si existe algún error en ellos. Se deberán realizar el mayor número de pruebas posibles hasta que se esté seguro que el nuevo elemento a integrar está completamente libre de errores.

7.2. Pruebas de integración

Una vez que un nuevo elemento se integra en el proyecto, es necesario comprobar la correcta interacción de este con los demás elementos existentes. En nuestro caso, las pruebas se realizaron con la integración de los siguientes elementos:

7.2.1. Menú Inicial:

Este menú solo requería la comprobación referente a la navegación entre menús así como la salida del programa correctamente.

7.2.2. Menú Datos:

Este menú es el encargado de la administración de toda la información almacenada en la base de datos, a excepción de los jugadores que únicamente son mostrados.

Hubo que comprobar por tanto que la modificación de cada campo, ya sea de las palabras o de las preguntas que se muestran, se trasladaba a la base de datos y a su vez se mostraban correctamente modificados de nuevo en el menú.

7.2.3. Menú Jugadores:

Lo más importante en este menú es comprobar que el color y la edad de cada jugador se modificaban correctamente en la base de datos. Por otro lado se comprobó que las puntuaciones referentes al jugador eliminado también eran eliminadas.

7.2.4. Menú Puntuaciones:

En este menú se comprueba que las puntuaciones se muestran en el orden que se van realizando para poder comprobar el progreso del jugador a lo largo del tiempo.

7.2.5. Menú Ranking:

Para mostrar el ranking se ha de comprobar que el orden de las puntuaciones se realiza correctamente campo por campo de cada puntuación.

7.2.6. Menú Jugar:

Para este menú se debe comprobar que se muestren en él todos los jugadores contenidos en la base de datos y que los jugadores que se marquen, sean los que participen en el juego posteriormente seleccionado.

7.2.7. Juego Rosco:

La mecánica de este juego es quizás la más compleja de probar de todo el programa. Es por ello que se realizaron un gran número de pruebas

para comprobar su correcto funcionamiento, ya sea dejando letras sin responder, dejando que se acabase el tiempo o pruebas referentes al cambio de jugador, mostrando posteriormente el mismo estado del juego que poseía antes de cambiar de jugador.

7.2.8. Juego QQSM:

Este juego presenta la dificultad de que solo hay una respuesta correcta, lo que nos complica la implementación del uso del comodín del 50% o de la elección de la respuesta correcta. Controlar que todas las preguntas de la base de datos son mostradas aleatoriamente también fue motivo de comprobación.

7.2.9. Clasificación:

Esta pantalla es la encargada de mostrar el estado final de la partida que acaba de terminar, por tanto se ha de comprobar que solo aparecen los jugadores que acaban de participar en la partida anterior y que se muestran las puntuaciones que les corresponden.

7.3. Pruebas funcionales

Estas pruebas verifican que el funcionamiento del programa es justamente el definido en el modelo de caso de usos, esto es, el desplazamiento por los menús es el correcto, se navega de uno a otro como se espera y cada botón pulsado realiza, una vez más, el efecto que se ha descrito en el modelo de casos de uso.

7.4. Pruebas no funcionales

En este caso se ha observado que, independientemente del entorno en el que se ejecute el programa, se tiene una buena experiencia de uso, es decir, los tiempos de navegación entre menús son aceptables, el acceso a la base de datos es razonable y en definitiva la respuesta del programa entra en los parámetros de tiempo esperados.

7.5. Pruebas de aceptación

Por último, se comprueba que la experiencia del usuario es la esperada. Se repartió el juego entre varias personas, relacionadas o no con el mundo de la informática, que probaron el juego y transmitieron su satisfactoria experiencia de uso además de consejos y posibles ampliaciones.

8. Conclusiones

8.1. Síntesis

El presente proyecto ha sido desarrollado con la ayuda de Guadalupe Ortiz Bellot, del departamento de Ingeniería Informática de la Escuela Superior de Ingeniería de la Universidad de Cádiz.

El principal objetivo de este proyecto, como ya se ha mencionado varias veces a lo largo de este documento, es potenciar el aprendizaje de una forma lúdica, empleando para ello una aplicación multiplataforma que, simulando concursos televisivos, permite al usuario aprender diversas terminologías independientemente del campo de estudio que se plantee.

La aplicación abarca los siguientes ámbitos:

- Poseer un registro de los jugadores y sus preferencias.
- Mantener un conjunto de datos que representen los conceptos que se quieran adquirir.
- Mostrar el progreso de cada jugador.
- Crear un ranking a nivel global que muestre las mejores puntuaciones.

8.2. Trabajo futuro

Gracias al uso de LibGDX podemos desarrollar, como se ha comentado a lo largo del proyecto, diferentes clientes para poder ejecutar el programa desarrollado, lo cual nos permitiría ampliar el público al que va dirigido sin que la plataforma sea un impedimento.

A continuación se detallan una serie de posibles ampliaciones que fácilmente pueden ser implementadas para fomentar y potenciar el aprendizaje, que es nuestro objetivo principal:

- Lo primero sería poder almacenar los conjuntos de preguntas y palabras por temática, esto es, poder crear varios temas de estudio que el usuario pueda administrar como mejor le convenga y especializar su aprendizaje a su antojo.
- Otro posible enfoque podría ser el aprendizaje de idiomas, empezando por la gramática, podrían crearse conjuntos de palabras y preguntas orientadas al aprendizaje de cualquier idioma ya que, al no estar presente ninguna lengua en la aplicación, no existiría ningún impedimento que cualquier usuario, independientemente de la lengua que hable, pueda usarlo sin verse limitado.
- El añadir múltiples juegos a la aplicación podría ser otro ámbito bastante amplio de desarrollo. Se han propuesto dos de ellos, bastante conocidos y que permitan un uso más intuitivo de la aplicación, no obstante, el desarrollo de otro tipo de juegos más complejos y especializados puede hacer que el aprendizaje se vea potenciado por ello.
- Los sonidos pueden ser una herramienta interesante para ser usada en el aprendizaje. Al no encontrarse en los requisitos del sistema se ha decidido no añadir ningún sonido, sin embargo añadir sonidos en la aplicación, ya sea en el menú o en los juegos, puede hacer más atractivo su uso e incluso utilizar esos sonidos como un nuevo enfoque en el aprendizaje.
- Por otro lado, la opción de alojar la aplicación en un servidor puede ser bastante atractiva, ya que podrían comprobarse el avance de todos los jugadores entre ellos sin necesidad de tener que estar usando una misma máquina o estar comparando físicamente sus puntuaciones, así además se podría tutorizar el avance de dichos usuarios.

- Ampliando la opción anterior, podría desarrollarse además la posibilidad de sincronizar las partidas entre distintos jugadores así como poder jugar en tiempo real entre ellos y comprobar cómo es su avance.
- Otra posible ampliación sería la de desarrollar un sistema de contraseñas para cada usuario y que solo él pueda entrar en su perfil y jugar con su cuenta.

8.3. Objetivos alcanzados

La realización de este proyecto comenzó con una serie de objetivos a desarrollar, los cuales han ido alcanzándose a medida que transcurría el tiempo. Actualmente podemos afirmar sin lugar a dudas que estos objetivos han sido cumplidos con creces y que los resultados son los esperados. Los objetivos inicialmente propuestos fueron:

- Desarrollo de aplicación didáctica ambientada en el juego “Pasapalabra” para el aprendizaje de palabras relacionadas con la informática.
- Establecimiento de la lógica necesaria para administrar una base de datos de jugadores y de palabras a emplear.
- Registrar las puntuaciones de los jugadores para su posterior análisis.
- Carga de palabras desde un fichero a la base de datos.
- Administración del perfil de cada jugador.

Además de estos objetivos, se fueron alcanzando otros a medida que se desarrollaba el proyecto para hacerlo más atractivo y funcional:

- Adaptación de la aplicación a diferentes entornos como son el de escritorio o Android.
- Creación de un ranking de partidas de jugadores.
- Desarrollo de un segundo juego ambientado en el programa “¿Quién quiere ser millonario?”.
- Desarrollo de una interfaz gráfica para la administración de la base de datos de una forma intuitiva.

Todas estas ampliaciones y objetivos desarrollados hacen de esta aplicación una apuesta real por el aprendizaje lúdico que es lo que se pretende alcanzar.

8.4. Experiencia personal

En primer lugar he de decir que la experiencia vivida con la realización de este proyecto ha sido muy grata y constructiva, he podido retomar la programación después de algún tiempo y sobre todo seguir aprendiendo de Java como lenguaje de programación.

La utilización de LibGDX como framework en el que basar el proyecto ha sido quizás la parte más enriquecedora, poder aprender acerca de sus funcionalidades y de las herramientas que posee es una experiencia que hace ver una vez más que el mundo de la informática está en constante desarrollo y que tienes que estar renovándote continuamente.

Por otro lado la integración de SQLite en el proyecto, sobre todo en la parte Android, ha sido un reto a superar, ya que suponía la integración de varias herramientas y lenguajes que tenían que funcionar perfectamente para que el programa funcionase como se esperaba.

Sin duda es grato darse cuenta que con ganas de aprender y esfuerzo todo es posible, sin duda el reducido tiempo disponible para realizar el proyecto ha sido el mayor impedimento para poder ampliarlo y aprender mucho más de estas tecnologías, sin embargo el resultado final es bastante satisfactorio.

9. Bibliografía

- [1] Argo UML, <http://argouml.tigris.org/> (último acceso, 4/5/2016)
- [2] GanttProject, <https://www.ganttproject.biz/> (último acceso, 4/5/2016)
- [3] Github, <https://github.com/> (último acceso, 13/5/2016)
- [4] Gradle, <http://gradle.org/> (último acceso, 4/4/2016)
- [5] Java, <http://www.oracle.com/technetwork/java/index.html> (último acceso, 12/4/2016)
- [6] LibGDX, <https://libgdx.badlogicgames.com/> (último acceso, 4/3/2016)

- [7] Libgdx Cross-platform Game Development Cookbook, https://books.google.es/books?id=opQeBQAAQBAJ&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false (último acceso, 14/4/2016)
- [8] IntelliJ IDEA, <https://www.jetbrains.com/idea/> (último acceso, 8/1/2016)
- [9] Pencil Project, <http://pencil.evolus.vn/Features.html> (último acceso, 4/5/2016)
- [10] Photoshop, <http://www.adobe.com/es/products/photoshop.html> (último acceso, 21/7/2015)
- [11] Sgoliver.net, <http://www.sgoliver.net/blog/bases-de-datos-en-android-i-primeros-pasos/> (último acceso, 22/4/2016)
- [12] SQLite, <https://www.sqlite.org/> (último acceso, 29/4/2016)
- [13] Vector Magic, <http://vectormagic.com/home> (último acceso, 26/4/2016)

Anexos

Anexo 1. Manual del usuario

A1.1. Requisitos del sistema

- Sistemas operativos: Windows, Android.
- Resolución de pantalla: 1024 x 576 óptima.
- Java instalado.

A1.2. Instalación

- PC
Es tan sencillo como hacer doble clic sobre el ejecutable.
- Android
En Android se deberá ejecutar el archivo apk.

A1.3. Juego

A1.3.1. Controles

El control del juego es bastante intuitivo, si nos encontramos en el cliente para escritorio bastará con usar el clic izquierdo del ratón para pulsar botones o seleccionar elementos, mientras que haciendo clic en el botón derecho del ratón en cualquier parte de la pantalla nos desplazaremos al menú anterior. Por el contrario si nos encontramos en el cliente para Android, tendremos las opciones de puntear en la pantalla del dispositivo, lo que equivaldría al botón izquierdo del ratón, o pulsar el botón atrás que nos proporciona el sistema operativo. Como apunte adicional, en el cliente para escritorio tenemos la opción de cerrar la aplicación haciendo clic en la equis de la ventana, lo que equivale a pulsar el botón “home” que nos proporciona el sistema operativo Android.

A1.3.2. Menús

A1.3.1.1. Menú inicial:

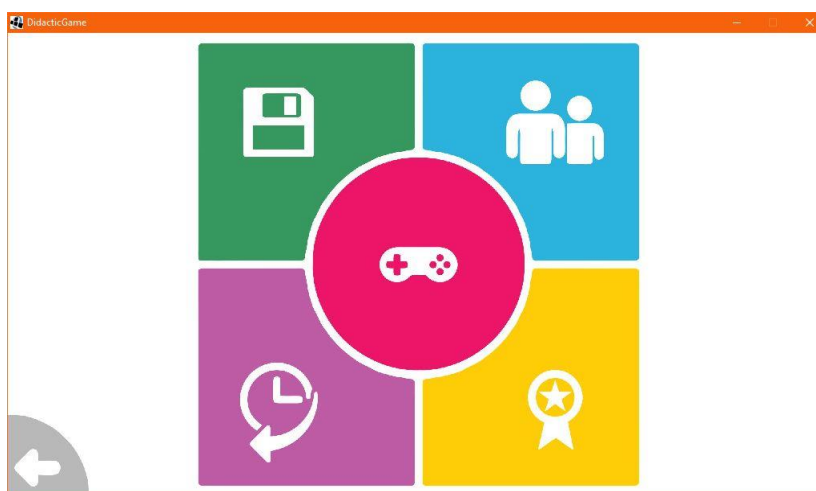


Figura 28: Menú Inicial

Esta es la primera pantalla que aparece al iniciarse el juego, Figura 28, en ella se encuentran los diferentes botones que nos permiten acceder a los distintos menús del juego, así como cerrar la aplicación si es el caso de haber ejecutado el juego en un entorno de escritorio.

A1.3.1.2. Menú Datos:



Figura 29: Menú Datos

Este menú, Figura 29, nos permite visualizar la lista de jugadores registrados en la base de datos.

Por otro lado, se pueden seleccionar las palabras que se mostrarán en el juego *Rosco* para poder visualizar y modificar la letra seleccionada para cada palabra así como la descripción de la palabra en cuestión.

También es posible seleccionar las preguntas del juego *QQSM* para visualizar y modificar la respuesta correcta y el contenido de cada respuesta.

Existen las opciones de añadir palabras o preguntas desde un fichero, Figura 30, que debe cumplir unos requisitos, eliminar la palabra o pregunta actualmente mostrada o añadir una palabra o una pregunta campo por campo, Figura 31:

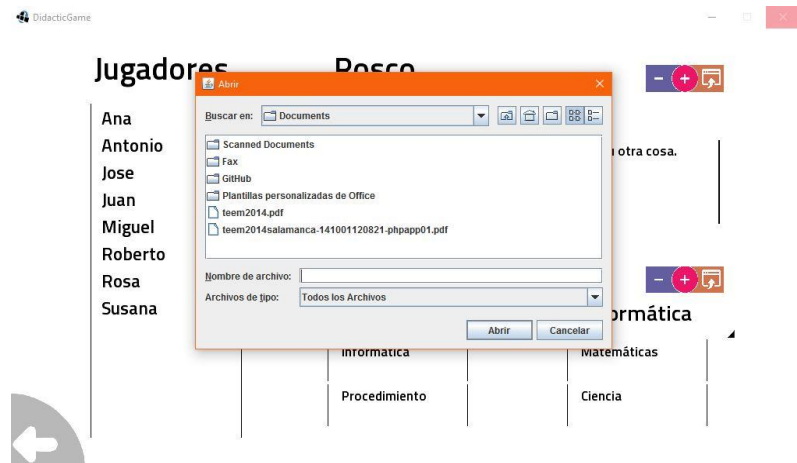


Figura 30: Menú Datos (Insertar Fichero)

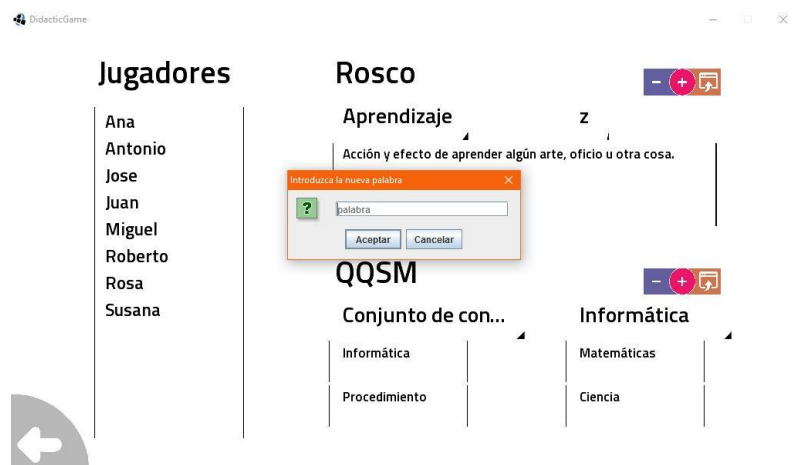


Figura 31: Menú Datos (Insertar Palabra)

A1.3.1.3. Menú Jugadores:

En este menú, Figura 32, se muestran individualmente los jugadores almacenados en la base de datos:



Figura 32: Menú Jugadores

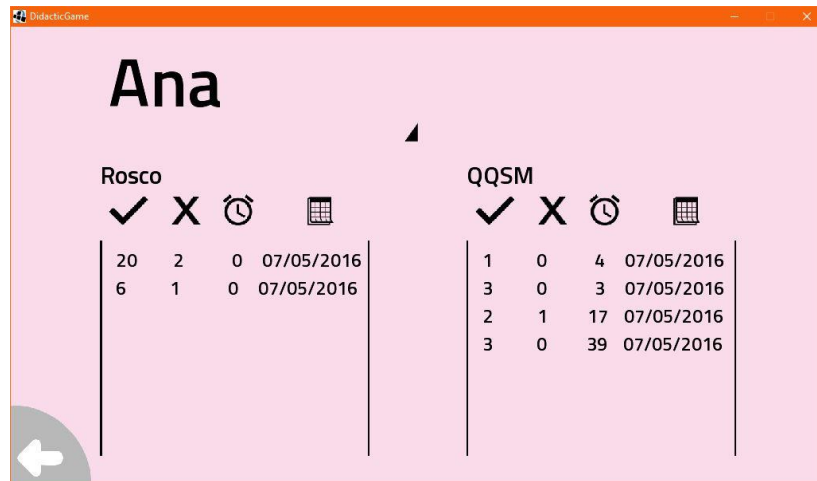
Es posible modificar el color y la edad de cada jugador, que por defecto es el blanco y 99 años, Figura 33:



Figura 33: Menú Jugadores (Seleccionar Color)

Pulsando el botón “+” se muestra un cuadro para añadir otro jugador introduciendo su nombre. Si se pulsa el botón “-” se elimina de la base de datos el jugador actualmente mostrado.

A1.3.1.4. Menú Puntuaciones:




Rosco			
20	2	0	07/05/2016
6	1	0	07/05/2016

QQSM			
1	0	4	07/05/2016
3	0	3	07/05/2016
2	1	17	07/05/2016
3	0	39	07/05/2016

Figura 34: Menú Puntuaciones

Este menú, Figura 34, nos permite seleccionar cualquier jugador de la base de datos para visualizar las puntuaciones que ha obtenido en cada juego.

A1.3.1.5. Menú Ranking:



Rosco			
1 posición -> Rosa:	25	1	1 07/05/2016
2 posición -> Miguel:	24	2	142 07/05/2016
3 posición -> Antonio:	23	3	142 07/05/2016
4 posición -> Susana:	23	3	142 07/05/2016
5 posición -> Jose:	22	2	142 07/05/2016

Figura 35: Menú Ranking

En él podemos visualizar las 10 mejores puntuaciones contenidas en la base de datos para cada juego, Figura 35. Mediante el selector de juego podemos visualizar las puntuaciones de cada juego en cuestión.

A1.3.1.6. Menú Jugar:



Figura 36: Menú Jugar

A este menú, Figura 36, se accede pulsando el menú central del menú inicial y en él se muestran todos los jugadores almacenados en la base de datos. Mediante el marcado de cada uno de estos jugadores se puede elegir cuál de ellos participará en el juego en cuestión.

Posteriormente al marcado de los jugadores se inicia el juego deseado pulsando uno de los dos botones que se muestran para iniciar el juego *Rosco* o el de *QQSM*.

A1.3.1.7. Juego Rosco:

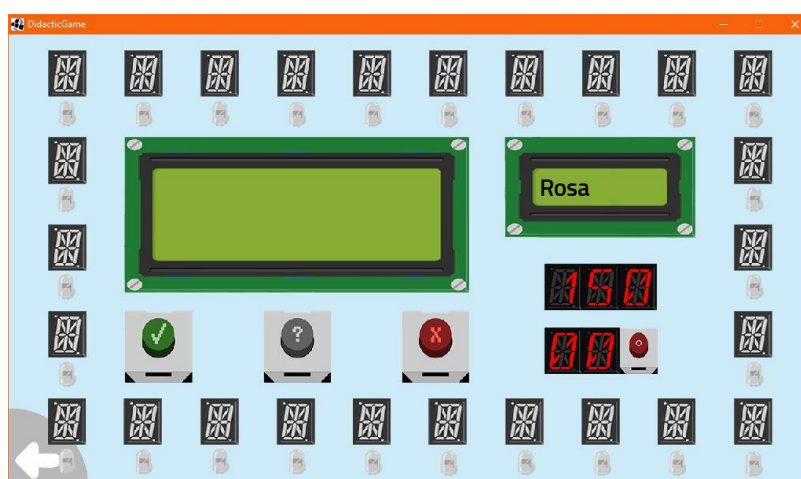


Figura 37: Juego Rosco (Estado Inicial)

Este es el estado inicial del juego *Rosco*, Figura 37, se ha elegido un estilo que nos recuerda a elementos usados en la electrónica, como son los 16 segmentos o las pantallas LCD. En él es posible modificar el tiempo, estableciendo un valor determinado en segundos o introduciendo el valor cero para anular el temporizador, o iniciar el juego pulsando el botón de inicio.



Figura 38: Juego Rosco (Jugando)

Esta es una instantánea cualquiera del juego *Rosco*, Figura 38, en la que se pueden apreciar las distintas letras acertadas, falladas o sin responder, la descripción de la palabra actual, el jugador actual, el tiempo restante y la puntuación actual.

A1.3.1.8. Juego QQSM:

Este es el estado inicial del juego *QQSM*, Figura 39. En él se puede apreciar el jugador que va a iniciar la partida así como el botón “GO” que permite iniciar el juego y la puntuación inicial que es 0.

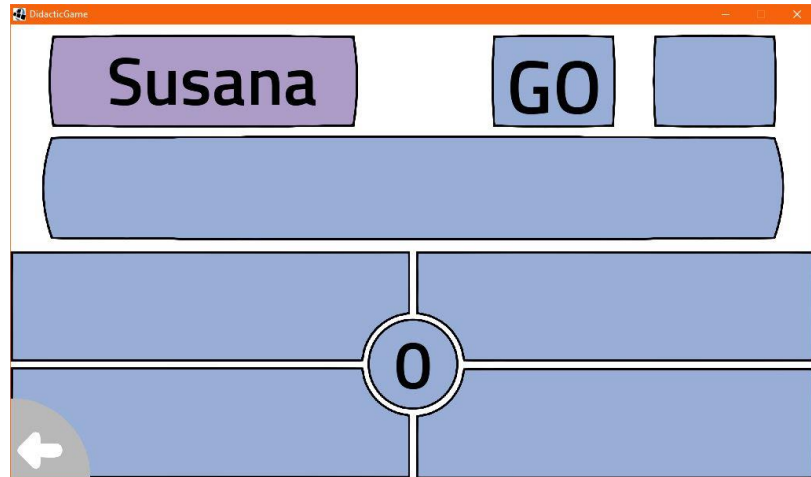


Figura 39: Juego QQSM (Estado Inicial)

Esta instantánea, Figura 40, nos permite visualizar el contador ascendente de tiempo que ha transcurrido desde el inicio de la partida, también se aprecia el comodín del 50% que aún no ha sido usado.

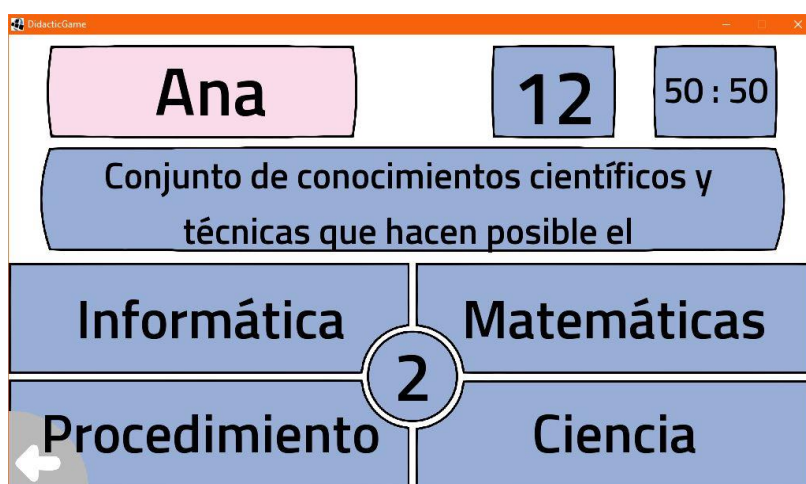


Figura 40: Juego QQSM (Jugando)

A1.3.1.9. Pantalla Clasificación:

Rosco				
	✓	X	🕒	📅
Juan:				
22	1	0	07/05/2016	
Miguel:				
24	2	142	07/05/2016	
Ana:				
6	1	0	07/05/2016	
Antonio:				
23	3	142	07/05/2016	
Rosa:				

Figura 41: Clasificación

Esta pantalla, Figura 41, se muestra tras cada sesión de cualquier juego. En ella se aprecian las puntuaciones de los jugadores que han participado en esta última sesión además del juego en cuestión.

A1.3.3. Introducción de datos

El programa presenta dos alternativas a la hora de introducir los datos referentes a las preguntas y palabras que luego se mostrarán en cada juego.

En primer lugar y la más intuitiva de las dos, es la opción de introducir los datos mediante la interfaz gráfica, un proceso más lento ya que hay que introducir los campos uno a uno, pero que puede ser una herramienta interesante cuando solo se pretende actualizar los datos existentes o añadir algunos de ellos de forma puntual.

En segundo lugar está la opción de introducir los datos mediante la lectura de ellos desde un fichero de texto. Esta alternativa es la más interesante cuando se pretende ampliar la base de datos de forma sustancial, ya que solo se ha de introducir los campos uno detrás de otro y no hay que estar tratando con la interfaz gráfica que puede resultar un trabajo más tedioso.

A1.3.2.1. Interfaz gráfica

Como hemos comentado, esta opción nos permite modificar los datos campo a campo.

En el apartado de *Rosco*, la palabra es un seleccionable de entre todas las palabras de la base de datos, la letra un seleccionable de entre todas las letras que forman la palabra en cuestión, y la descripción un campo editable, cuyo contenido modificado se actualizara al pulsar la tecla "intro".

En el apartado *QQSM*, la pregunta es un seleccionable de entre todas las preguntas de la base de datos, la respuesta correcta es un seleccionable de entre todas las respuestas a esa pregunta, y las respuestas son editables, que al igual que la descripción de la palabra, su contenido modificado no se almacenará hasta pulsar la tecla "intro".

En la siguiente imagen, Figura 42, se puede diferenciar cada campo por separado:

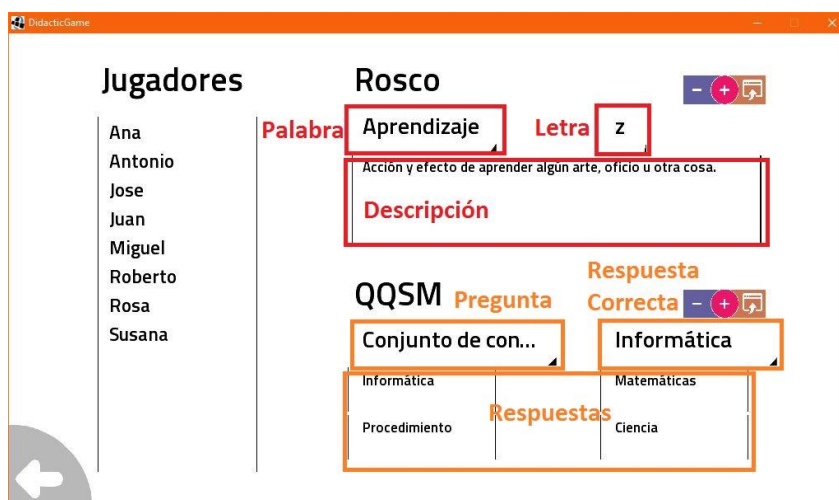


Figura 42: Menú Datos (Descripción)

A1.3.2.2. Fichero

La segunda alternativa es la introducción de todos los campos de las palabras o preguntas mediante un fichero de texto.

Las restricciones para la introducción de estos datos son las siguientes:

- Cada palabra o pregunta se debe introducir en un salto de línea distinto.
- Cada campo debe estar separado por el carácter “/”
- El orden de introducción de los campos es el siguiente:
 - a) Palabra: “palabra/ posición de la letra/ descripción”
Ejemplo: “comer/5/ Ingerir alimento”
 - b) Pregunta: “pregunta/ respuesta correcta/ primera respuesta/ segunda respuesta/ tercera respuesta/ cuarta respuesta”
Ejemplo: “¿Qué día es hoy?/ Domingo/ Lunes/ Martes/ Domingo/ Sabado”

Anexo 2. Manual del desarrollador

A continuación se realiza una breve introducción que facilita la comprensión del desarrollo que se ha llevado a cabo para el programa.

A2.1. Herramientas utilizadas

- SDK 1.8
- IntelliJ IDEA Community Edition 2016
- Gradle
- Maven
- Git para IntelliJ IDEA
- DB Navigator Plugin para IntelliJ IDEA
- SQLite
- LibGDX
- Librería FreeFontGenerator

A2.2. Conexión con la base de datos

Para realizar la conexión con la base de datos SQLite se deberá emplear un driver concreto dependiendo del sistema en el que nos encontremos.

```
try {
    if (Gdx.app.getType() == Application.ApplicationType.Android) {
        Class.forName("org.sqldroid.SQLDroidDriver");
        bdUrl = "jdbc:sqldroid:";
    } else if (Gdx.app.getType() == Application.ApplicationType.Desktop) {
        Class.forName("org.sqlite.JDBC");
        bdUrl = "jdbc:sqlite:";
    }
} catch (ClassNotFoundException clas) {Gdx.app.log("sql_clase",
clas.toString());}

jugadores_bd = DriverManager.getConnection(direccion(bdUrl, "jugadores"));
```

A2.3. Manejo de la base de datos

A continuación se muestran varios ejemplos para mostrar cómo se realiza el mantenimiento de la base de datos.

- Añadir palabras de un fichero:

```
public void anadir_fichero_palabras(File direccion) {
    Palabra palabra = new Palabra();
    FileHandle fichero = new FileHandle(direccion.getPath());

    String[] lineas = fichero.readString().split("\n");
    String[] campos;
    for (int i = 0; i < lineas.length; ++i) {
        campos = lineas[i].split("/");
        palabra.palabra = campos[0];
        palabra.posicion_letra = Integer.valueOf(campos[1].trim()) - 1;
        palabra.descripcion = campos[2];
        anadir_palabra(palabra);
    }
}
```

- Leer las preguntas de la base de datos:

```
public void leer_preguntas(ArrayList<Pregunta> preguntas) {
    try {
        ResultSet rs = qqsm_sentencia.executeQuery("select * from qqsm");

        for (int i = 0; rs.next(); ++i) {
            preguntas.add(i, new Pregunta());
            preguntas.get(i).respuestas = new String[4];
            preguntas.get(i).pregunta = rs.getString(1);

            preguntas.get(i).correcta = rs.getString(2);
            for (int j = 3; j < 7; ++j)
                preguntas.get(i).respuestas[j - 3] = rs.getString(j);
        }

        rs.close();
    } catch (SQLException sql) { Gdx.app.log("sql_leer_preguntas",
        sql.toString()); }
}
```

A2.4. Clases y objetos comunes

A continuación se muestra el diagrama de clases de la aplicación, Figura 43, se muestra simplificado para su mejor comprensión:

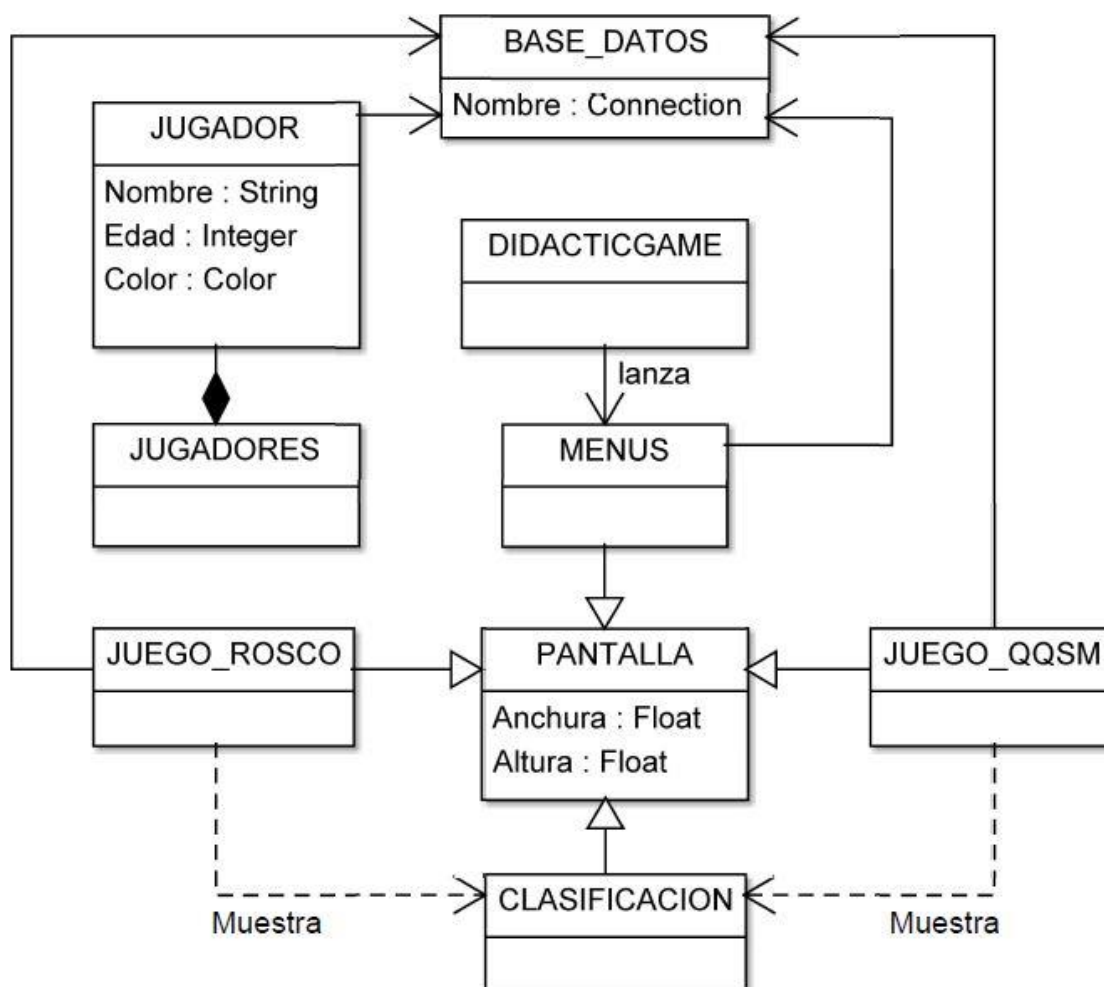


Figura 43: Diagrama de clases

Para la implementación de estas clases fue de gran ayuda la lectura del libro *Libgdx Cross-platform Game Development Cookbook* [7], desarrollado por antiguos alumnos de la Universidad de Cádiz.

- **OrthographicCamera**

Es un objeto obligatorio para nuestro juego. Determina la sección de la pantalla que vamos a ver ajustando el tamaño de la cámara. En nuestro caso el tamaño de la cámara siempre será el mismo que el de la pantalla, así que no es necesario modificarlo.

- **Texture**

Es el objeto básico de textura, con él cargaremos una imagen y lo utilizaremos para dibujarlo por pantalla.

- **SpriteBatch**

Esta clase se encarga de agrupar todas las texturas y formar una única imagen que es la que se mostrará por pantalla.

- InputAdapter

Esta clase, junto con otras que realizan la misma función en diferentes ámbitos, es la encargada de procesar el clic del ratón o el toque en la pantalla para realizar la acción pertinente.

- Class Screen

Esta clase es la encargada de mostrar todos los elementos en la pantalla y se caracteriza por los siguientes métodos:

- Render()

Este es el principal método empleado para mostrar por pantalla cada elemento. Se ejecuta continuamente con un espacio de tiempo determinado por las imágenes por segundo a los que se ejecute el programa.

- Show()

Método que se ejecuta cuando la pantalla en cuestión es la actualmente mostrada en pantalla, se usará por tanto para actualizar los elementos que muestra y que hayan podido ser modificados en otras partes del programa.

A2.5. Creación de textos

Para mostrar los textos en pantalla se ha empleado la librería FreeFontGenerator que nos permite, de una forma eficiente, generar textos en tiempo de ejecución a cualquier tamaño y de cualquier tipografía.

```
public static FreeTypeFontGenerator generador_texto = new
FreeTypeFontGenerator(Gdx.files.internal("fuentes/TitilliumWeb-
SemiBold.ttf"));
public static FreeTypeFontGenerator.FreeTypeFontParameter tamano_texto =
new FreeTypeFontGenerator.FreeTypeFontParameter();
```

A2.6. Redimensión de imágenes

La aplicación puede ser usada en diferentes entornos así como resoluciones de pantalla, para ello se ha de establecer las proporciones de cada elemento mostrado en pantalla de forma dinámica.

```
public float anchura_juego = Gdx.graphics.getWidth();
public float altura_juego = Gdx.graphics.getHeight();

public float proporcion_x (double valor) {
    return (float) (anchura_juego * valor);
}

public float proporcion_y (double valor) {
    return (float) (altura_juego * valor);
}
```