TRABAJOS FIN DE GRADO

Universidad de Jaén

Escuela Politécnica Superior de Jaén

Diseño e implementación de un juego de Sudoku

Autora: María de las Maravillas Luque Carmona

Grado: Ingeniería Informática

Director: Ángel Luis García Fernández
Departamento de Informática

Fecha: Septiembre 2024

(Página intencionalmente en blanco)

Universidad de Jaén

Departamento de Informática

Don Ángel Luis García Fernández, tutor del Trabajo Fin de Grado titulado:‘**Diseño e implementación de un juego de Sudoku**’, que presenta Doña María de las Maravillas Luque Carmona, otorga el visto bueno para su entrega y defensa en la Escuela Politécnica Superior de Jaén.

Jaén, Septiembre de 2024

La alumna:

Firmado por LUQUE CARMONA MARIA DE LAS MARAVILLAS - ***8117** el día 11/09/2024 con un certificado emitido por AC FNMT Usuarios

María de las Maravillas Luque Carmona

El tutor:

Firmado por GARCIA FERNANDEZ ANGEL LUIS - ***3879** el día 11/09/2024 con un certificado emitido por AC FNMT Usuarios

Ángel Luis García Fernández

(Página intencionalmente en blanco)

Agradecimientos

Escribiendo estas líneas doy por finalizada una etapa que me ha aportado muchísimo más de lo que esperaba, pero no hubiese sido posible sin ciertas personas a las que me gustaría agradecérselo concretamente.

En primer lugar a mi tutor Ángel Luis, gracias por tu gran labor como profesor, ojalá todos los profesores fueran como tú.

A José, mi pareja, gracias por siempre confiar en mi, por enseñarme, apoyarme y ayudarme, siempre has sabido lo que necesitaba mejor que yo misma. Lo que me llevo de esta carrera sin duda es haberte conocido.

A mis amigos, Laserna, Mario y Sky, jamás esperé encontrar a tres personas tan maravillosas hablando de colonialismo chino en África, me habéis enseñado tantísimo que no se ni por dónde empezar a agradecéroslo, gracias de corazón.

A toda mi familia, en especial a mis padres y abuelos, porque siempre han querido lo mejor para mí y me han enseñado lo que es trabajar por lo que quieres.

A mi hermano Alfonso, por enseñarme a usar un ordenador cuando aún no tenía edad para usarlo y sentar las bases de lo que es ahora mi vocación.

Y en especial a mi hermano Fernando, por ser mi eterno mentor en todos los ámbitos de la vida, por dar ejemplo de trabajo, esfuerzo, dedicación y constancia, por demostrarme lo que significa no rendirse, por ser el que nunca falla y por creer siempre que triunfaré, contigo como hermano ya he triunfado en esta vida y en mil más.

FICHA DEL TRABAJO FIN DE TÍTULO	
Titulación	Grado en Ingeniería Informática
Modalidad	Proyecto de Ingeniería
Especialidad (solo TFG)	Sistemas de Información
Mención (solo TFG)	Sin mención
Idioma	Español
Tipo	General
TFT en equipo	No
Autor/a	María de las Maravillas Luque Carmona
Fecha de asignación	16/04/2024
Descripción corta	El objetivo de este trabajo es realizar el diseño e implementación de una aplicación para generar tableros de Sudoku con dificultad variable y jugar en ellos. El usuario podrá configurar las características del tablero a generar, y la aplicación implementará las reglas del juego, permitiendo que el usuario juegue, e indicándole distintas métricas (tiempos, mejores resultados, promedios, etcétera).

NORMAS APLICADAS EN ESTE DOCUMENTO	
LOCALES	
TFT-UJA:2017	Normativa de Trabajos Fin de Grado, Fin de Máster y otros Trabajos Fin de Título de la Universidad de Jaén (Normativa marco UJA aprobada en Consejo de Gobierno)
TFT-EPSJ:2017	Normativa sobre Trabajos Fin de Grado y Fin de Máster en la Escuela Politécnica Superior de Jaén (Normativa EPSJ aprobada en Junta de Escuela)
TFT-EPSJ	Criterios de evaluación y normas de estilo para TFG y TFM de la Escuela Politécnica Superior de Jaén
NACIONALES E INTERNACIONALES	
ISO 2145:1978	Documentación - Numeración de divisiones y subdivisiones en documentos escritos
UNE 50132:1994	Traducción de la ISO 2145
APA 6ª edición	Estilo de referencias y citas de APA (American Psychological Association)

NORMAS UTILIZADAS COMO BASE O REFERENCIA	
NACIONALES	
UNE 157001:2014	Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico
UNE 157801:2007	Criterios generales para la elaboración de proyectos de sistemas de información
Estas normas se han utilizado como base o referencia para la inclusión de algunos contenidos y definiciones sobre elaboración de proyectos, entendiendo como proyecto la documentación consensuada entre una empresa y un cliente,que da lugar al perfeccionamiento de un contrato para la elaboración de una obra o la prestación de un servicio. Por consiguiente, no debe esperarse la aplicación de estas normas en cuanto a la completitud de los contenidos ni a la organización de los mismos.	

Contenido

Índice de ilustraciones

Índice de tablas

1 ESPECIFICACIÓN DEL TRABAJO

En este capítulo se presenta la especificación del trabajo, con una estructura y contenidos **inspirados** en los criterios y recomendaciones que establece la norma UNE 157801:2007 - "*Criterios Generales para la elaboración de proyectos de Sistemas de Información*".

A lo largo del documento se utilizarán términos y acrónimos cuya descripción aparecen en el apartado 6(Definiciones y abreviaturas).

1.1 Introducción

El Sudoku es un juego de lógica basado en un tablero de 81 casillas, distribuidas en 9 filas y 9 columnas, en el que hay que introducir números del 1 al 9. Inicialmente vienen dadas algunas pistas como en el ejemplo de la Ilustración 1.1:

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Ilustración 1.1: Ejemplo de Sudoku incompleto

Los números deberán introducirse siguiendo únicamente 3 reglas básicas del juego:

1. En cada fila deben aparecer los números del 1 al 9 sin repetidos.
2. En cada columna deben aparecer los números del 1 al 9 sin repetidos.
3. En cada sub-cuadrícula 3x3 deben aparecer los números del 1 al 9 sin repetidos.

Para resolverlo, deberá usarse la lógica para colocar los números faltantes, de manera que se cumplan las normas estipuladas previamente. El juego se dará por finalizado cuando el tablero esté completo con los 81 números necesarios y no se viole ninguna regla, podemos verlo en la Ilustración 1.2.

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Ilustración 1.2: Ejemplo de Sudoku resuelto

1.2 Objetivos del trabajo

Este trabajo se ha desarrollado teniendo en cuenta el cumplimiento de los siguientes objetivos:

1. Demostración del grado de madurez en la elaboración de análisis y diseño de aplicaciones de escritorio.
2. Estudiar las tecnologías que existen en la actualidad y selección de la más idónea para la resolución del problema planteado.
3. Creación de un producto software completo y correcto, listo para la utilización por usuarios finales.
4. Diseño de una interfaz gráfica amigable, comprensible y visualmente atractiva, basándose en los principios del minimalismo.
5. Desarrollo de un producto software que cumpla con los requisitos de accesibilidad para su correcta utilización.

1.3 Antecedentes y estado del arte

En el siglo XVIII, Leonhard Euler desarrolló un sistema probabilístico para representar una serie de números sin repeticiones, llamado *cuadrado latino* (Ilustración 1.3). A raíz de este tipo de sistema se podían representar distintos dígitos en forma de matriz, de manera que no coincidiesen en fila o columna con otro número igual.

3	4	8
6	7	2
5	9	1

Ilustración 1.3: Ejemplo de cuadrado latino

Fue en el año 1970 cuando el sistema probabilístico pasó a ser considerado pasatiempo por su publicación en una revista de puzzles y enigmas matemáticos, aunque no llegó a tener una fama notoria.

Años después, en 1984, volvió a ser publicado, esta vez en *Nikoli*, una revista de enigmas matemáticos, donde su presidente Maki Kaji le dio por primera vez el nombre de Sudoku, basándose en las palabras sū = número y doku = solo, que le daban explicación a la forma en la que se resolvía, puesto que cada número debe estar solo y de manera única en su recuadro, fila y columna adyacente. Esta vez sí alcanzó una gran popularidad con la publicación.

La primera versión por ordenador apareció en 1985, para el *Commodore 64* (Ilustración 1.4), por obra de *Loadstar Softdisk Publishing*.

Ilustración 1.4: Consola Commodore 64 (Jauzat, L., & Jauzat, L., 2023)

Con la llegada a Europa en los 2000, el juego empezó a aparecer en televisión y en múltiples revistas como *The Times*, llegando así a ser un pasatiempo conocido mundialmente poco después.

Actualmente existen infinidad de aplicaciones para dispositivos móviles y juegos de ordenador con los cuales jugar a este tipo de acertijos matemáticos, siendo una de las versiones más famosas de escritorio la creada hace justamente 10 años, en 2014, por la empresa *Xbox Game Studios*, en colaboración con *Microsoft* (Ilustración 1.5).

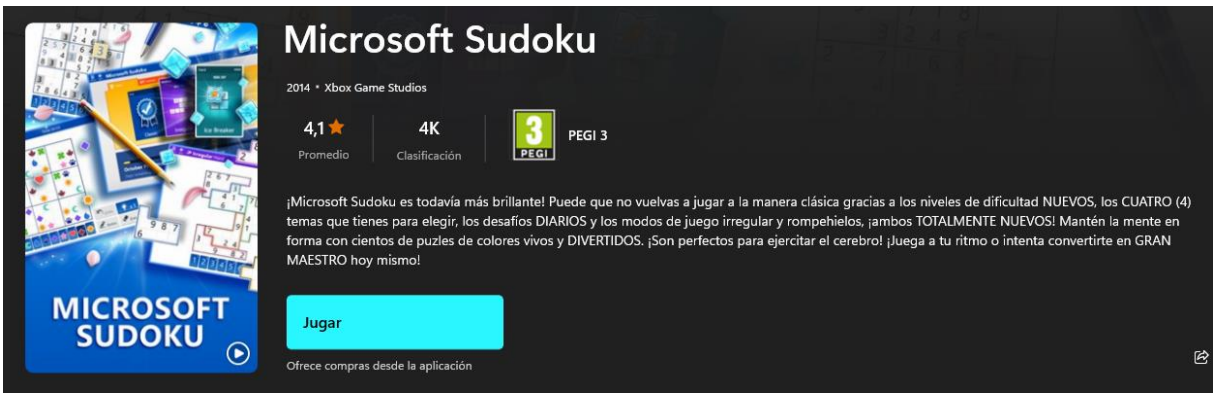


Ilustración 1.5: Juego desarrollado por Xbox Game Studios en Microsoft Store

Junto al auge actual del desarrollo de aplicaciones móviles y de escritorio, se plantea una problemática muy acusada, ya que en la mayoría de ocasiones se sacrifica jugabilidad y accesibilidad a cambio de la generación de ingresos, normalmente mediante anuncios tipo pop-up en la aplicación o programa. De aquí nace la necesidad de crear un juego de software libre que permita a todo tipo de usuarios la descarga y jugabilidad, de manera que sea totalmente accesible, gratuito y mejorable de forma comunitaria mediante aportaciones y sugerencias.

1.4 Descripción de la situación de partida

A continuación se analizará la situación de partida del proyecto, haciendo una investigación sobre productos software existentes en el mercado y se comentarán las deficiencias encontradas.

1.4.1 Descripción del entorno actual

Para comprender el entorno existente, comprobaremos los resultados al buscar un juego de resolución de Sudokus en dos plataformas que permiten la descarga de juegos de ordenador:

1. *Steam* (Valve Corporation, 2024): la principal plataforma de distribución digital de videojuegos (Ydoate, 2021). La Ilustración 1.6 muestra el resultado de esta búsqueda.

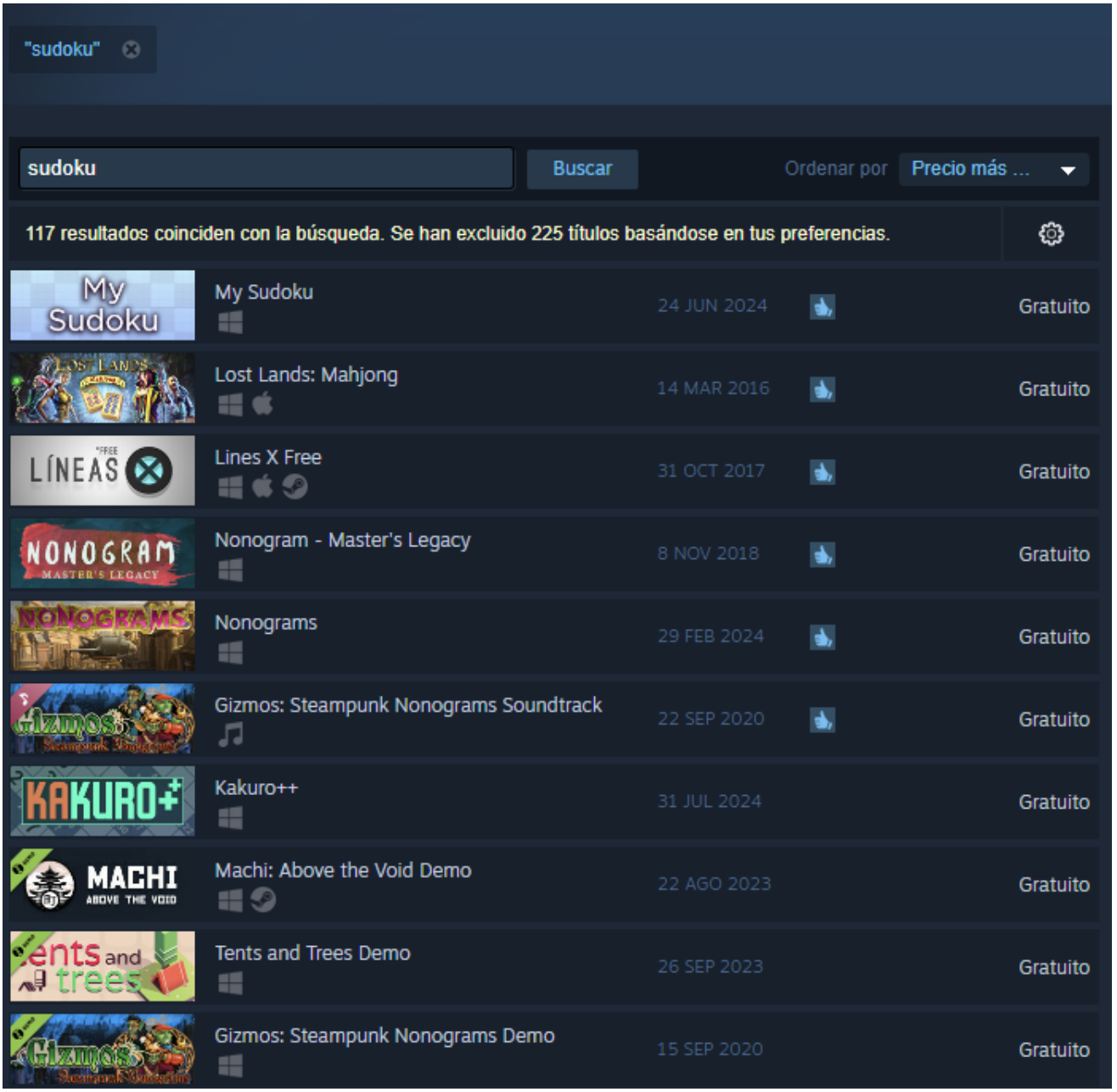


Ilustración 1.6: Búsqueda de juegos de Sudoku en Steam

2. *Microsoft Store* (Microsoft Apps, 2024)*:* tomada como referencia para usuarios de Windows, (Ilustración 1.7).

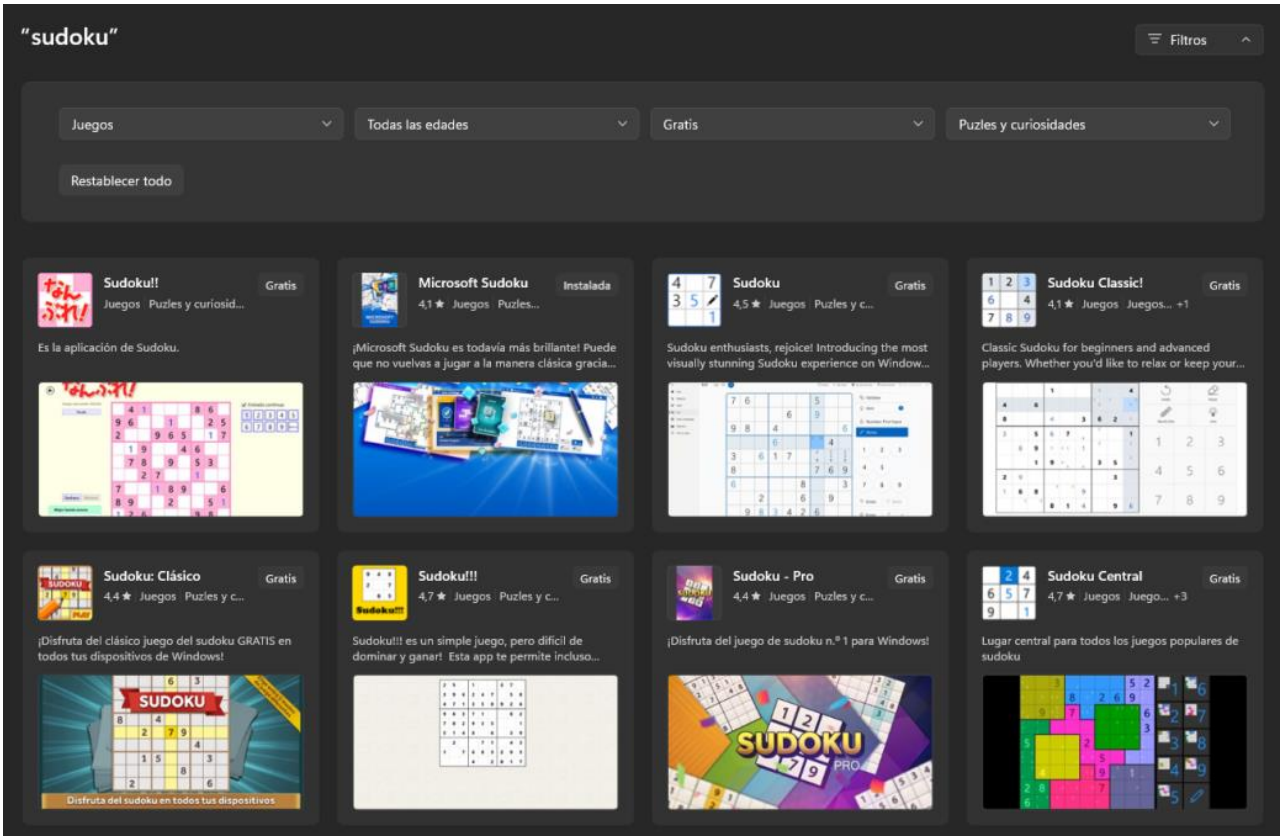


Ilustración 1.7: Búsqueda de juegos de Sudoku en Microsoft Store

En el primer caso, partimos de la base de que únicamente existen 10 juegos de Sudoku que se puedan descargar de manera gratuita, lo cual reduce significativamente la variedad desde la que escoger. Además, como se puede apreciar en la Ilustración 1.6, si lo que buscamos es un juego que únicamente tenga Sudokus y no tenga otros añadidos, lo cual no es pedir demasiado, se queda en 2 el número de posibilidades.

En la segunda búsqueda, por contraposición, podemos ver en la Ilustración 1.7, que hay cientos de juegos disponibles para el usuario.

1.4.2 Resumen de las deficiencias y carencias identificadas

Se pueden destacar principalmente dos deficiencias encontradas tras el análisis y búsqueda de este tipo de juegos:

1. **Publicidad**: si analizamos los juegos disponibles (gratuitos), podemos encontrar de manera rápida uno o varios anuncios que arruinan la experiencia inmersiva que debería de aportar cualquier tipo de juego (Ilustración 1.8).

Ilustración 1.8: Publicidad en el menú inicial de Microsoft Sudoku

Por ejemplo, tras abrir Microsoft Sudoku, en menos de 2 segundos ya ha aparecido una ventana emergente que abre un navegador incluyendo publicidad, como se puede ver en la Ilustración 1.9, esta vez de manera totalmente invasiva. De hecho, esta segunda publicidad carece totalmente de sentido, porque ha quedado demostrado que ni siquiera se emplean métodos de personalización de anuncios, ya que ha aparecido habiendo iniciado sesión con una cuenta de Xbox, que también tiene instalada la aplicación que justamente se recomienda. Todo esto se resume en una experiencia nefasta para el usuario que simplemente desea jugar.

Ilustración 1.9: Publicidad invasiva de Microsoft Sudoku

2. **Privacidad**: Muchos de estos juegos no cuentan con políticas de privacidad claras o transparentes para el usuario, de manera que podrían recopilar datos personales, como la ubicación, identificadores de dispositivos, historial de búsqueda o información del dispositivo sin autorización explícita del usuario. Dicha información podría ser vendida a terceros como empresas de publicidad sin que el usuario tenga conocimiento o control sobre ello.

3. **Accesibilidad**: Si el usuario decide optar por juegos más *indie* para intentar evitar el abuso de publicidad por parte de empresas que buscan un gran beneficio económico, en caso de tener cualquier tipo de discapacidad visual, tendría que descartar la mayoría de estos juegos, que no se han hecho accesibles para todas las personas (Ilustración 1.11 e Ilustración 1.12). El objetivo de la accesibilidad es que todas las interfaces sean perceptibles, operables y comprensibles para las personas, independientemente de si tienen algún tipo de discapacidad. La importancia de esta característica se acentúa aún más si hablamos de videojuegos, ya que un juego que no se entiende, en este caso debido a su interfaz gráfica (Ilustración 1.10), provoca un sentimiento de frustración que afecta a la experiencia de usuario, e incluso a la autoestima de la propia persona.

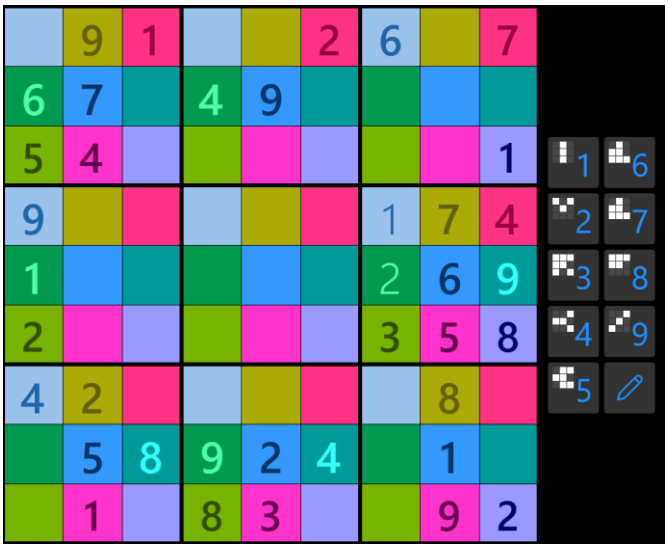


Ilustración 1.10: Captura de juego de Sudoku Central (Fasung Design)

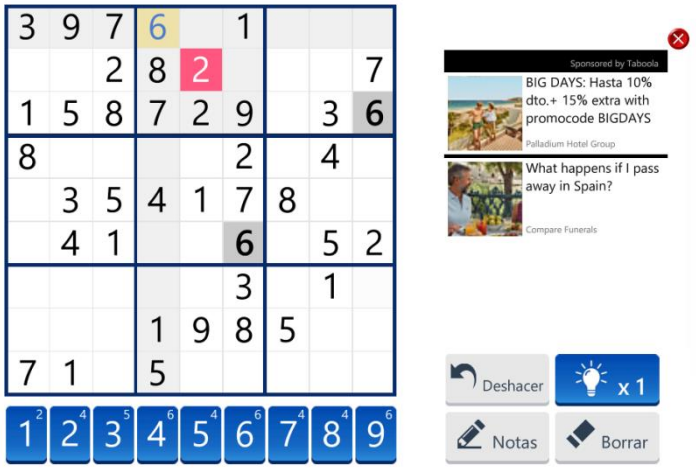


Ilustración 1.11: Captura de juego de Microsoft Sudoku

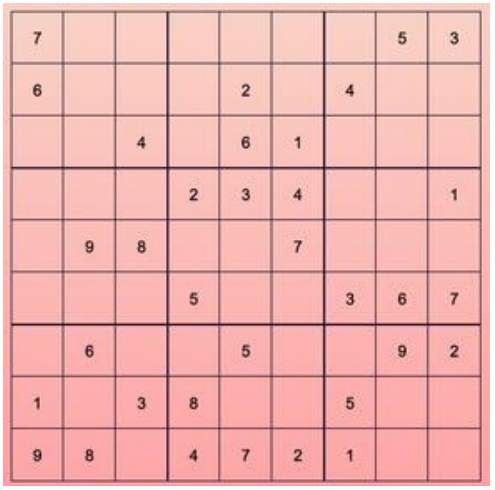


Ilustración 1.12: Captura de juego de MySudoku (Splutterworth)

Si nos centramos en los principios recogidos en los principios de accesibilidad proporcionados por (Web Accessibility Initiative, 2019), tomando como ejemplo únicamente dos de ellos:

1. (Web Accessibility Initiative, 2019): Se indica que el color no debe ser usado como único medio de transmisión de información, y que debe existir esa información proporcionada en formato texto. Sin embargo, si se observan las capturas de la Ilustración 1.10 y de la Ilustración 1.11, podemos ver que el color es el único medio de transmisión de información, que además es vital para el desarrollo del juego ya que indica los errores. Se utiliza el color rojo, en caso de Microsoft Sudoku, y una mezcla de colores en Sudoku Central, y en ningún caso se permite obtener esa información en formato textual.

2. (Web Accessibility Initiative, 2019): Se explica que el contraste entre las imágenes y el texto presentado debe tener un ratio de al menos, 4.5:1, pero si observamos la Ilustración 1.10 o la Ilustración 1.12, no hace falta fijarse demasiado para comprender que el contraste de los colores y el texto es demasiado bajo.

1.5 Requisitos iniciales

A partir de la especificación del problema a resolver y de las deficiencias encontradas en el mercado actual, se han establecido los siguientes requisitos iniciales:

1. El programa debe ofrecer al usuario un juego sin publicidad de ningún tipo.
2. El programa debe ser visualmente minimalista, sin abuso del color.
3. El programa debe tener un tutorial de uso para los jugadores nuevos que no entiendan cómo funcionan los menús y ajustes disponibles.
4. El programa debe tener un selector de dificultad para que el usuario elija según sus necesidades y gustos.
5. El programa debe ser de código abierto, debe estar ubicado en un repositorio público para que cualquiera pueda descargarlo, modificarlo y hacer con él lo que desee, exceptuando cualquier actividad lucrativa para todo aquel que no sea el creador, en este caso creadora.
6. El programa debe tener una parte colaborativa en el repositorio en el que se encuentre, permitiendo así la interacción de los usuarios con experiencia en programación para sugerir mejoras, proponer actualizaciones y evitar errores.
7. El programa debe ser accesible para los usuarios que tengan discapacidades visuales, permitiendo el uso de texto en vez del color para transmitir la información.
8. El programa debería tener la posibilidad de adaptar algún nivel de dificultad, ya sea para aumentarlo o disminuirlo.

1.6 Alcance

Los distintos entregables que se incluirán en este proyecto son los mencionados a continuación:

- **Memoria**: en la que encontrará descrito todo el proceso de análisis, diseño e implementación del sistema de manera detallada, incluyendo la justificación de tecnologías usadas, metodologías empleadas y un manual de usuario para su fácil uso e instalación.
- **Código fuente**: todo el código que se desarrolle se almacenará en un repositorio público con el cumplimiento del correspondiente requisito mencionado anteriormente, en este caso se ha elegido GitHub para esta labor (GitHub, 2024).
- **Vídeo demostración:** un pequeño vídeo del programa desarrollado en el que se vea cómo funciona, distinguiendo entre modo accesible y modo por defecto.

El objetivo final de este proyecto es el de desarrollar un producto software completo pero en fase de prototipo, debido a la restricción de personal y horas empleadas para su realización, así que quedará fuera del alcance el posterior mantenimiento e implementación de mejoras.

1.7 Hipótesis y restricciones

El TFG se define como una asignatura de 12 créditos, lo que supone que la duración total del proyecto será de 300 horas, incluyendo todas las etapas del ciclo de vida, con la excepción del mantenimiento. Por consiguiente, la principal restricción aplicable es la limitación de la duración del trabajo.

La siguiente restricción aplicable será el número de personas participantes en este proyecto, que constará de únicamente una persona. A causa de esto, la calidad de la aplicación de todos los ámbitos relacionados con campos ajenos a Ingeniería Informática quedará reducida a una etapa inicial, como puede ser el diseño y la internacionalización del programa, entre otros.

1.8 Estudio de alternativas y viabilidad

La principal decisión que había que tomar era el sistema operativo para el que se quería desarrollar el juego, puesto que debido a las restricciones indicadas en el apartado de Hipótesis y restricciones, no se disponía de tiempo para desarrollar un juego multiplataforma. Se planteaban entre las alternativas los sistemas operativos Windows, MacOS y Linux, ya que son los más usados.

Sistema Operativo	Cuota de Mercado
Windows	63.13%
OS X	17.78%
Desconocido	12.5%
Chrome OS	3.74%
Linux	2.83%
FreeBSD	0.01%

Ilustración 1.13: Sistemas operativos más usados en el mundo (sistemasoperativos.info, 2023)

Según los datos presentados en la Ilustración 1.13, el sistema operativo a elegir debería ser Windows, puesto que es el más usado con una cuota de mercado del 63.13%. Además entran a colación otros factores determinantes como la imposibilidad de usar MacOS, puesto que no se dispone de ningún dispositivo propio que use ese sistema operativo. Por estos motivos se escogió **Windows**.

La segunda decisión a tomar es la del lenguaje de programación que se usará para desarrollar el programa. Según (Dev, 2023), Python se posiciona como el segundo lenguaje de programación más demandado en 2023, precedido únicamente por JavaScript. A pesar de esto Python, ofrece una mayor versatilidad de desarrollo, además de que ya se posee experiencia previa en este lenguaje, lo que simplificaría la curva de aprendizaje y desarrollo. Además de ser uno de los lenguajes más demandados, también se posiciona primero en (Ramirez, 2024) y en (Alonso, 2024), por lo que **Python** ha acabado siendo el elegido finalmente.

1.9 Descripción de la solución propuesta

La solución propuesta consiste en un programa ejecutable, que no requiere de instalación para facilitar su sencillez de uso. Al ejecutar el programa aparecerá un menú inicial en el que el usuario podrá modificar ajustes de dificultad y accesibilidad, contará también con un tutorial que le hará un *tour* por el programa, tendrá estadísticas de partidas anteriores, contendrá información acerca del autor y el proyecto en el que está englobado el juego y por último, el usuario tendrá un botón para empezar a jugar una partida con el nivel de dificultad configurado.

Respecto a la estructura interna, se ha utilizado programación orientada a objetos, dado que es la opción que más se adapta al nivel de modularización deseado para una mejor detección de errores.

En cuanto a diseño de la interfaz de usuario, será un diseño minimalista, simple y sin demasiados colores ni imágenes, que permitirá al usuario alternar entre el modo accesible y el modo por defecto, modificando algunas indicaciones de color durante la partida para que se entiendan de manera adecuada, junto a una leyenda que aparecerá en pantalla.

1.10 Tecnologías utilizadas

Entre las distintas tecnologías que se han usado para el desarrollo de este proyecto, se destacan las siguientes:

- **Entorno de desarrollo:** como entorno se ha elegido *Visual Studio Code* (Microsoft, 2024)**,** puesto que permite un desarrollo cómodo, tiene disponible infinidad de plugins y funciona realmente bien con Git (Git, 2024) como control de versiones.
- **Python 3.9** (Ilustración 1.14)**:** será el lenguaje de programación que se usará para desarrollar el programa en su totalidad. Elegido por su versatilidad y simpleza, este lenguaje permite un desarrollo cómodo y escalable. Una característica fundamental es que se trata de un lenguaje de código abierto, lo cual encaja a la perfección con la temática del proyecto, que se basa en promover el software libre y de código abierto. Además de ser totalmente legible, es un lenguaje interpretable y no compilado, lo que simplifica la configuración y puesta en marcha, también soporta programación orientada a objetos, lo cual es muy útil para la modularización del código y su debida estructuración. Por último, la infinidad de documentación y tutoriales disponibles en Internet sobre este lenguaje hacen que el proceso de desarrollo se simplifique, evitando así multitud de errores posibles, permitiendo la optimización a la misma vez del código.

Ilustración 1.14: Logo de Python (Python Software Foundation, 2024)

- **Pygame** (Ilustración 1.15)**:** esta biblioteca es realmente popular en Python para el desarrollo de juegos en dos dimensiones (2D). A diferencia de otras bibliotecas, ofrece ventajas específicas para crear un juego de Sudoku. Se trata de una biblioteca ideal para principiantes, ya que tiene una curva de aprendizaje relativamente baja y permite enfocarse en el desarrollo del juego sin necesidad de configuración técnica excesiva. Es software libre y gratuito, así que está alineado con el propósito de este trabajo y también dispone de una documentación amplia y bien organizada, junto a una comunidad muy activa. En cuanto a desarrollo, permite personalizar totalmente todo lo que se muestra en pantalla, como los números, el tablero o los botones. También permite actualizar fácilmente el estado de la pantalla, lo cual es esencial para un juego de Sudoku en el que se introducirán números constantemente y se necesita una tasa de refresco de pantalla relativamente alta para dar una visión sin cortes del juego. Otra ventaja añadida es que permite gestionar la entrada del teclado y ratón de una manera muy cómoda. Esto es esencial debido a que los números de las celdas vacías del Sudoku se introducirán mediante teclado. Por último, permite una separación cómoda de la lógica del juego y la representación visual.

Ilustración 1.15: Logo de Pygame (Pygame, 2024)

- **Numpy** (Ilustración 1.16)**:** al igual que las tecnologías anteriores, esta biblioteca también es de código abierto y cuenta con infinidad de documentación accesible en Internet. Esta biblioteca es muy útil, ya que permite la gestión de vectores multidimensionales de una manera realmente cómoda y eficiente, como es el caso de las matrices en las que se ha basado el tablero de Sudoku, puesto que se representará en una matriz de 9 filas y 9 columnas. Esta biblioteca tiene implementadas diversas operaciones matemáticas y lógicas que harán de la gestión de matrices en el juego una tarea eficiente computacionalmente y mucho más cómoda que con el uso de Python base.

Ilustración 1.16: Logo de Numpy (NumPy team, 2024)

- **CSV:** Para la gestión de archivos de texto externos a la lógica principal del programa, como puede ser el histórico de puntuación o la gestión y almacenamiento de la configuración de usuario, se ha usado la biblioteca base de Python, que permite gestionar ficheros CSV de forma directa en el código (Python, 2024)**.**
- **PyInstaller** (Ilustración 1.17)**:** Para la exportación del proyecto a un archivo ejecutable de Windows se ha usado esta biblioteca externa de Python, que permite agrupar todos los archivos, carpetas, imágenes e iconos en un mismo ejecutable de una manera cómoda para evitar errores.

Ilustración 1.17: Logo de PyInstaller (PyInstaller, 2024)

- **Canva** (Ilustración 1.18)**:** Plataforma de diseño gráfico que proporciona diversas herramientas muy útiles, usada para la realización de diseños de logos, edición de imágenes de fondo y creación de imágenes vectorizadas para insertar en botones.

Ilustración 1.18: Logo de Canva (Canva, 2024)

- **Git** (Ilustración 1.19) **y Github** (Ilustración 1.20): Software de control de versiones basado en la eficiencia, confiabilidad y compatibilidad. Es un proyecto de código abierto y gratuito. Git se ha usado junto a GitHub, que es un servicio basado en la nube que tiene implementado Git para almacenar las versiones del proyecto a lo largo de las iteraciones y distintas fases del desarrollo.

Ilustración 1.19: Logo de Git (Git, 2024)

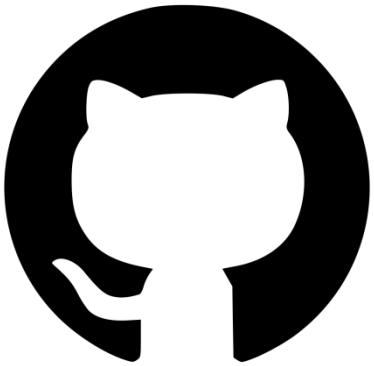


Ilustración 1.20: Logo de GitHub (GitHub, 2024)

1.11 Metodología de desarrollo de software

En cuanto a la metodología empleada, se han valorado las distintas opciones, como la metodología clásica incremental, la metodología en cascada y la metodología ágil, entre otras. Finalmente se ha decidido optar por una metodología clásica incremental, la cual es una modificación de la metodología en cascada (Ilustración 1.21), caracterizada por tener un enfoque lineal y secuencial en el que se tiene una serie de etapas muy definidas. Las fases de análisis, diseño, desarrollo, pruebas, implementación y mantenimiento van encadenadas una tras otra.

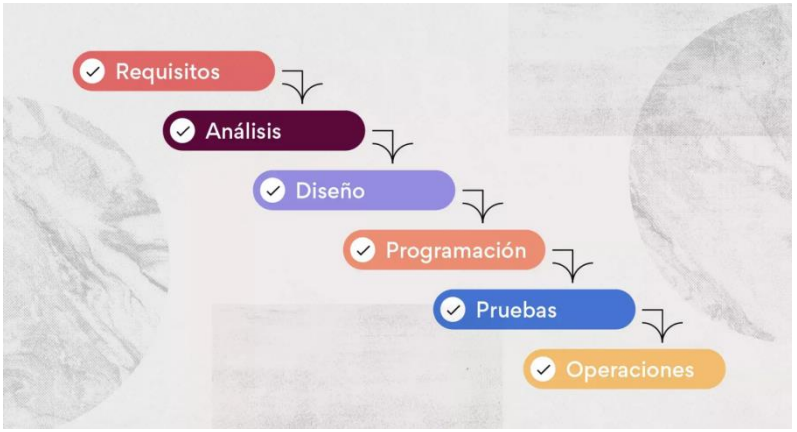


Ilustración 1.21: Esquema de la metodología en cascada (Sarah Laoyan, 2024)

Aunque este tipo de metodología es realmente útil, no se adapta del todo bien a los requisitos que tiene el desarrollo de este proyecto, siendo indispensable poder adaptar y añadir funcionalidades en mitad de cualquier etapa, a la par que cambiar los requisitos conforme se van realizando pruebas con usuarios finales.

A diferencia de la metodología en cascada, en la metodología incremental (Ilustración 1.22) se van desarrollando las versiones del producto software de manera que

en cada iteración se tiene un producto funcional entregable, al que se le van añadiendo mejoras y diversas funcionalidades a lo largo de todo el ciclo.

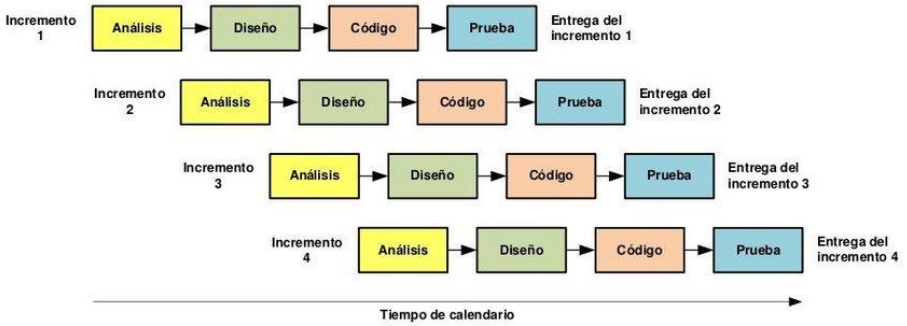


Ilustración 1.22: Esquema de la metodología incremental (ResearchGate, 2016)

Se ha elegido esta metodología por las siguientes razones:

- El proyecto tiene un alcance y un tamaño bien definido desde el inicio.
- El desarrollo del proyecto permite ser dividido en incrementos funcionales de manera sencilla.
- Esta metodología permite insertar nuevos requisitos en mitad del ciclo de vida del proyecto e implementar diversas mejoras.
- La flexibilidad es una característica fundamental en este proyecto, ya que se desarrollará en continua comunicación con varios usuarios finales, que aportarán su visión del juego, necesidades, problemas y mejoras, por lo que habrá que poder realizar modificaciones de manera cómoda.
- Permite asumir un menor riesgo en la etapa de desarrollo, ya que si se produce cualquier aparición de problemas se pueden identificar y solventar en la siguiente iteración.

1.12 Estimación del tamaño y esfuerzo

Ya que el presente proyecto es un TFG, no existen restricciones de tipo económico, sino de tipo temporal (un número aproximado de horas). Por consiguiente, los cálculos de tamaño del proyecto están supeditados al tiempo disponible. Debido al objetivo del proyecto, que en gran medida es hacer un producto de software libre y de código abierto, solo se usarán herramientas gratuitas, así que las opciones serán limitadas. En cuanto al esfuerzo, se dispone de tan un solo efectivo (la persona autora del trabajo).

Para estimar el esfuerzo del desarrollo del producto software, se ha optado por **COCOMO básico de tipo orgánico,** puesto que el proyecto es pequeño a nivel de líneas de código y no aporta gran innovación a nivel técnico. Para esto es necesario el uso de las siguientes ecuaciones:

$$E = a \cdot KLDC^{b}$$

1.1: Estimación del esfuerzo (personas/mes)

$$TD = c \cdot E^{d}$$

1.2: Tiempo de Desarrollo (meses)

Donde:

- Los coeficientes a, b, c y d vienen dados inicialmente por el método. Dependiendo del tipo de COCOMO que más se ajusta al proyecto, dichos coeficientes se incluyen a continuación:

Tipo de proyecto	Parámetro a	Parámetro b	Parámetro c	Parámetro d
Orgánico	2.4	1.05	2.5	0.38
Medio	3.0	1.12	2.5	0.35
Empotrado	1.6	1.2	2.5	0.32

Tabla 1.3: Parámetros del modelo COCOMO

- KLDC es el número de líneas de código, expresado en miles.

El tamaño estimado del producto software es de 2.000 líneas de código, así que conociendo este dato, podemos obtener que:

$E = 2.4 \cdot 2^{1.05} = 4.96$ personas-mes ≈ 5 personas-mes

$TD = 2.5 \cdot 5^{0.38} = 4.60$ meses

El tiempo de desarrollo estimado será finalmente de 4.6 meses, sin incluir la redacción de la memoria y el proceso previo de investigación.

1.13 Planificación temporal

La planificación temporal abarcará desde el inicio del proyecto hasta la finalización de la redacción de esta memoria, para ello se utilizará la Tabla 1.4 en la que aparecen todas las tareas a realizar, asignándolas a la persona encargada de llevarlas a cabo:

Tarea a realizar	Descripción	Tiempo estimado (semanas)	Persona encargada
Análisis inicial	Investigación y recogida de necesidades para el proyecto.	1	Jefe de proyecto
Desarrollo de una interfaz básica	Desarrollo de una ventana básica con un par de botones para empezar a probar utilidades de Pygame y verificar la funcionalidad.	1	Ingeniero
Desarrollo de un menú inicial	Desarrollo de un menú que aparezca al ejecutar, que permita al usuario elegir lo que desea hacer.	1	Diseñador
Desarrollo de botones	Diseño y desarrollo de botones, estilo, funcionalidad, colores iniciales…	1	Diseñador
Creación de la ventana de juego principal	Desarrollo de la ventana principal de juego, construcción visual del tablero de Sudoku, pruebas con tablero vacío/lleno y distribución en pantalla de botones durante el juego.	1	Diseñador
Creación de funcionalidades durante el juego	Desarrollo y verificación de funcionalidades durante el juego, creación de temporizador, introducción de números mediante teclado, borrado de tablero y comprobación de solución con un Sudoku dado.	2	Programador senior
Mejora visual del tablero	Mejoras visuales de colores durante el juego, cambio en botones, añadir sombras y verificación de la posibilidad de introducir un número en una casilla.	1	Diseñador
Lógica de gestión de Sudokus	Gestión de Sudokus en código, añadir número, quitar número, comprobación de reglas de Sudoku y verificar con la solución.	1	Ingeniero

Creación de dificultad fácil	Creación de Sudokus de dificultad fácil de manera eficiente y medianamente rápida.	1	Programador senior
Creación de dificultad media	Creación de Sudokus de dificultad media de manera eficiente y medianamente rápida.	1	Programador senior
Selector de dificultad	Permitir al usuario seleccionar una dificultad y crear un Sudoku pseudo-aleatorio según la selección.	0.5	Ingeniero
Mejora en menú inicial y créditos	Mejoras visuales al menú inicial, creación de ventana informativa sobre la creadora del proyecto.	0.5	Ingeniero
Creación de dificultad difícil	Creación de Sudokus de dificultad difícil de manera eficiente y medianamente rápida, corrección de selector de dificultad.	1	Programador senior
Corrección de errores	Corrección de errores que van apareciendo en el código a medida que se prueba.	1	Ingeniero
Accesibilidad	Cambio de todos los colores y menús para que sean accesibles y añadir una leyenda en el modo accesible dentro del juego.	1	Ingeniero
Tutorial de uso	Crear y diseñar el tutorial de uso que actúa como *tour* por el juego, permitiendo a los usuarios la comprensión y ajustes.	1	Diseñador
Creación de pantalla de ajustes y de puntuación	Diseño y creación de pantalla de ajustes que permiten modificar la experiencia de juego al usuario, creación de pantalla de puntuaciones, gestión de archivos CSV de puntuación y configuración…	1	Ingeniero
Optimización del proyecto	Optimización de todo el código lo máximo posible para ser medianamente eficiente y rápido de ejecutar, últimas mejoras visuales y revisión de funcionalidades.	1	Ingeniero de pruebas
Modularización del código	Repartición del código en archivos distintos para una mayor claridad y modularización.	1	Ingeniero

Tabla 1.4: Planificación temporal del proyecto

Esta estimación da un total de 19 semanas. Cada una cuenta con 5 días laborables (de lunes a viernes), siendo un total de 95 días laborables. Teniendo en cuenta que cada mes tiene aproximadamente 22 días laborables, equivaldría a 4.3 meses de trabajo a jornada completa, debido a que se realiza en temporada de verano. La Ilustración 1.23 muestra el diagrama de Gantt de esta planificación.

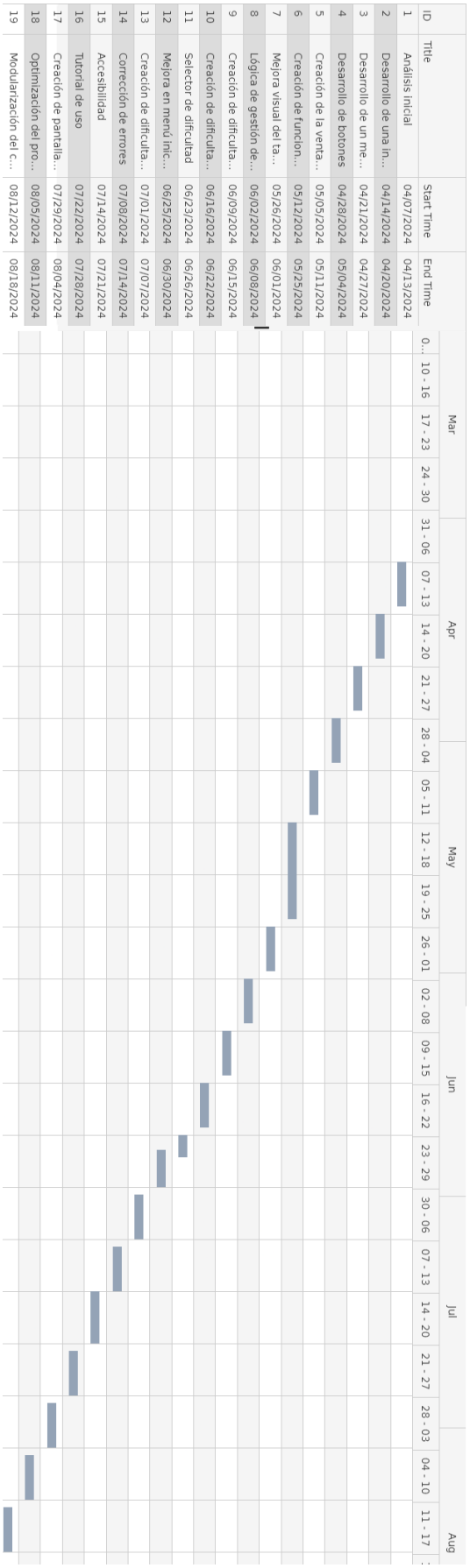


Ilustración 1.23: Diagrama de Gantt de planificación temporal

1.14 Presupuesto

Para desarrollar el presupuesto, se harán uso de varios apartados: elementos software, elementos hardware, personal y costes indirectos. Por último, se incluirá un apartado que resumirá los costes y se estimará el coste total del proyecto.

1.14.1 Elementos software

En este apartado aparecerán los diferentes productos software necesarios para el desarrollo del proyecto por completo. Como ya se ha mencionado anteriormente, el objetivo final del proyecto es proporcionar un software libre y gratuito para la comunidad. Por ello, todos los elementos software se han seleccionado de manera premeditada para que sean gratuitos, y a ser posible, de código abierto para que esté alineado con el propio objetivo.

El único elemento indispensable para el desarrollo que es de pago y no ha podido eludirse es el sistema operativo, mencionado anteriormente en el apartado 1.8. Se trata de Windows 11 versión Home. Todos los elementos se estima que se usarán durante **5 meses** completos.

Elemento	Vida útil estimada	Precio	Amortización	Coste final
Windows 11 Home	3 años	145€	145€/36 meses= 4,02€/mes	20,1€

Tabla 1.5: Coste de elementos software

1.14.2 Elementos hardware

En los elementos hardware únicamente se contará con un ordenador personal.

Elemento	Vida útil estimada	Precio	Amortización	Coste final
ASUS Vivobook Pro 14X OLED	5 años	999,9€	999,9€/60 meses= 16,66€/mes	83,33€

Tabla 1.6: Coste de elementos hardware

1.14.3 Costes indirectos

En los costes indirectos para el desarrollo del proyecto se incluyen los servicios necesarios. Se considerarán 5 meses de uso para cada uno, que es la duración total del proyecto. Estos computarán como un 10% del coste del proyecto.

- **Servicio de luz:** es indispensable la corriente eléctrica para el uso y mantenimiento de los equipos informáticos.
- **Servicio de agua:** al igual que la luz, es indispensable para el personal que trabaje en el proyecto, en este caso la creadora.
- **Servicio de wifi:** también igual que los dos servicios anteriores, la conexión a Internet es obligatoria para el desarrollo del proyecto.

1.14.4 Personal

Para la realización del proyecto serán necesarios 5 empleados, organizados de la siguiente manera:

- **Ingeniero**: se considerará el salario base de un ingeniero *middle* (punto intermedio entre *junior* y *senior*). Estimado por (Glassdoor España, 2024).
- **Programador senior:** se considerará el salario medio en España para un puesto similar. Estimado por (Glassdoor España, 2024).
- **Jefe de proyecto:** se considerará el salario medio en España para un puesto similar. Estimado por (Glassdoor España, 2024).
- **Ingeniero de pruebas:** se considerará el salario medio en España para un puesto similar. Estimado por (Glassdoor España, 2024).
- **Diseñador:** se considerará el salario medio en España para un puesto similar. Estimado por (Glassdoor España, 2024).

Elemento	Coste semanal	Nº semanas	Coste final
Ingeniero	24.000€/52 semanas = 461,5€	8 semanas	3.230,5€
Programador senior	47.500€/52 semanas = 913,4€	4 semanas	4.567€
Jefe de proyecto	40.000€/52 semanas = 769,2€	1 semana	769,2€
Ingeniero de pruebas	22.000€/52 semanas = 423€	1 semana	423€
Diseñador	18.000€/52 semanas = 346,1€	5 semanas	1.730,5€
Subtotal			**10.720,2€**

Tabla 1.7: Coste de salarios

1.14.5 Coste total del proyecto

A continuación se desglosa el coste total del proyecto.

Elemento	Coste
Elementos software	20,1€
Elementos hardware	83,33€
Personal	10.720,2€
Subtotal	**10.823,63€**

Tabla 1.8: Subtotal de costes del proyecto

Elemento	Coste
Subtotal de costes	10.823,63€
Beneficio (10%)	1.082,36€
Costes indirectos (10%)	1.082,36€
Suma total	**12.988,35€**

Tabla 1.9: Subtotal de costes totales del proyecto

Elemento	Coste
Subtotal de costes	12.988,35€
IVA (21%)	2.727,55€
Coste total	**15.715,90€**

Tabla 1.10: Coste total del proyecto

El coste final del proyecto serían **15.715,90€,** incluyendo el IVA del 21%, costes indirectos del 10% que incluyen las diferentes facturas de luz, agua o wifi del periodo y un beneficio del 10% por el desarrollo del juego.

1.15 Normas y referencias

A continuación, se presentan las normas, reglamentos y referencias de cualquier tipo que han sido de aplicación en la elaboración del trabajo y/o en la puesta en práctica del mismo.

Se ha seguido la normativa APA para la bibliografía y referencias usadas en toda la memoria.

1.15.1 Mecanismos de control de calidad

El aseguramiento de la calidad en la elaboración del trabajo implica la verificación de la completitud (falta de omisiones), la integridad de la documentación, así como una redacción clara, concisa y entendible por todos los participantes e interesados en dicho trabajo.

Al estar el trabajo desarrollado por una sola persona y verificado por un tutor, no es necesario llevar un documento de control de edición y revisión de la documentación. De esta forma, se han utilizado mecanismos básicos para la verificación de la integridad y completitud, que incluyen un control de versiones a nivel de documento y un control sencillo de la trazabilidad de especificaciones y requisitos.

Para el control de versiones se ha utilizado en todo momento GitHub (GitHub, 2024) que incluye a Git (Git, 2024), también incluido en el entorno de desarrollo elegido Visual Studio Code (Microsoft, 2024).

2 DISEÑO INICIAL

En todo producto software debe haber una fase en la que se defina el diseño inicial que tendrá el sistema, ya que esto evitará que se produzcan errores durante el desarrollo debidos a una mala definición de los requisitos, alcance o diseño.

2.1 Especificaciones del sistema

La definición de requisitos es una de las tareas más importantes a realizar antes de implementar cualquier producto software. Existen dos tipos de requisitos:

- **Requisitos funcionales**: son los requisitos que describen lo que el sistema deberá hacer.
- **Requisitos no funcionales**: son los requisitos que se enfocan en cómo debe comportarse el sistema desarrollado o las restricciones que deberá tener.

A continuación, se hará una exposición detallada de los requisitos definidos del producto software a desarrollar, tanto los funcionales como los no funcionales.

2.1.1 Requisitos funcionales

1. **RF1:** El programa debe ser capaz de crear una partida de Sudoku. Se debe poder elegir el nivel de dificultad deseado (seleccionando entre Fácil, Medio y Difícil).
2. **RF2**: El programa debe permitir la resolución de un Sudoku, permitiendo al usuario rellenar las casillas del tablero con números del 1 al 9. Se deberá proporcionar también la opción de comprobar si la solución es correcta.
3. **RF3:** El programa debe generar tableros pseudo-aleatorios de Sudoku automáticamente.
4. **RF4**: El programa debe validar los números introducidos por el usuario y comprobar si son correctos, es decir, si cumplen las reglas, comprobando que el mismo número no se repita en la misma celda 3x3, fila o columna.
5. **RF5**: El programa debe mostrar el tiempo transcurrido desde que se ha comenzado a jugar en el tablero, permitiendo también pausar el cronómetro o reanudarlo para una medición más precisa.
6. **RF6**: El programa debe almacenar y mostrar la puntuación, que será expresada en minutos, de manera que exista una tabla de puntuaciones en las que se almacenen las mejores partidas del jugador.
7. **RF7**: El programa debe tener una interfaz gráfica para mostrar el tablero y permitir la interacción con el jugador.
8. **RF8**: El programa debe permitir resolver de manera automática e instantánea el Sudoku, mostrando la solución correcta al jugador.

2.1.2 Requisitos no funcionales

1. **RNF1**: Rendimiento. El sistema debe generar tableros de Sudoku en menos de 4 segundos.
2. **RNF2**: Usabilidad. La interfaz gráfica debe ser intuitiva y fácil de usar, incluso para usuarios nuevos. También debe permitir al usuario interactuar con el tablero usando el ratón y el teclado.
3. **RNF3**: Escalabilidad. El sistema debe estar preparado para futuras mejoras y adiciones posteriores de funcionalidades sin comprometer el rendimiento.

4. **RNF4**: Accesibilidad. El sistema debe ser accesible para usuarios con algún tipo de discapacidad, permitiendo cambiar de modo para facilitar la comprensión del juego.

2.2 Análisis y diseño del sistema

2.2.1 Análisis del sistema

En este caso, el juego de Sudoku tendrá únicamente un tipo de usuario, al que llamaremos jugador, tendrá una interfaz simple pero completa, permitiendo al jugador mantener un registro de sus mejores partidas, modificar los ajustes de accesibilidad y aumentar o disminuir la dificultad máxima, para jugadores expertos o que estén aprendiendo aún y quieran ir más despacio.

Para iniciar el juego, el jugador únicamente tendrá que darle a un botón para empezar y, después de seleccionar la dificultad deseada, aparecerá el Sudoku y empezará a correr el temporizador.

2.2.2 Diseño del sistema

2.2.2.1 Diseño preliminar

En esta fase, se definirá la estructura básica del sistema, cómo interactúan los componentes y la arquitectura que soporta el juego. El objetivo es la obtención de una vista general de cómo quedan conectadas las distintas partes del sistema.

2.2.2.1.1 Arquitectura del sistema

El juego funciona completamente de forma local, sin necesidad de un servidor, y debido a que no tiene características en línea, se puede jugar sin necesidad de conexión a Internet.

Entre los diversos componentes del sistema se encuentran los mostrados en la Ilustración 2.1.

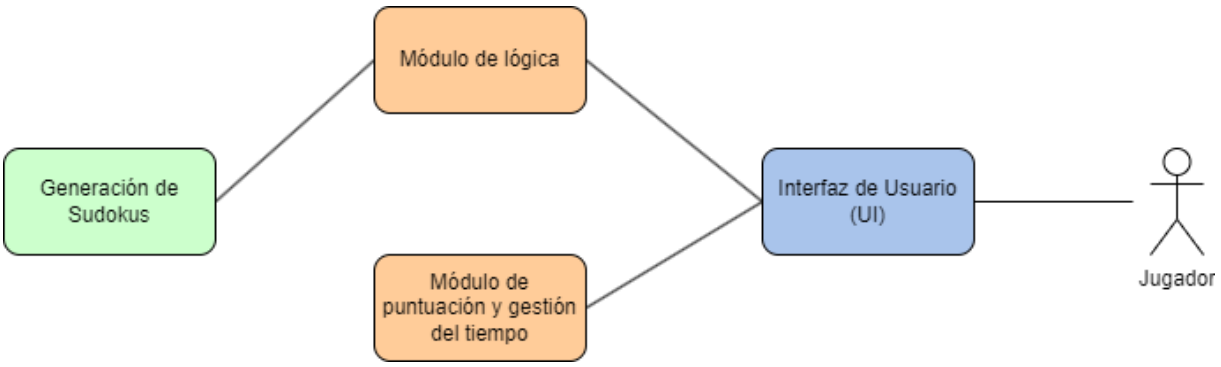


Ilustración 2.1: Organización de componentes del sistema

- **Interfaz de Usuario (UI):** es la interfaz gráfica donde el jugador podrá ver y manipular el tablero de Sudoku para jugar.
- **Módulo de lógica del juego:** es la parte más interna del juego, donde están detalladas las reglas del mismo, se hará la generación de tableros, la validación de movimientos y todas las acciones que el usuario realice, como borrar el tablero, reiniciar el temporizador o resolver el tablero.
- **Módulo de puntuación y gestión de tiempo:** estos dos módulos van de la mano, puesto que la medida de tiempo es la puntuación que se usa en el juego, siendo el menor tiempo la mejor puntuación y el mayor tiempo la peor puntuación.
- **Generación de Sudokus:** contendrá los algoritmos necesarios para la creación y la resolución de tableros de Sudoku pseudo-aleatorios con dificultad variable, todos ellos resolubles y válidos.

2.2.2.1.2 Flujos de trabajo

Los diversos flujos de trabajo que se definen en este proyecto serán:

- **Crear una partida:** el jugador seleccionará el nivel de dificultad deseado y el sistema generará un nuevo tablero de Sudoku adaptado a esa dificultad.
- **Interacción con el tablero:** el jugador introducirá mediante teclado los números del 1 al 9 en la celda que seleccione mediante el ratón, también podrá borrar el número introducido, y se validará en tiempo real si se infringen las reglas básicas del Sudoku.

- **Resolución de Sudoku:** existen dos opciones: o bien el jugador ha completado el tablero y desea comprobar si lo ha resuelto correctamente, o bien el jugador no sabe cómo continuar y desea ver la solución.
- **Mostrar puntuación y tiempo:** en este caso, el tiempo actuará de puntuación, contabilizándolo en minutos y segundos. Al final de la partida, el sistema mostrará el tiempo transcurrido y la dificultad en la que se ha conseguido, para después almacenar esta información.

2.2.2.2 Diseño detallado

En este apartado se detallará la descripción técnica de los componentes y módulos que conforman el sistema al completo.

2.2.2.2.1 Interfaz de Usuario (UI)

La interfaz está desarrollada usando el lenguaje de programación Python, junto a la biblioteca PyGame, que permite la creación de juegos simples 2D. Posee varios elementos, como son el propio tablero de Sudoku, botones para gestionar el tiempo, mostrar solución y borrar tablero, entre otros. También se cuenta con varios menús, un panel de información para el tiempo transcurrido y una pantalla de ajustes que permite al usuario configurar el juego a su gusto de manera simple.

2.2.2.2.2 Módulo de lógica del juego

Este módulo se puede dividir a su vez en varios sub-módulos, como son:

1. **Validación de Sudoku**: cuando se introduce un número por teclado, se comprueba de manera instantánea que no coincida con otro número igual en su celda adyacente 3x3, en su misma fila y en su misma columna.
2. **Generación de Sudoku**: se usa un algoritmo de *backtracking* para generar tableros que sean válidos y resolubles. Más información en el apartado 2.2.2.2.4.
3. **Algoritmo de resolución automática:** se mostrará la resolución del Sudoku cuando el usuario lo desee de manera instantánea, ya que es un tablero que ya posee el programa de manera previa, así que solamente quedará mostrarlo.
4. **Gestión de archivos:** el programa gestionará dos archivos CSV, uno de puntuación y otro de configuración, que inicialmente estará con valores por

defecto, y en el caso de la puntuación, vacío. Si los archivos no existen, se crearán en la carpeta personal del usuario, en una carpeta llamada *SudoKu.*

2.2.2.2.3 Módulo de puntuación y gestión de tiempo

Como se ha mencionado anteriormente, la puntuación y el tiempo van muy ligados, ya que el sistema que puntúa la partida se basa únicamente en el tiempo transcurrido desde el inicio hasta la resolución final y correcta del Sudoku. La fórmula seguida es menor tiempo = mejor puntuación.

2.2.2.2.4 Generador de Sudoku

El generador de Sudokus está compuesto por varios módulos a su vez. La Ilustración 2.2 muestra un diagrama de flujo de su funcionamiento.

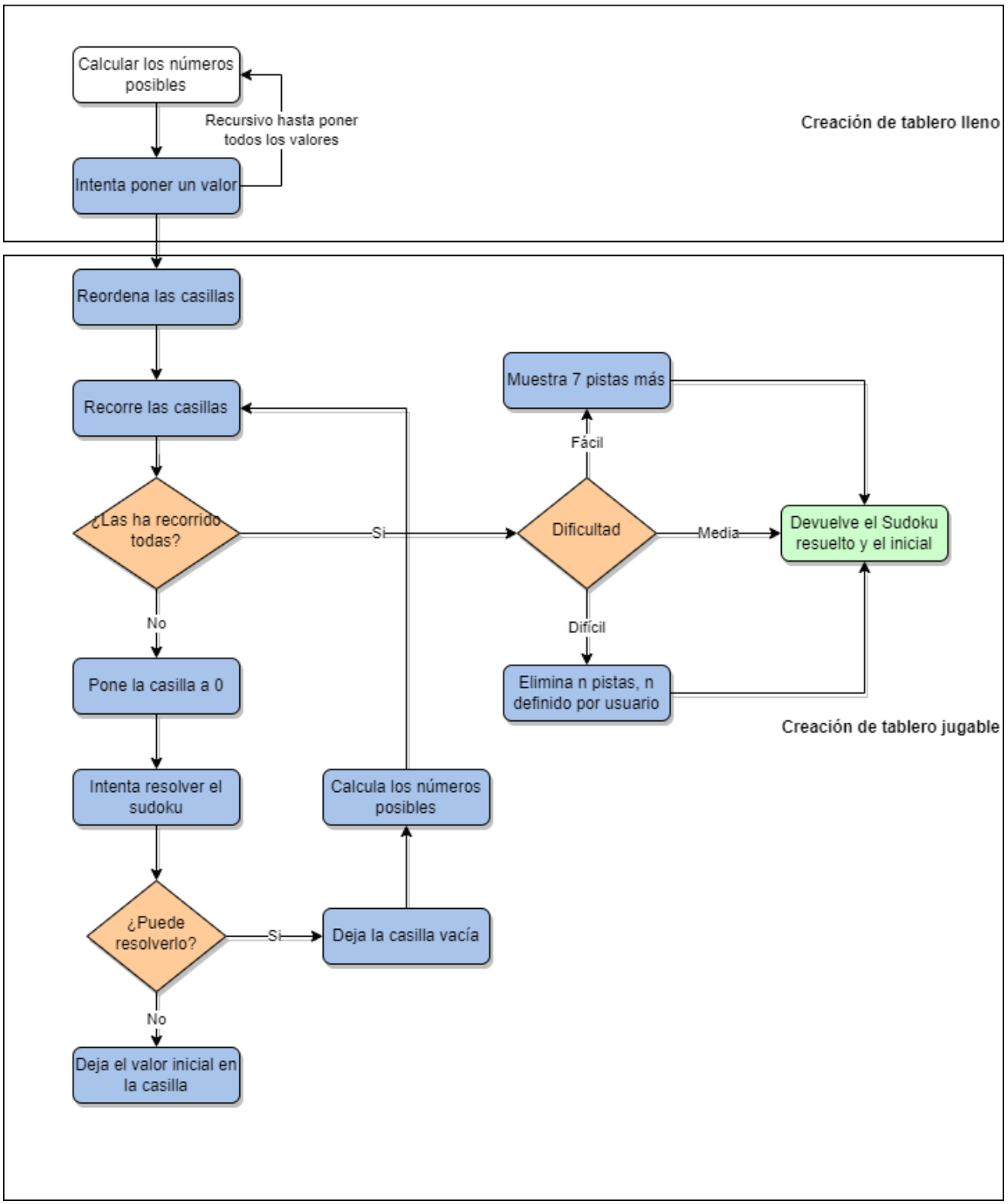


Ilustración 2.2: Diagrama de flujo del módulo de creación de Sudokus

Se descompone en los siguientes pasos, divididos en la primera parte que es la creación del tablero lleno y la segunda parte, que es la creación del tablero jugable, con algunas casillas vacías.

1. **[Creación de tablero lleno] Calcula los números posibles**: para cada celda vacía del Sudoku, calcula qué números se pueden poner sin saltarse las reglas del juego. Inicialmente, como el tablero está vacío, en cada celda los valores posibles serán [1, 9].
2. **[Creación de tablero lleno] Intenta poner un valor**: reordena pseudo-aleatoriamente los valores posibles de cada casilla. Teniendo los valores posibles para cada una, se intenta poner el primer valor y devuelve si se ha podido insertar o no (inicialmente podrá), después se llama recursivamente a la función 1 para cada una de las casillas. Sigue así sucesivamente hasta recorrer el tablero entero e ir rellenando las casillas usando los nuevos valores posibles calculados, el resultado será un tablero completo.
3. **[Creación de tablero jugable] Reordena las casillas**: inserta en un vector todas las casillas, de forma [[0,0], [0,1], [0,2], ... , [9,9]] y las reordena de manera pseudo-aleatoria para empezar a eliminar pistas.
4. **[Creación de tablero jugable] Recorre las casillas reordenadas**: en cada iteración selecciona una casilla concreta.
5. **[Creación de tablero jugable] Comprueba si las ha recorrido todas**: en caso de no haberlas recorrido por completo, continúa con el siguiente paso. En caso contrario, comprobará la dificultad introducida por el usuario.
    - Si la dificultad es **Fácil**, partiendo del Sudoku sin resolver del que se dispone ahora mismo y el Sudoku resuelto, se seleccionan pseudo aleatoriamente 7 casillas y se añaden al Sudoku sin resolver como pistas iniciales. Posteriormente se devuelve el Sudoku resuelto y el que no está resuelto aún.
    - Si la dificultad es **Media**, se devuelve el Sudoku resuelto y el que no está resuelto, sin modificaciones.
    - Si la dificultad es **Difícil**, se seleccionan n pistas, siendo n el parámetro configurable por el jugador en los ajustes, y se eliminan esas pistas, haciendo que únicamente se pueda resolver usando métodos de *backtracking*. Por último se devuelve el Sudoku resuelto y sin resolver.
6. **[Creación de tablero jugable] Pone la casilla a 0 e intenta resolver el Sudoku:** el programa comprueba, en el caso hipotético de que esa casilla no

contenga ningún valor, ¿se podría resolver el Sudoku? Si el resultado es que no se puede, deja el valor como estaba inicialmente, pero en caso contrario dejará la casilla vacía y calculará de nuevo los números posibles como al inicio, en el apartado 1.

2.2.2.3 Diagrama de clases

El diagrama de clases del proyecto quedaría como se muestra en la Ilustración 2.3.

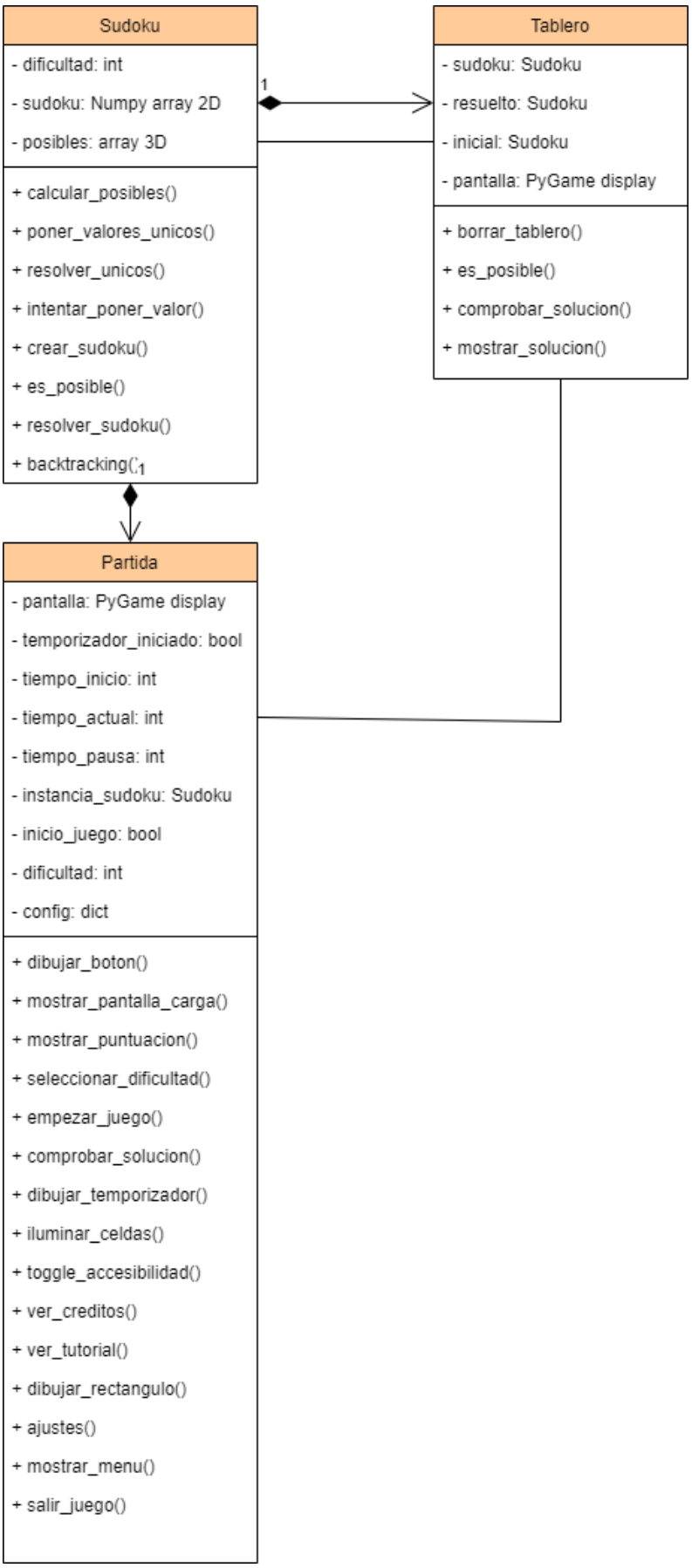


Ilustración 2.3: Diagrama de clases del proyecto

Se puede apreciar que hay 3 clases principales, *Sudoku*, *Partida* y *Tablero*. En cuanto a relaciones, hay una asociación entre *Sudoku* y *Tablero* para gestionar la interacción del usuario con el Sudoku desde el tablero, una composición entre *Sudoku* y *Partida*, ya que desde el juego se mostrará el Sudoku. Hay también una asociación entre *Tablero* y *Partida*, que permitirá interferir con el tablero desde el propio juego, usando teclado y ratón. Por último, una composición entre *Tablero* y *Sudoku*, ya que el tablero tendrá varios objetos de tipo Sudoku.

2.2.2.4 Casos de uso

De acuerdo a lo explicado en el apartado 2.1.1 sobre los requisitos funcionales, vemos que coinciden con los casos de uso, y por lo tanto, con la funcionalidad de todo el sistema, por lo que se establecen los que se muestran en la Tabla 2.1.

ID	Actor	Caso de uso
1	Jugador	Crear una partida de Sudoku
2	Jugador	Resolver un Sudoku y comprobar la solución
3	Jugador	Generar tableros pseudo-aleatorios de Sudoku
4	Jugador	Validar números introducidos por el usuario
5	Jugador	Mostrar y pausar el cronómetro
6	Jugador	Almacenar y mostrar puntuación
7	Jugador	Interacción gráfica con el tablero
8	Jugador	Resolución automática del Sudoku

Tabla 2.1: Casos de uso definidos en el proyecto

2.2.2.4.1 Diagrama de casos de uso

Con todos los casos de uso definidos, el diagrama de la Ilustración 2.4 los incluye a todos:

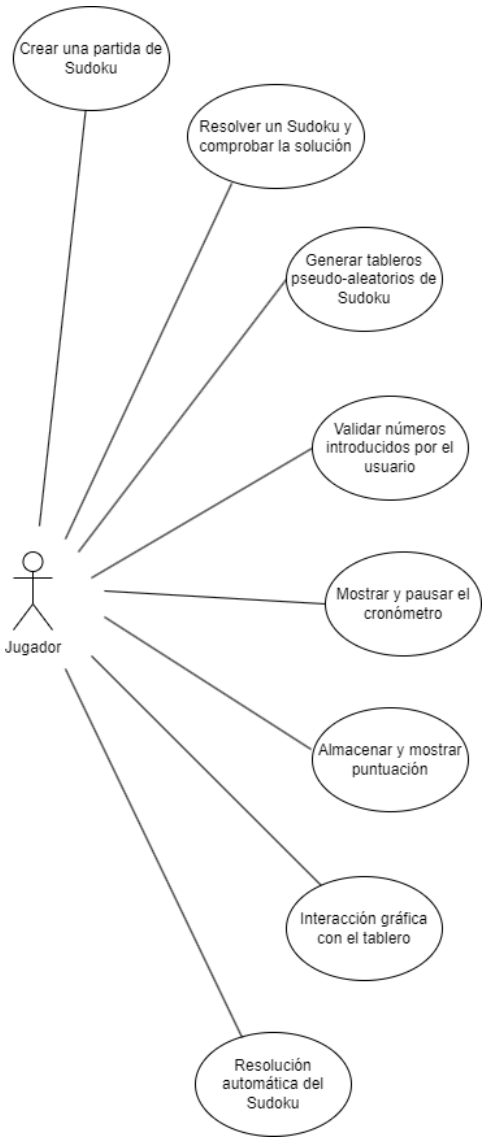


Ilustración 2.4: Diagrama de casos de uso

2.2.2.4.2 Especificaciones de los casos de uso

A continuación en las tablas Tabla 2.2 a Tabla 2.9 quedarán detallados todos los casos de uso definidos previamente, junto a sus actores, descripción, condiciones previas, operaciones básicas, escenario principal y escenario alternativo, en caso de ser necesario. Cada caso de uso está vinculado a un requisito funcional.

2.2.2.4.2.1 Crear una partida de Sudoku (RF1)

Caso de uso	Crear una partida de Sudoku
ID	1
Descripción	El jugador crea una nueva partida de Sudoku y selecciona el nivel de dificultad.
Actor primario	Jugador
Condiciones previas	El juego debe estar en la pantalla inicial o en un estado listo para crear una nueva partida.
Operaciones básicas	Selección del nivel de dificultad
Escenario principal	1. El jugador selecciona "Empezar" en el menú principal. 2. El sistema presenta las opciones de dificultad: Fácil, Medio, Difícil. 3. El jugador selecciona el nivel. 4. El sistema genera un tablero de Sudoku y lo muestra.
Escenarios alternativos	

Tabla 2.2: Especificación de caso de uso (RF1)

2.2.2.4.2.2 Resolver un Sudoku y comprobar la solución (RF2)

Caso de uso	Resolver un Sudoku y comprobar la solución
ID	2
Descripción	El jugador introduce números en las celdas del tablero y verifica si la solución es correcta.
Actor primario	Jugador
Condiciones previas	El tablero debe haber sido generado y estar visible en la pantalla.
Operaciones básicas	Introducir números en celdas, verificar solución.
Escenario principal	1. El jugador selecciona una celda vacía. 2. Introduce un número entre 1 y 9. 3. El jugador selecciona la opción "Comprobar solución". 4. El sistema valida la solución. 5. El sistema informa si la solución es correcta o incorrecta.
Escenarios alternativos	2.a. Si el jugador introduce un número fuera del rango 1-9, el sistema no lo inserta en el tablero.

Tabla 2.3: Especificación de caso de uso (RF2)

2.2.2.4.2.3 Generar tableros pseudo-aleatorios de Sudoku (RF3)

Caso de uso	Generar tableros pseudo-aleatorios de Sudoku
ID	3
Descripción	El sistema genera un tablero pseudo-aleatorio de Sudoku automáticamente.
Actor primario	Sistema
Condiciones previas	El jugador debe haber seleccionado un nivel de dificultad.
Operaciones básicas	Generación automática de tableros
Escenario principal	1. El jugador selecciona "Empezar". 2. El jugador selecciona la dificultad deseada. 3. El sistema genera un tablero pseudo-aleatorio. 4. El tablero se presenta al jugador.
Escenarios alternativos	

Tabla 2.4: Especificación de caso de uso (RF3)

2.2.2.4.2.4 Validar números introducidos por el usuario (RF4)

Caso de uso	Validar números introducidos por el usuario
ID	4
Descripción	El sistema valida si los números introducidos por el jugador en el tablero son correctos.
Actor primario	Jugador
Condiciones previas	El jugador debe haber iniciado una partida y el tablero debe estar visible.
Operaciones básicas	Validación de números según las reglas del Sudoku
Escenario principal	1. El jugador introduce un número en una celda. 2. El sistema verifica si el número es válido (no se repite en la fila, columna o subcuadro). 3. Si es válido, se actualiza el tablero.
Escenarios alternativos	3.a. Si no es válido, se cambiará su color a rojo.

Tabla 2.5: Especificación de caso de uso (RF4)

2.2.2.4.2.5 Mostrar y pausar el cronómetro (RF5)

Caso de uso	Mostrar y pausar el cronómetro
ID	5
Descripción	El sistema muestra el tiempo transcurrido y permite pausar y reanudar el cronómetro.
Actor primario	Jugador
Condiciones previas	El tablero debe estar generado y el cronómetro debe haber comenzado.
Operaciones básicas	Pausar y reanudar el cronómetro
Escenario principal	1. El sistema muestra el tiempo en pantalla. 2. El jugador selecciona el botón de pausa. 3. El cronómetro se detiene. 4. El jugador selecciona el botón de reanudar. 5. El cronómetro se reanuda.
Escenarios alternativos	

Tabla 2.6: Especificación de caso de uso (RF5)

2.2.2.4.2.6 Almacenar y mostrar puntuación (RF6)

Caso de uso	Almacenar y mostrar puntuación
ID	6
Descripción	El sistema almacena la puntuación en minutos y segundos, y muestra una tabla de las mejores puntuaciones.
Actor primario	Jugador
Condiciones previas	El jugador debe haber completado el Sudoku o haberse resuelto automáticamente.
Operaciones básicas	Almacenar y mostrar puntuaciones
Escenario principal	1. El jugador completa el Sudoku. 2. El sistema calcula la puntuación basada en el tiempo. 3. El sistema almacena la puntuación en la tabla. 4. El jugador accede a la tabla de puntuaciones y la visualiza.
Escenarios alternativos	

Tabla 2.7: Especificación de caso de uso (RF6)

2.2.2.4.2.7 Interacción gráfica con el tablero (RF7)

Caso de uso	Interacción gráfica con el tablero
ID	7
Descripción	El jugador interactúa con el tablero mediante una interfaz gráfica, ingresando números y navegando por las celdas.
Actor primario	Jugador
Condiciones previas	El tablero debe estar visible en la interfaz gráfica.
Operaciones básicas	Interacción con el tablero mediante clics y teclado
Escenario principal	1. El sistema presenta el tablero en la interfaz. 2. El jugador selecciona una celda. 3. El jugador introduce un número en la celda seleccionada. 4. El sistema actualiza el tablero con el número ingresado.
Escenarios alternativos	

Tabla 2.8: Especificación de caso de uso (RF7)

2.2.2.4.2.8 Resolver automáticamente el Sudoku (RF8)

Caso de uso	Resolver automáticamente el Sudoku
ID	8
Descripción	El jugador solicita al sistema que resuelva automáticamente el Sudoku.
Actor primario	Jugador
Condiciones previas	El tablero de Sudoku debe estar generado y el jugador debe estar en medio de una partida.
Operaciones básicas	Resolución automática del Sudoku
Escenario principal	1. El jugador selecciona "Mostrar solución". 2. El sistema completa el tablero con la solución correcta. 3. El tablero se muestra resuelto.
Escenarios alternativos	

Tabla 2.9: Especificación de caso de uso (RF8)

2.2.2.5 Diseño de la interfaz

Las ilustraciones Ilustración 2.5 a Ilustración 2.12 muestran los diversos wireframe usados para el diseño de la interfaz, uno por cada pantalla que deberá tener el producto software desarrollado.

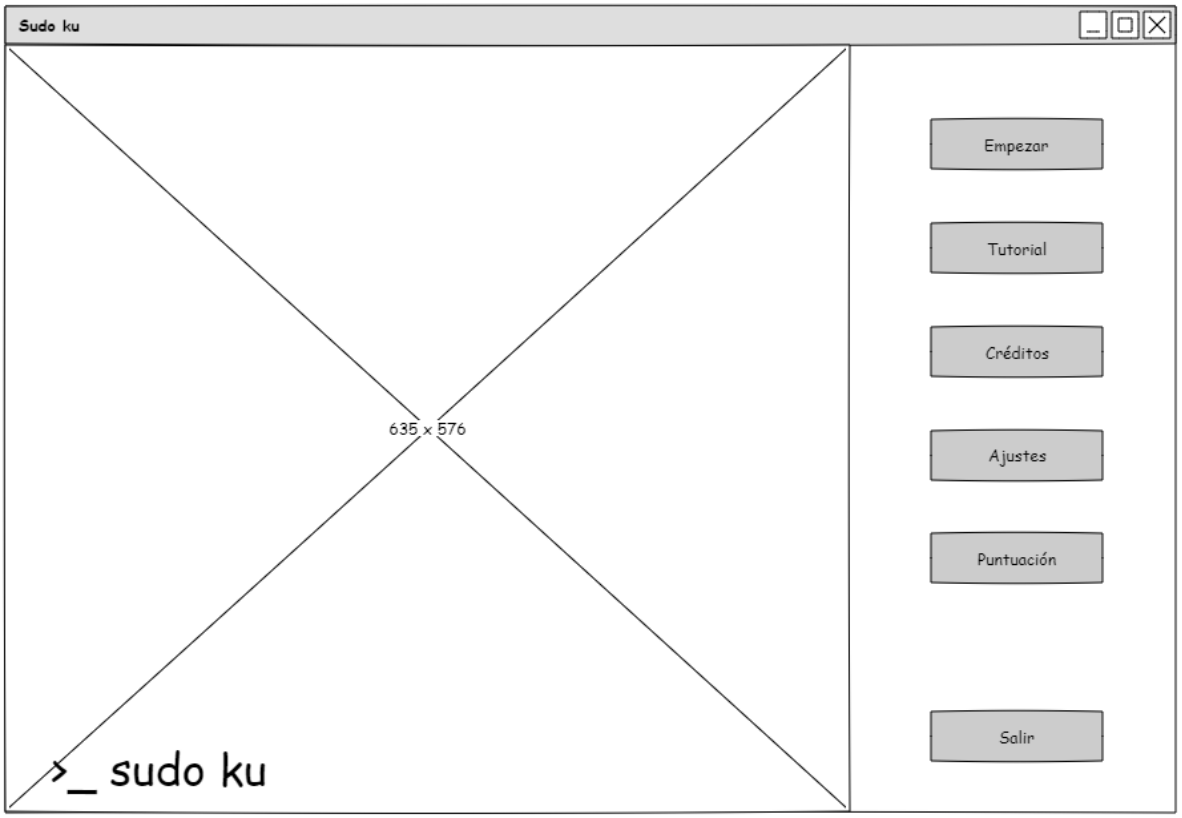


Ilustración 2.5: Wireframe del menú inicial

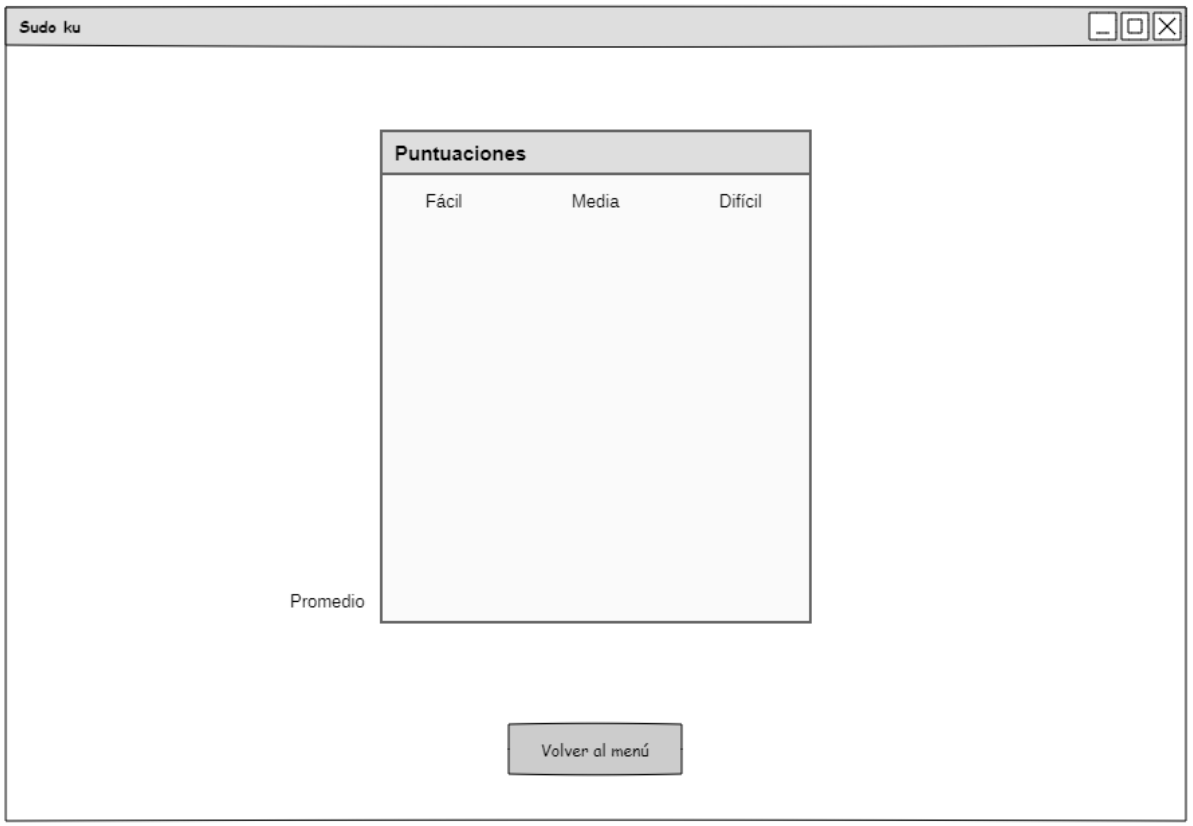


Ilustración 2.6: Wireframe de la pantalla de puntuaciones

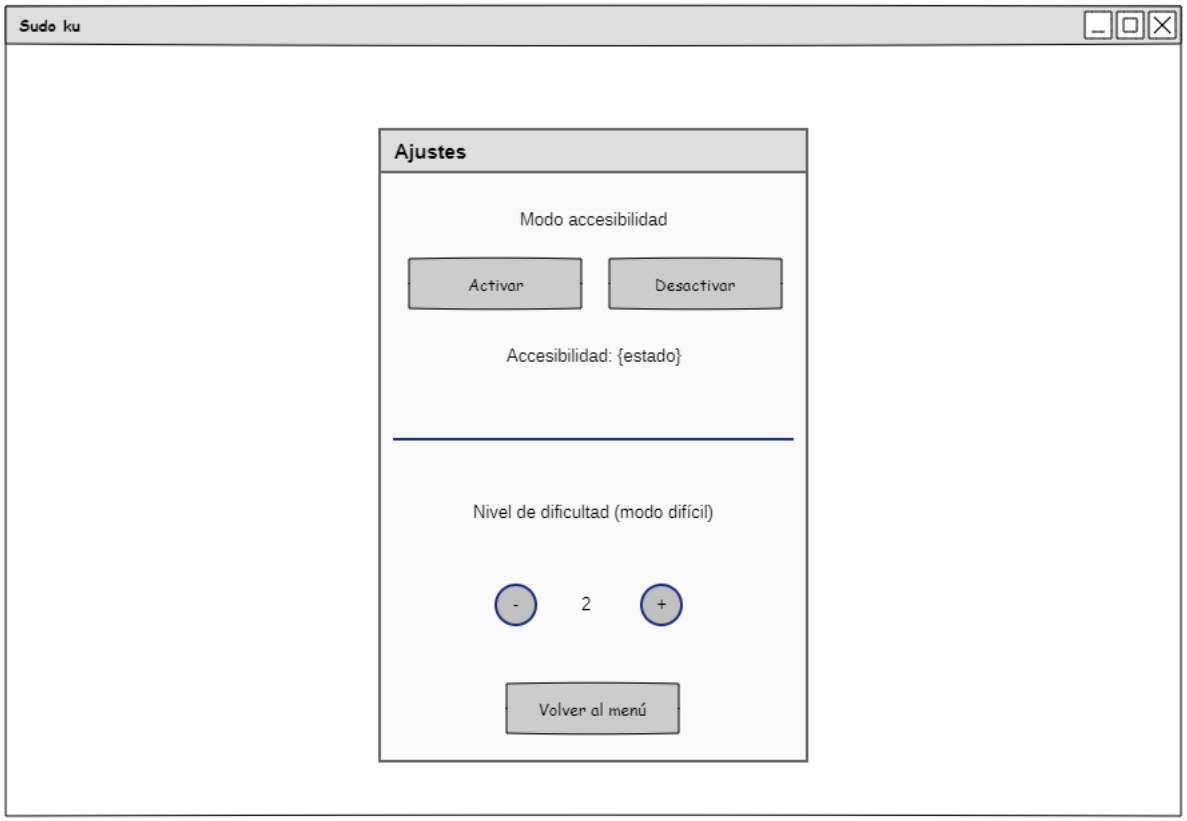


Ilustración 2.7: Wireframe de la pantalla de ajustes

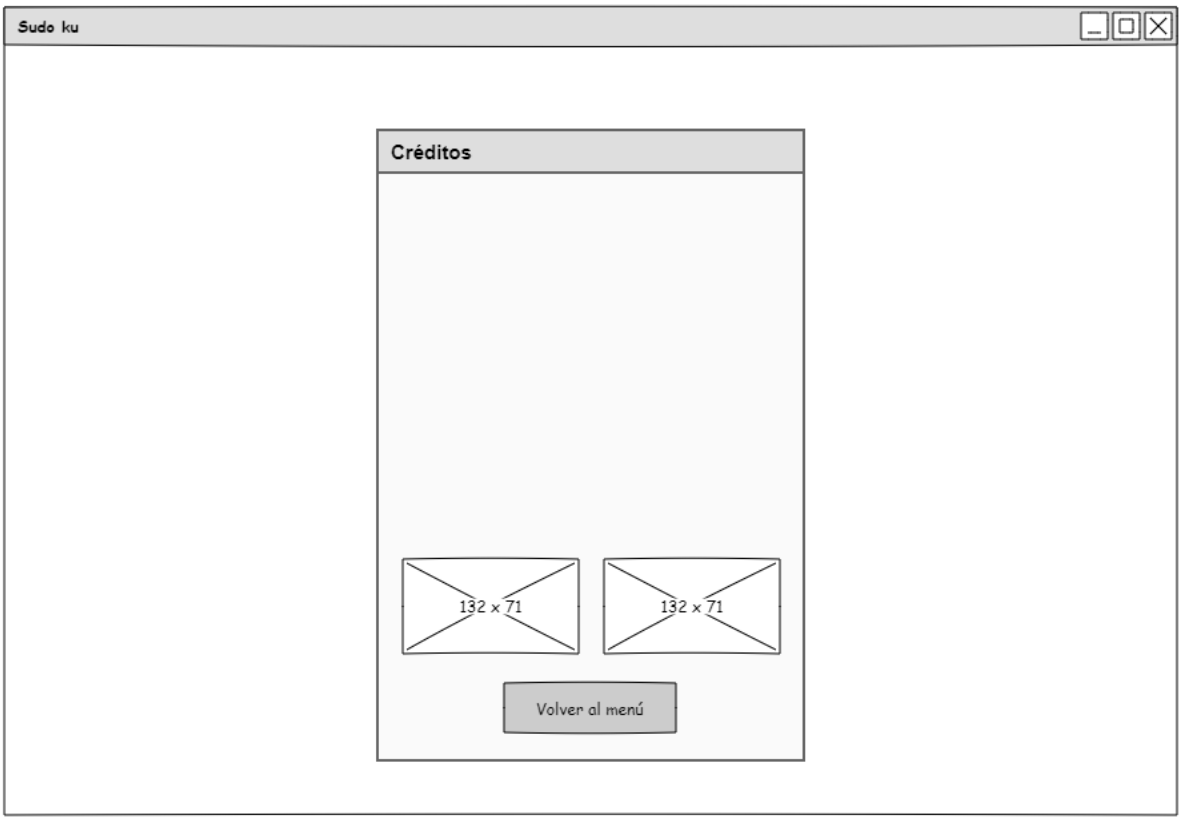


Ilustración 2.8: Wireframe de la pantalla de créditos

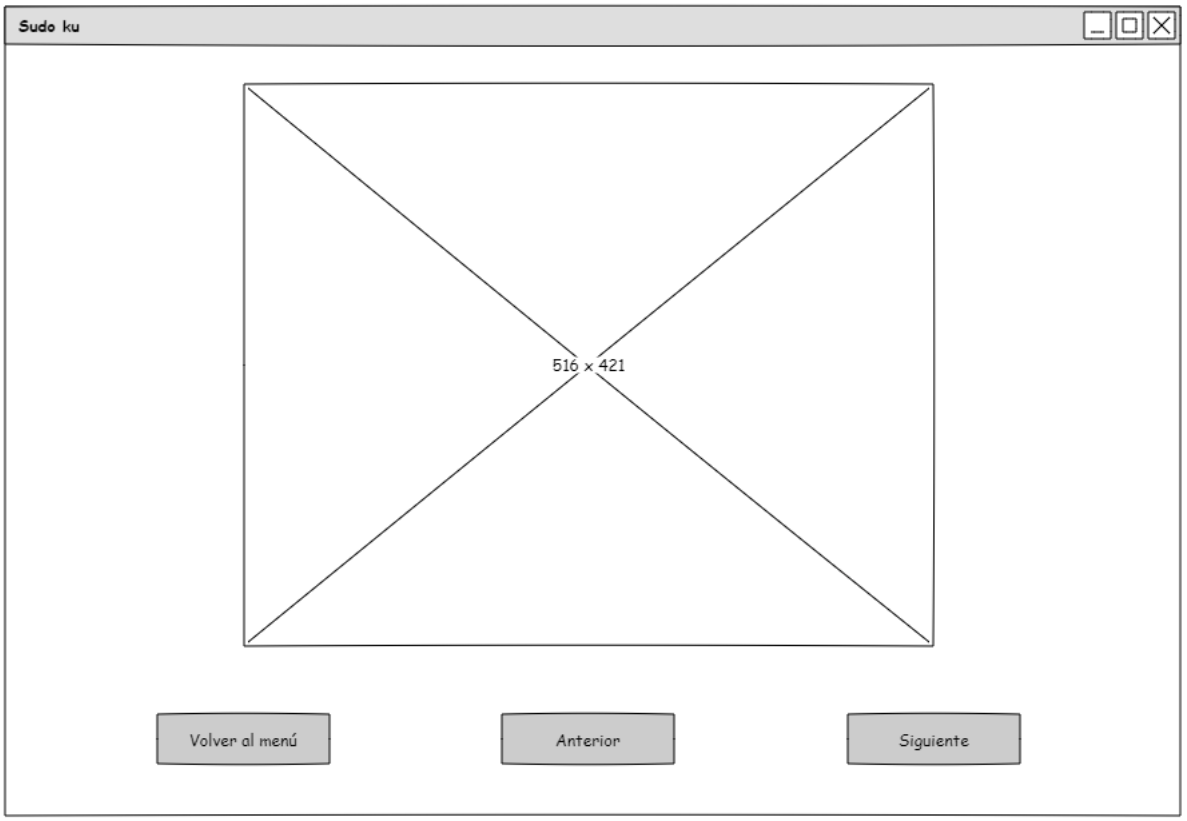


Ilustración 2.9: Wireframe de la pantalla de tutorial

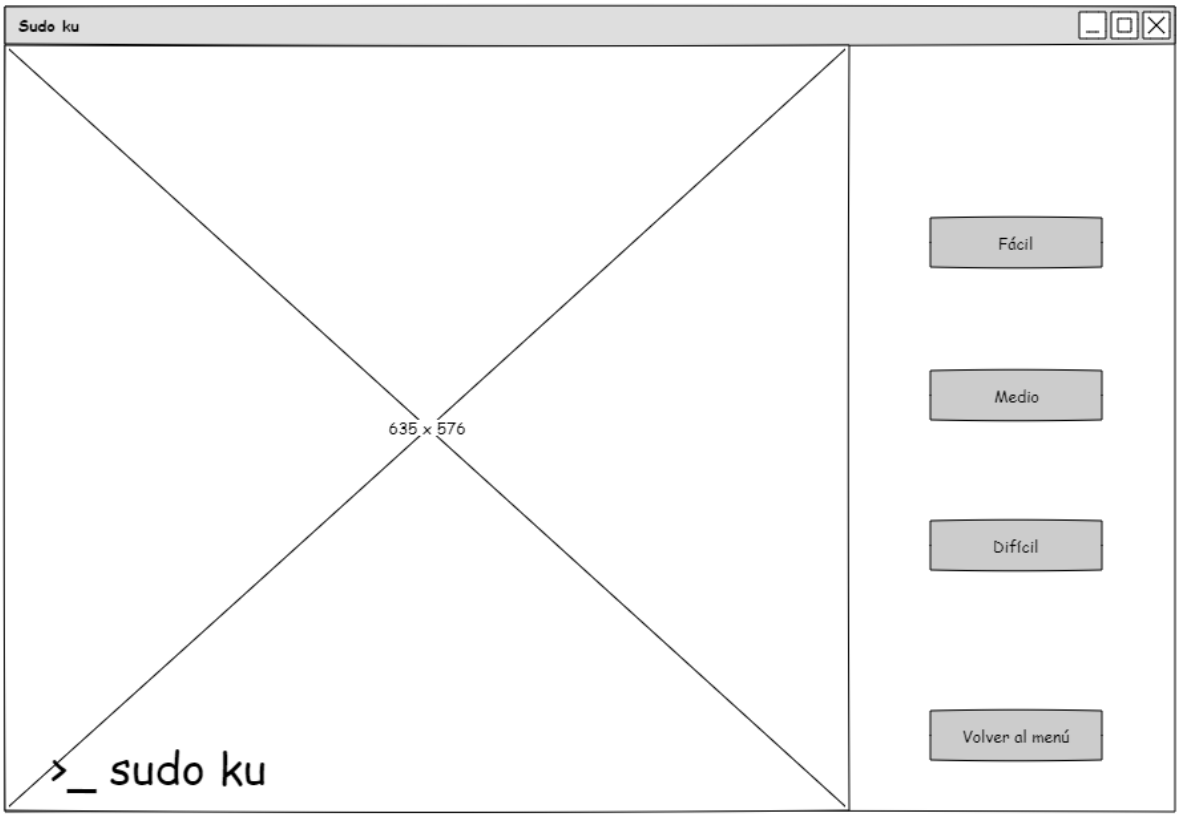


Ilustración 2.10: Wireframe de la pantalla de selección de dificultad

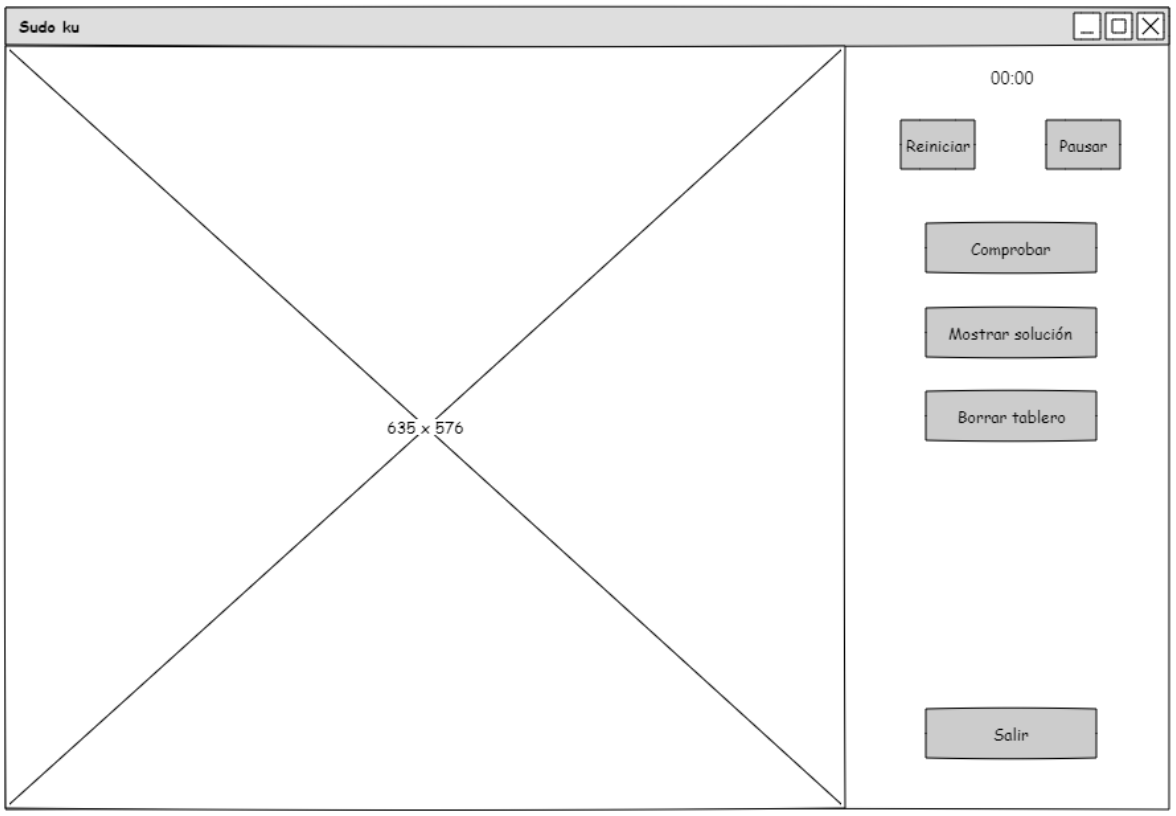


Ilustración 2.11: Wireframe de la partida de Sudoku

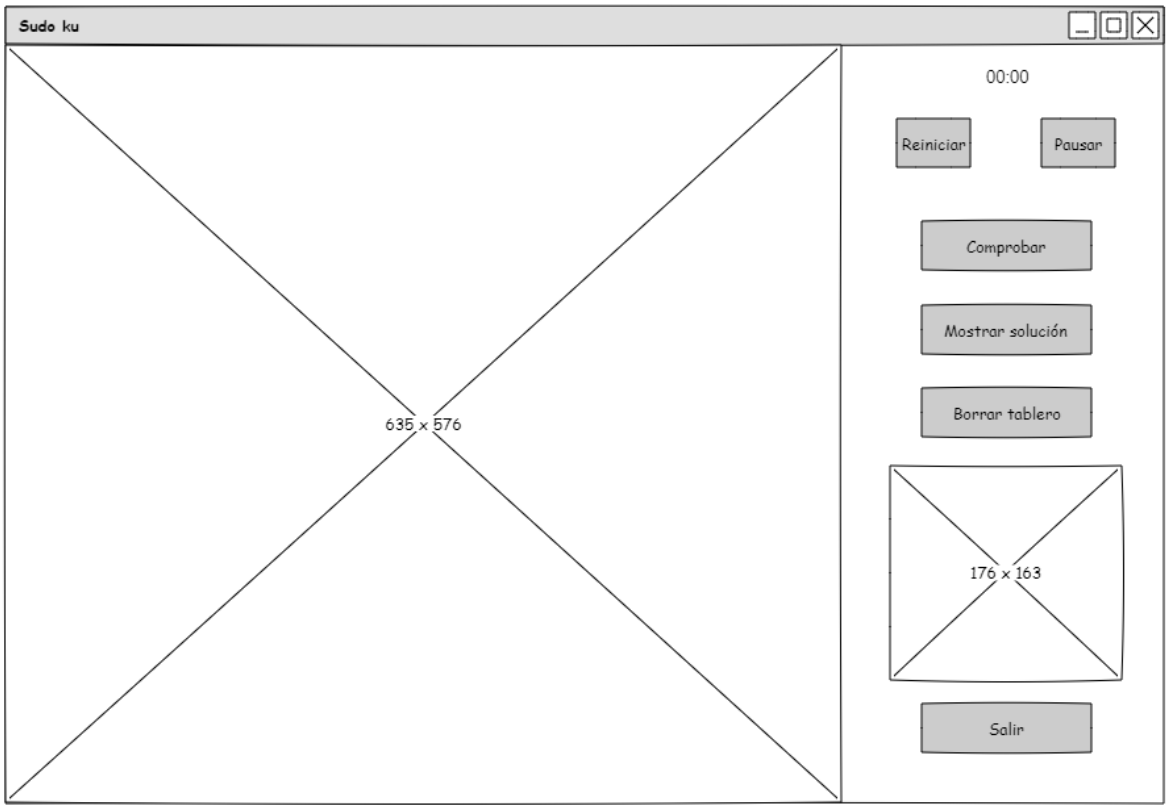


Ilustración 2.12: Wireframe de la partida de Sudoku en modo accesible

3 DESARROLLO

Como ya se ha mencionado anteriormente, la metodología de desarrollo a utilizar es una metodología **incremental**, por lo que en los apartados siguientes se explicará cada incremento realizado de manera detallada y con imágenes para comprobar el avance de una forma mucho más visual.

3.1 Primera iteración: Análisis e investigación

El paso previo a todas las iteraciones es el de investigación. A continuación se detallarán todos los aspectos investigados.

- La primera investigación fue el **análisis de las alternativas gratuitas** que ofrece el mercado actual. Para ello, se siguió un proceso de búsqueda exhaustivo en varias herramientas de descargas de juegos, luego se realizó un filtrado para descartar los productos software menos interesantes y posteriormente se procedió a la descarga de todos ellos. La prueba consistía en jugar una partida en cualquier modo y ver qué llamaba la atención, qué se podía mejorar y qué destacaba como aspecto positivo de cada alternativa.
- Después se investigó acerca de la creación y clasificación de sudokus según su nivel de dificultad. Las conclusiones de esta investigación fueron las siguientes:
    - El nivel de dificultad viene determinado por el número de pistas y el número de veces que el usuario encuentra un punto en el que se bifurca el árbol de decisiones. Esto último quiere decir que un Sudoku se podría considerar de nivel difícil si y sólo si tiene al menos un **punto de decisión** en el que hay varias alternativas igualmente válidas. En este caso, el usuario deberá elegir un camino y comprobar si lleva a una solución. En caso de no ser resoluble por ese camino, deberá volver hacia atrás para probar la otra alternativa. El **número de pistas** influye significativamente en la dificultad, considerando que a menor número de pistas dadas inicialmente, mayor dificultad de resolución del nivel. Por último, también se descubrió que el número mínimo de pistas que debe tener un Sudoku es 17. Por debajo de ese número deja de ser resoluble por su complejidad, según (McGuire, 2012).

- Los tableros de Sudoku normalmente se crean usando ***backtracking***, de manera que se intenta poner un número en cada celda de la matriz 9x9, se comprueba si viola las reglas del sudoku y en caso de que el número no esté permitido, lo elimina y prueba con otro.

3.2 Segunda Iteración: Creación de una ventana base

En la segunda iteración, la primera de desarrollo, el objetivo fue la creación del proyecto, el desarrollo de una ventana básica inicial que sirviese de base para las siguientes iteraciones y la creación de un menú que apareciese al ejecutar, de manera que permitía al usuario seleccionar lo que deseara hacer a continuación (Ilustración 3.1).

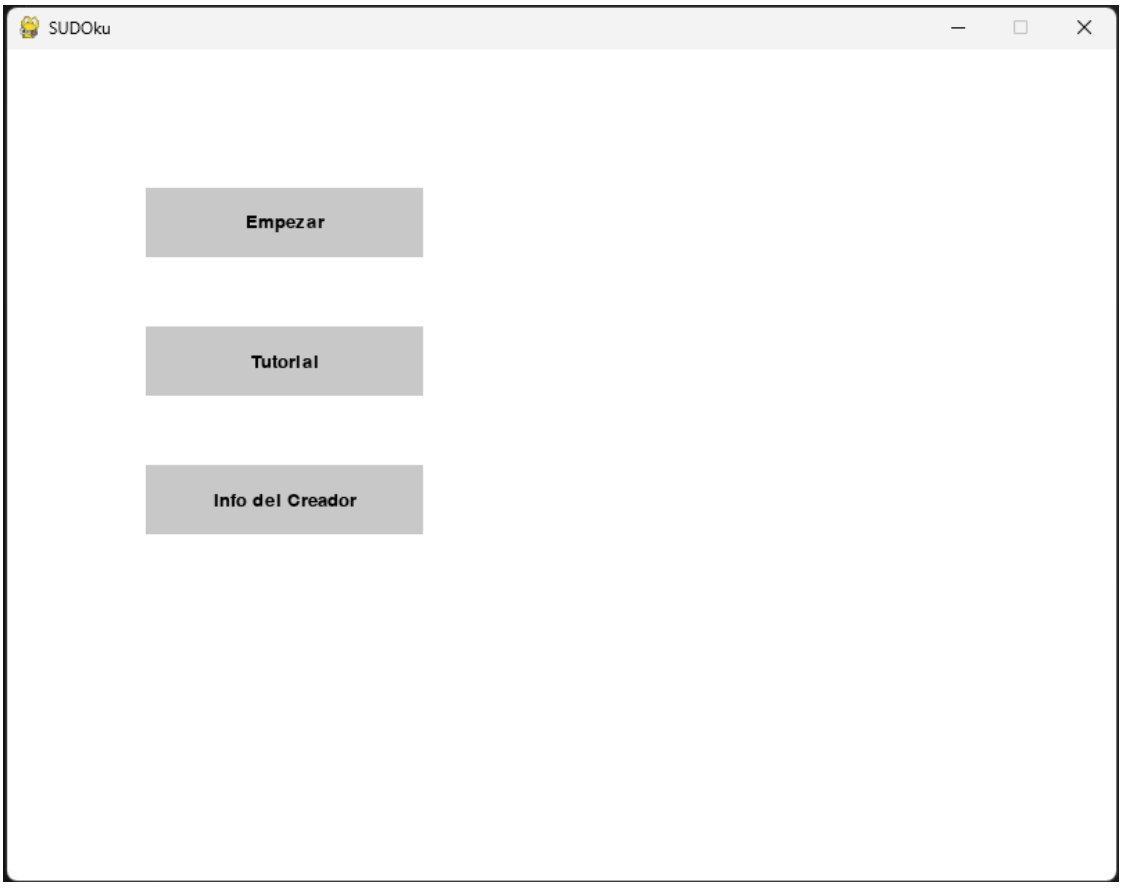


Ilustración 3.1: Ventana inicial de la segunda iteración

Como se puede ver en la Ilustración 3.1: Ventana inicial de la segunda iteración, el menú inicial se configuró de manera muy básica, simplemente para comprobar cómo funcionaba la librería PyGame. Se incluyeron 3 botones con un fondo gris a la izquierda de la pantalla: *Empezar,* que daba paso a la Ilustración 3.2: Ventana de juego de la segunda iteración, *Tutorial* e *Info del Creador*, que inicialmente estaban vacíos ambos.

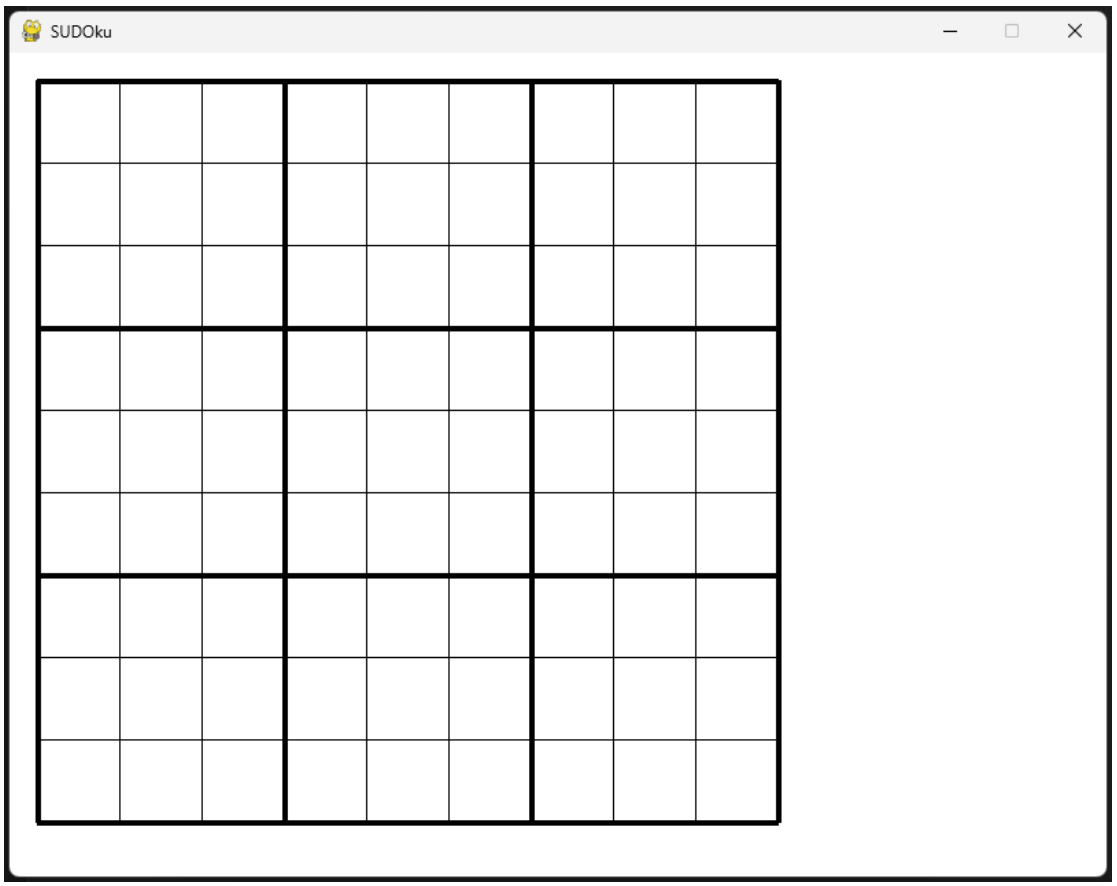


Ilustración 3.2: Ventana de juego de la segunda iteración

Al seleccionar *Empezar*, se dibujaba un tablero de Sudoku completamente vacío en pantalla. En ese momento no se disponía de ningún otro botón y no estaba implementada la interacción con el tablero mediante teclado y ratón.

3.3 Tercera iteración: Interacción con el tablero

En esta iteración el objetivo principal era conseguir la interacción del jugador con el tablero mediante el teclado y ratón y una mejora de la ventana de juego.

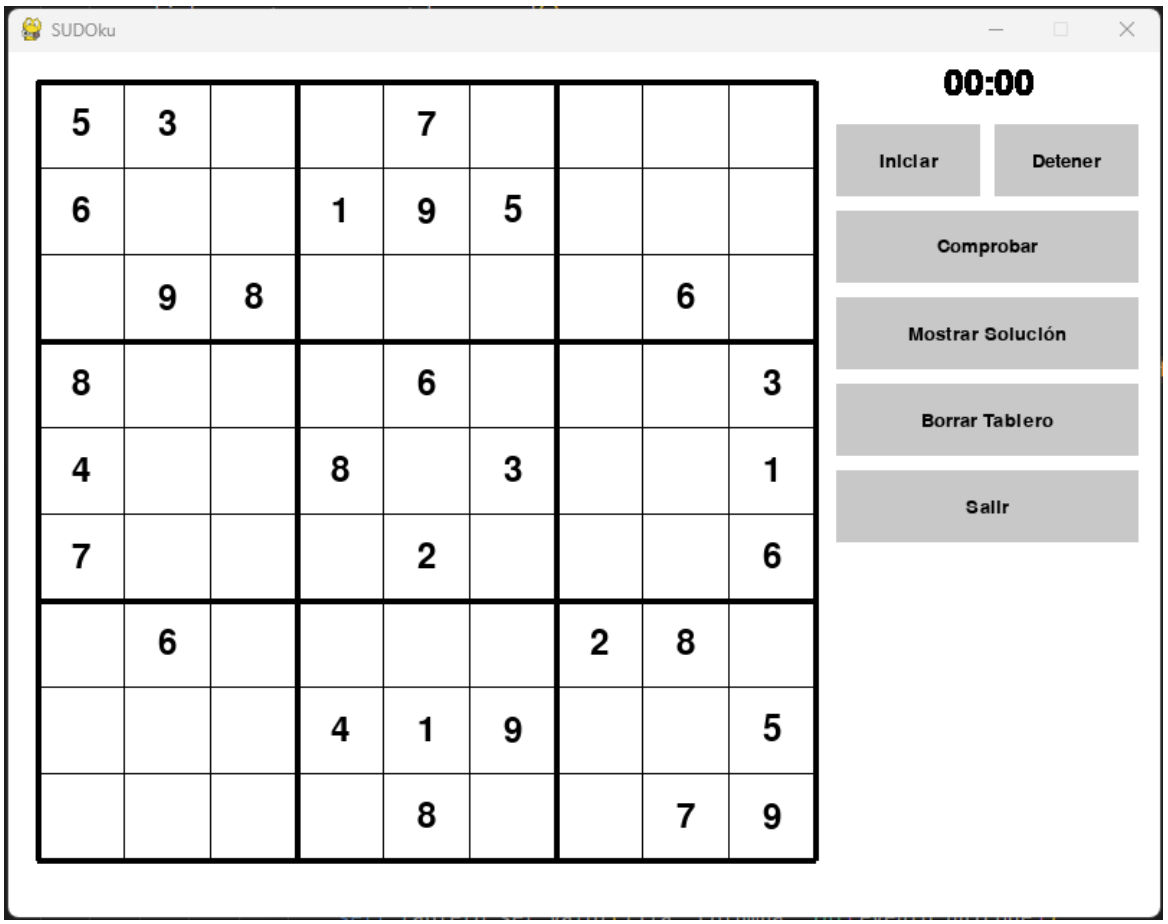


Ilustración 3.3: Ventana de juego de la tercera iteración

De esta manera, se añadieron 6 nuevos botones en la parte derecha de la interfaz (Ilustración 3.3):

1. **Iniciar**: este botón permitía al usuario interactuar con el temporizador, al accionarlo, el jugador podía iniciar el temporizador para que empezase a contar el tiempo.
2. **Detener**: este botón, al igual que el anterior, permitía al usuario interactuar con el temporizador, esta vez deteniendo el temporizador por completo.
3. **Comprobar**: al accionarlo permitía comprobar la solución introducida al tablero de Sudoku, indicando con dos colores, morado para celdas con números incorrectos o vacíos y verde para celdas con números correctos. Ejemplo en la Ilustración 3.5: Ejemplo de valores insertados comprobados en la tercera iteración.
4. **Mostrar solución**: este botón permitía al usuario mostrar la solución correcta en el tablero.

5. **Borrar tablero**: al accionar este botón, todas las celdas que no formasen parte de las pistas iniciales se quedaban vacías, permitiendo al usuario empezar de nuevo si se atascaba o simplemente quería intentarlo de nuevo.
6. **Salir**: este botón permitía salir del programa.

Además de los botones, se introdujo el temporizador en pantalla, que en ese momento presentaba algunos fallos visuales de superposición, corregidos posteriormente. También se añadió la representación gráfica de los números del tablero inicial con pistas, de manera que el usuario pudiese ver correctamente el juego.

Respecto a la interacción con el propio tablero, se permitió que el usuario pudiese seleccionar una celda e introducir un número, siempre que se encontrase en el rango [1, 9], de lo contrario no se introduciría. Todo esto se mostraba visualmente mediante colores que indicaban la celda seleccionada y las adyancentes, como se puede ver en la Ilustración 3.4: Celdas seleccionadas en la tercera iteración. También se modificó el color del número insertado, que pasaba de ser negro a rojo si el número interfería con alguna regla del juego, como la no repetición de números en la misma sub-matriz 3x3 o en la misma fila o columna.

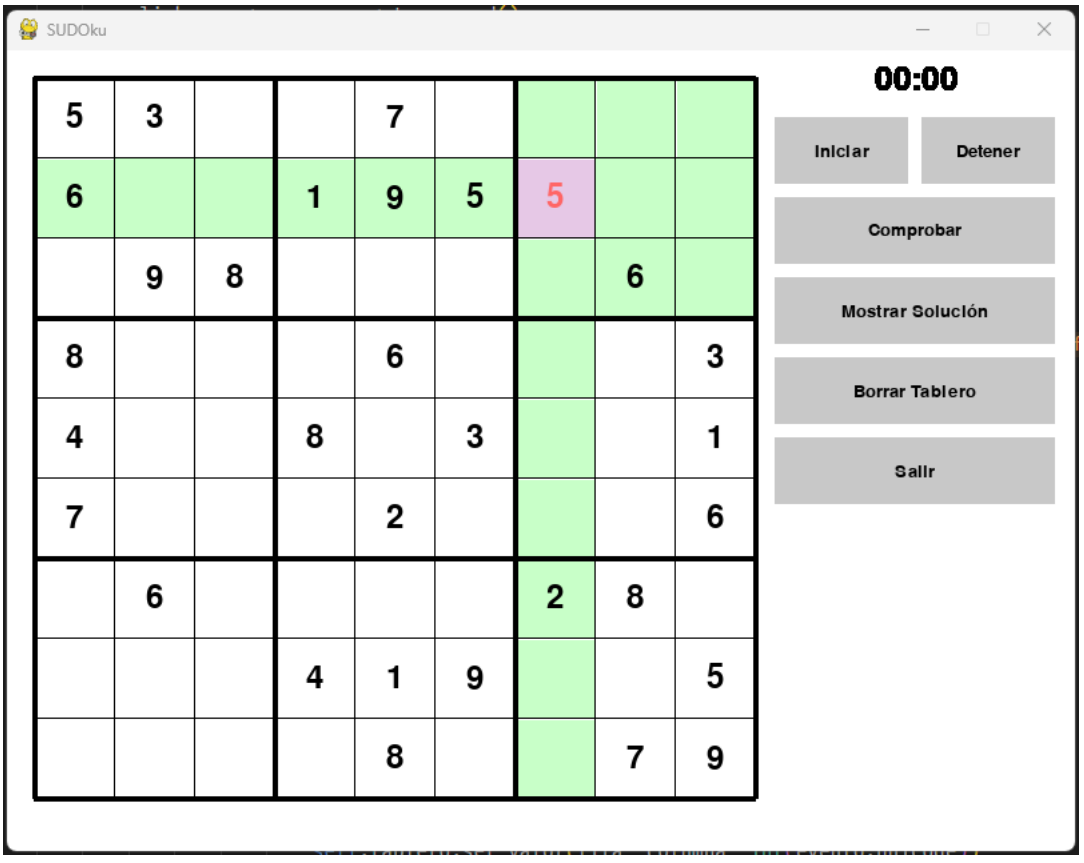


Ilustración 3.4: Celdas seleccionadas en la tercera iteración

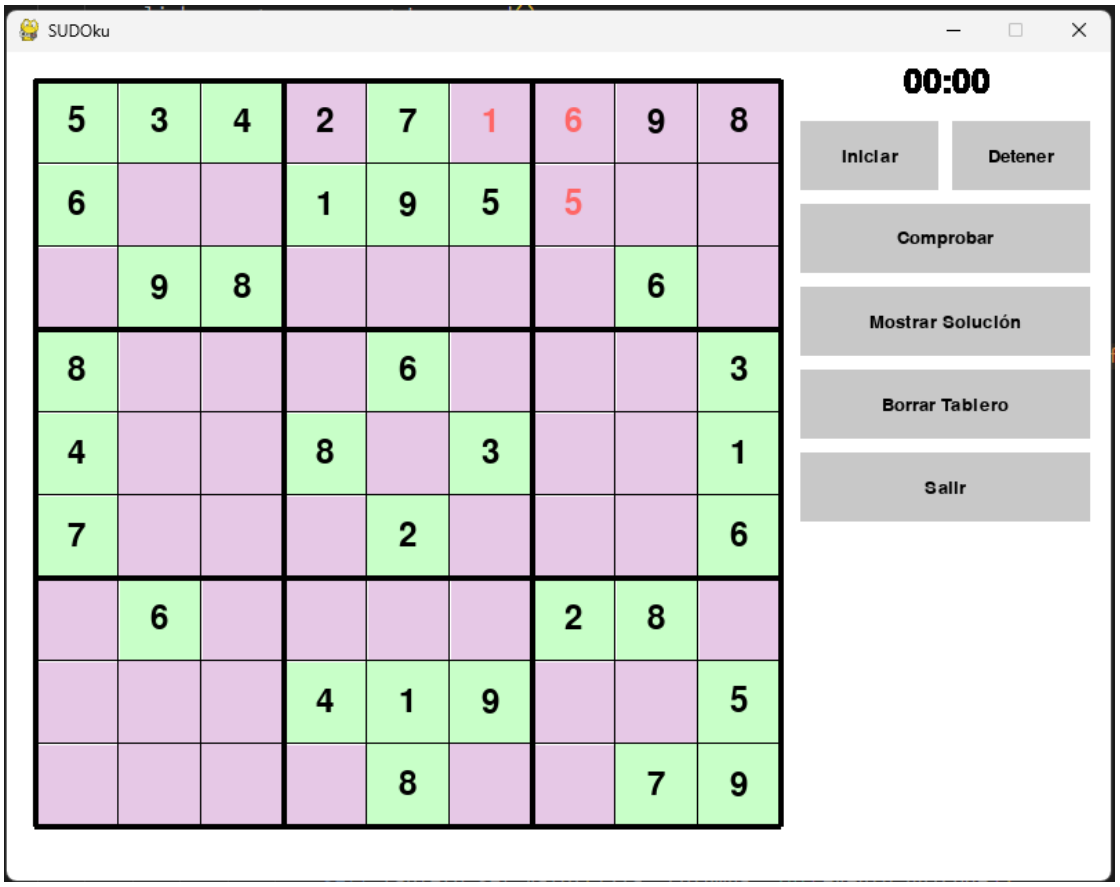


Ilustración 3.5: Ejemplo de valores insertados comprobados en la tercera iteración

3.4 Cuarta iteración: Creación de nivel fácil, medio y temporizador

Esta iteración se centró en 2 objetivos diferenciados: crear Sudokus pseudo-aleatorios de nivel fácil o medio y la corrección de errores del temporizador.

Para el primer objetivo se desarrollaron diversas funciones: primero se hicieron las que permitían resolver cualquier Sudoku de nivel fácil y medio, luego se hicieron las que creaban un tablero de Sudoku resuelto que cumpliese con todas las reglas y por último se unieron para crear Sudokus de esa dificultad, de manera que se seguía el siguiente flujo:

1. Primero se genera un tablero resuelto, tomando como base los números del 1 al 9 reordenados pseudo-aleatoriamente. Para generarlo se usa un algoritmo de *backtracking*, se selecciona un número de la lista reordenada y se introduce en la casilla. Posteriormente se comprueba que se cumplan las reglas, y si no

interfiere se pasa al siguiente. En caso de interferir, va hacia atrás y prueba otro número.

2. Cuando ya se tiene el tablero resuelto, se empiezan a eliminar pistas, a medida que se eliminan las pistas se comprueba si el tablero de Sudoku es resoluble en su forma actual, si deja de ser resoluble en algún punto se vuelve al estado inmediatamente anterior y se finaliza el proceso.
3. Si el nivel es fácil, del Sudoku que se devuelve se añadirán 7 pistas más, para simplificar el nivel aún más.

Algo importante de este proceso es que al finalizar la creación del Sudoku, se devuelve tanto el Sudoku preparado para jugar, como la solución. De esa manera se puede comprobar o mostrar posteriormente la solución de manera inmediata, con complejidad constante.

3.5 Quinta iteración: Mejoras visuales y selector de dificultad

En esta iteración el objetivo fue realizar una serie de mejoras visuales para ir analizando alternativas de formas y colores para los botones, y crear un selector de dificultad que permita al usuario elegir la dificultad deseada y que se genere un Sudoku.

Para los cambios visuales se modificaron los colores de los botones, se añadieron sombras, se cambió el tipo de letra para que encajase con el diseño minimalista que se pretendía (Ilustración 3.7) y se añadieron figuras representativas para los botones del temporizador (Ilustración 3.6).

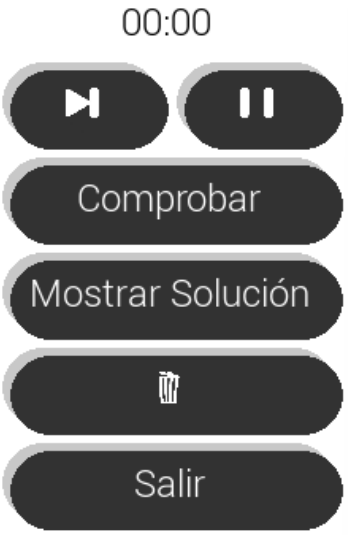


Ilustración 3.6: Mejoras de los botones y temporizador en la quinta iteración

	6		1				5	2
			7			4		3
					4	9		
	7				5	2	1	
5	2						9	
3		1				6	8	
		4		1	8		3	
6								
2				9				

Ilustración 3.7: Mejora de fuente de letra en tablero en la quinta iteración

Para completar el objetivo de la selección de dificultad, se añadió una pantalla extra al pulsar *Empezar*, que permitiese al usuario elegir entre las 3 dificultades posibles que tiene el juego (Ilustración 3.8).

Ilustración 3.8: Selector de dificultad en la quinta iteración

Una vez seleccionada la dificultad, se muestra al usuario una pantalla de carga simple mientras se crea el Sudoku de la dificultad deseada (Ilustración 3.9). La duración total de la pantalla de carga es de unos 3 segundos.

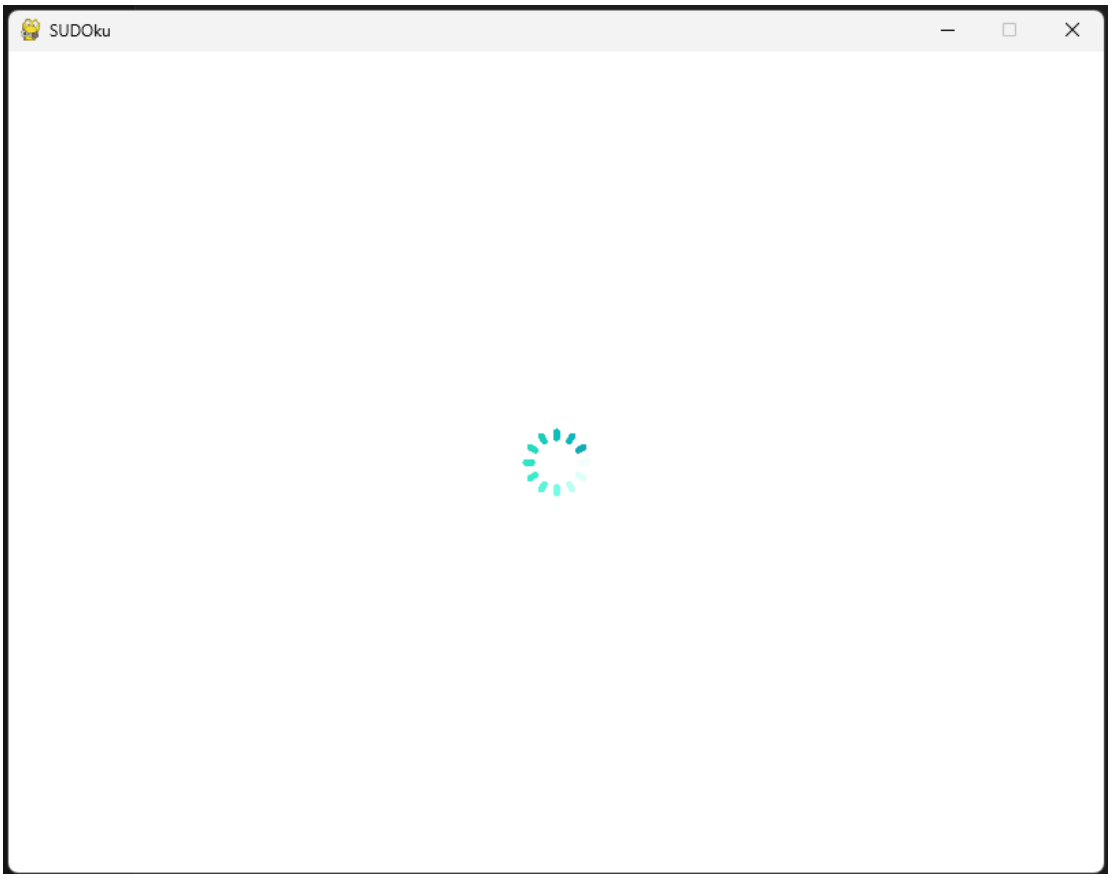


Ilustración 3.9: Pantalla de carga de juego en la quinta iteración

3.6 Sexta iteración: Puntuación y mejoras visuales

En esta iteración los objetivos fueron los siguientes:

- Realizar otra vuelta de mejoras visuales en botones, fondo, selector de dificultad y colores.
- Guardado de puntuación al finalizar una partida.
- Añadir botones en el menú inicial para *Ajustes* y *Puntuación.*
- Añadir un mensaje de enhorabuena al jugador cuando consiga resolver un Sudoku en el que se muestre la dificultad jugada y el tiempo en el que se ha conseguido.
- Añadir pantalla de puntuación que muestre las mejores puntuaciones del jugador para cada nivel, ordenadas de mejor a peor puntuación.

Para conseguir todos los objetivos, se empezó realizando las mejoras gráficas oportunas, se modificaron los colores de los botones para que fuesen más amigables con la interfaz, se añadió un fondo de pantalla para el juego y se añadieron los botones de *Ajustes* y *Puntuación* como se puede ver en la Ilustración 3.10.

Ilustración 3.10: Menú inicial del juego en la sexta iteración

A continuación, se implementó la ventana de puntuación, la cual tenía un diseño básico que luego acabó siendo mejorado en iteraciones posteriores. En esta ventana se mostraban las mejores puntuaciones para cada nivel en formato tabla y se incluyó un botón para volver al menú inicial (Ilustración 3.11).

Ilustración 3.11: Ventana de puntuaciones en la sexta iteración

Además, se realizaron cambios de diseño en el selector de dificultad, alineándolo con las pantallas anteriores y la estética que estaba tomando el juego en general (Ilustración 3.12).

Ilustración 3.12: Selector de dificultad en la sexta iteración

Por último, se añadió un mensaje al resolver el Sudoku y acabar la partida (Ilustración 3.13).

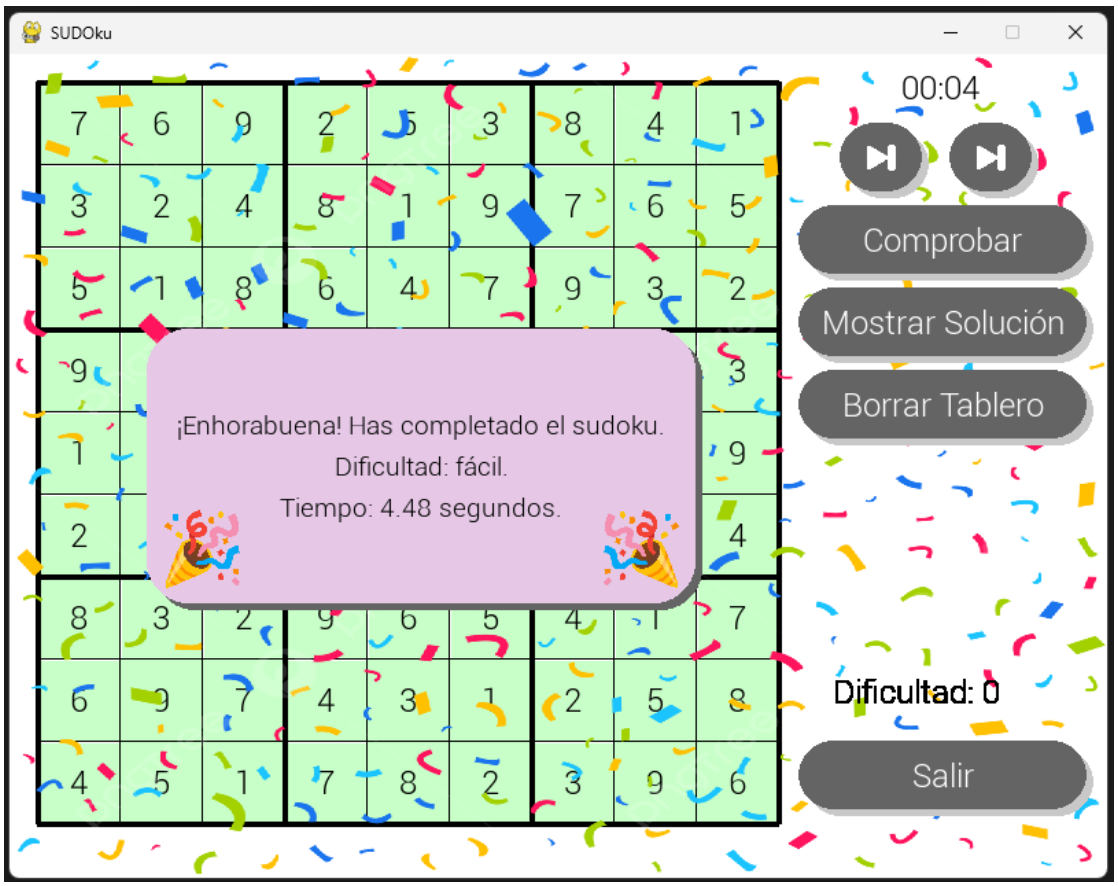


Ilustración 3.13: Mensaje de enhorabuena mostrado al usuario en la sexta iteración

3.7 Séptima iteración: Logo, ajustes y mejoras estéticas

En esta iteración los objetivos eran en su mayoría meramente estéticos, como añadir un logo al programa, mejorar la ventana de puntuaciones o añadir una estadística del promedio por cada nivel. Como único objetivo no estético estaba el crear la ventana de ajustes y añadir algo de configuración para el usuario en esta ventana.

Primero se realizó el cambio de logo del juego. Para ello, se diseñó un logo simple usando (Canva, 2024) y se añadió como logo al programa (Ilustración 3.14).

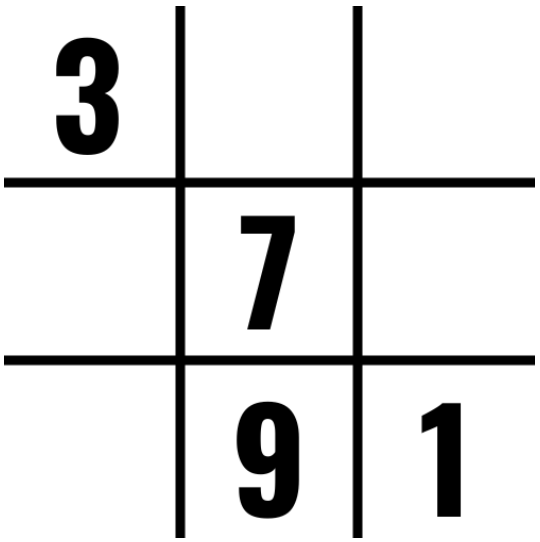


Ilustración 3.14: Logo del juego

En la Ilustración 3.15 podemos ver cómo queda en la pantalla, junto al nombre provisional del juego.

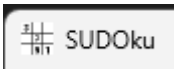


Ilustración 3.15: Logo y nombre en la séptima iteración

Tras la modificación del logo, se realizaron mejoras en la ventana de puntuación. Para ello, se añadió una estadística de promedio basada en todas las puntuaciones recogidas, se redujo el número de puntuaciones mostradas de 7 a 6 para equilibrar la pantalla, se añadió una tabla recubriendo los datos, se modificó el tipo de fuente usada para los títulos y por último, se añadieron emojis de medallas de oro, plata y bronce en las primeras tres puntuaciones de cada nivel (Ilustración 3.16).

Ilustración 3.16: Ventana de puntuaciones en la séptima iteración

A continuación, se procedió a añadir la ventana de Ajustes, inicialmente con sólo un parámetro configurable por el usuario, enlazado al archivo de configuración, que permitía modificar el modo de accesibilidad deseado. El modo de accesibilidad tiene como objetivo modificar los colores de la selección de casillas durante el juego y proporcionar una leyenda para evitar que el usuario tenga que guiarse únicamente por los colores sin una explicación de su significado y uso. Esta ventana permite, además de modificar el modo de accesibilidad, mostrar el modo actual para no confundir al usuario entre distintos modos posibles. De esta manera, se mostrará mediante texto si el modo accesible está activado o desactivado (Ilustración 3.17).

Ilustración 3.17: Ventana de ajustes en la séptima iteración

3.8 Octava iteración: Versión 1 de accesibilidad y nivel difícil

Esta iteración tiene dos objetivos claramente definidos:

1. **Accesibilidad**: para conseguir los objetivos de accesibilidad propuestos hay que analizar cada color de la partida de Sudoku y realizar modificaciones para que se entiendan mejor para los jugadores que lo necesiten (los colores de botones y menú se actualizarán en la siguiente iteración).
2. **Nivel difícil**: además de los niveles fácil y medio, se añadirá un nivel difícil parametrizable por el usuario por si desea una mayor dificultad o un nivel difícil que no sea tan complicado.

Para la accesibilidad vamos a analizar el estado inicial. Si nos fijamos en la Ilustración 3.5: Ejemplo de valores insertados comprobados en la tercera iteración, podemos ver que el color seleccionado para el error es el rojo, pero no se le indica al usuario en ningún momento qué significa ese color, al igual que no se indica en ningún momento qué significa el morado al seleccionar una celda y las celdas con verde adyacentes. Esto presenta una problemática según (Web Accessibility Initiative, 2019), por lo que se modificarán de la siguiente manera:

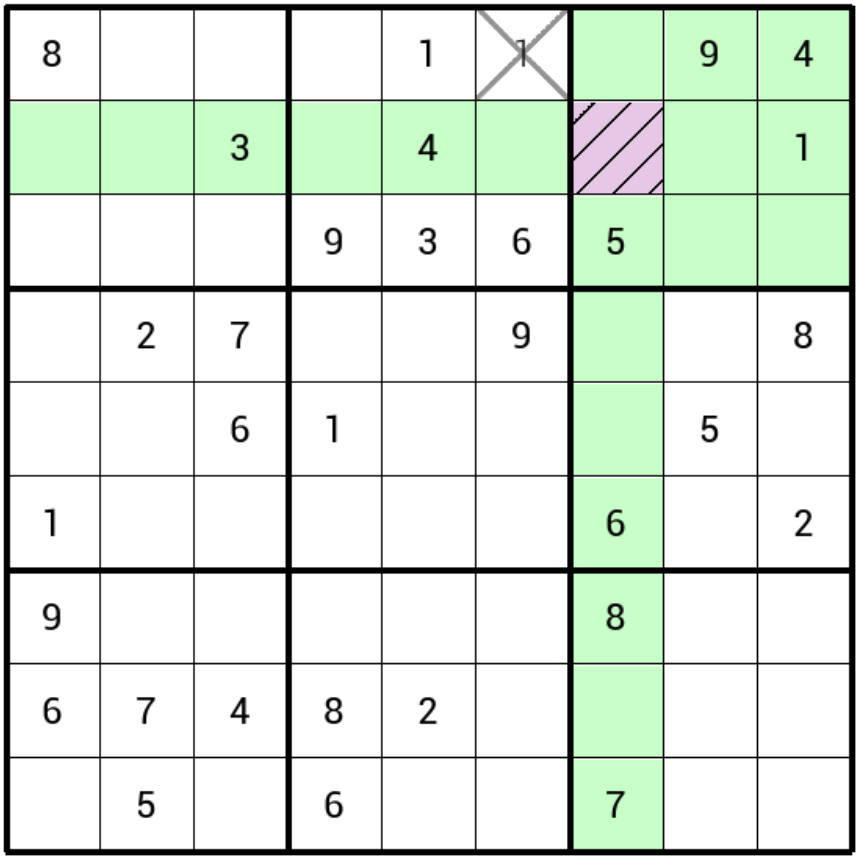


Ilustración 3.18: Juego accesible en la octava iteración

1. Se ha modificado la pista visual para valor erróneo, de tal forma que en vez mostrarse el número en rojo, lo cual puede ser un problema para personas con cualquier tipo de ceguera de color, se mostrará una cruz que cubrirá la celda por completo (ver fila 1, columna 6) en la Ilustración 3.18.
2. En el caso de que la persona que juega no pueda ver correctamente las diferencias entre el morado (celda seleccionada) y el verde (celda adyacente), se ha modificado el fondo de la celda seleccionada, añadiendo líneas diagonales que ocupan totalmente la celda (ver fila 2, columna 7) en la Ilustración 3.18.
3. El color de las celdas adyacentes se ha mantenido en verde, ya que no interfiere con los apartados anteriores que se han modificado.

Además de la modificación de colores y selección de celdas, se ha añadido una pequeña leyenda en pantalla para que el jugador no se sienta perdido en ningún momento por los símbolos y colores, la cual se puede ver en la Ilustración 3.19.

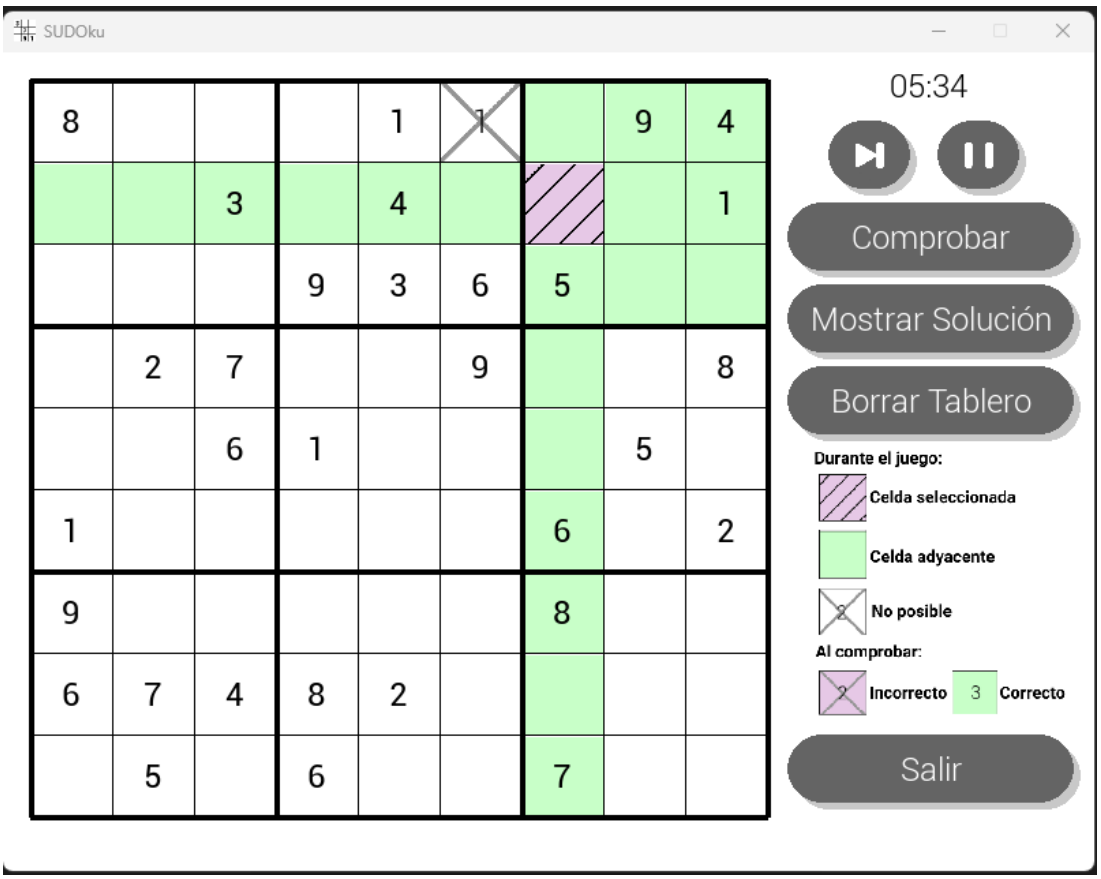


Ilustración 3.19: Ventana de juego con leyenda para accesibilidad

En cuanto al nivel difícil, lo primero a realizar fue modificar la pantalla de ajustes, permitiendo al usuario modificar la dificultad del nivel difícil. Para ello, se realizó una separación con los ajustes de accesibilidad y se añadieron ajustes para dificultad, incluyendo un modificador sencillo (Ilustración 3.20).

Ilustración 3.20: Pantalla de ajustes en la octava iteración

Como se puede ver en la Ilustración 3.20, se ha añadido una línea separadora y un modificador de dificultad, permitiendo al usuario mover el nivel en el rango [1, 7]. Cuando el jugador selecciona un nivel por encima de 5, se incluye un aviso que el usuario deberá aceptar informando de que si se sube más de 5 la dificultad puede ocasionar la creación de sudokus con múltiples soluciones (Ilustración 3.21).

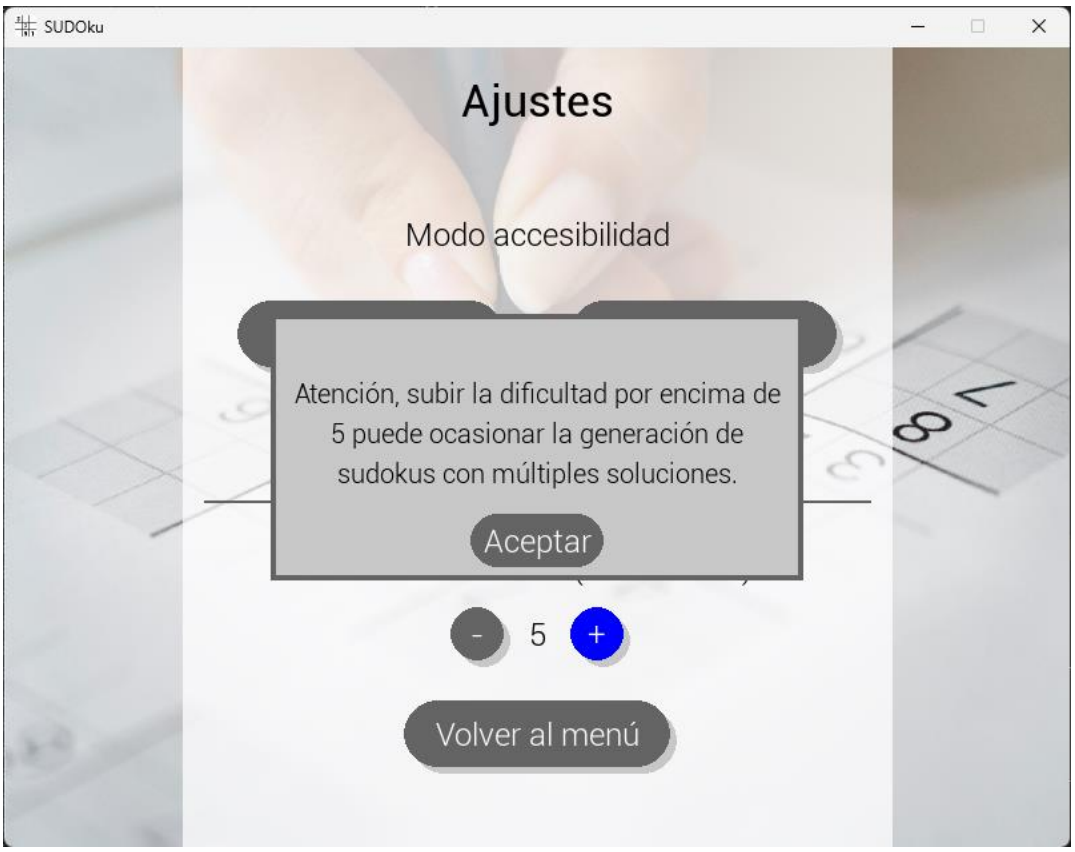


Ilustración 3.21: Aviso de modificación de dificultad en ajustes

Por último, para crear el nivel difícil se toma de punto inicial el Sudoku nivel medio, al que se le eliminarán n pistas, siendo n el número parametrizable por el usuario, pasando de una media de 31 pistas a una media de 24 pistas según el nivel que elija el jugador. De esta manera, se garantiza que al menos en una ocasión, el usuario deberá realizar un proceso de *backtracking* para resolver el Sudoku.

Sintetizando la creación de los distintos niveles de dificultad:

- **Nivel fácil**: es un Sudoku medio al que se le han añadido 7 pistas extra.
- **Nivel medio**: es un Sudoku con el mínimo número de pistas para el que se puede resolver sin que el jugador encuentre un punto de decisión en el que todos los caminos son posibles. Es decir, todo el Sudoku se puede resolver realizando iteraciones donde se introducen los únicos números posibles en determinadas celdas, no se encuentra en ningún momento una situación en la que todas las casillas tienen más de un valor posible.
- **Nivel difícil**: es un Sudoku medio al que se le han quitado n pistas, de manera que se obliga al jugador a tomar una decisión en la que hay varios caminos igualmente viables y posibles, solo que únicamente uno llevará a la solución.

En este punto del juego, es necesario explicar las dos opciones de resolución de tableros que hay disponibles:

1. **Método clásico**: consiste en, según los valores posibles de cada celda, introducir únicamente los valores de las celdas que sólo tienen un número posible. Una vez introducidos todos los números únicos en las celdas correspondientes, se procede a realizar otra iteración, y así sucesivamente hasta resolver el Sudoku o llegar a un punto de decisión igualitaria (en el nivel difícil únicamente).

2. ***Backtracking***: todos los Sudokus se pueden resolver usando este método, pero solo será obligatorio usarlo en el nivel difícil. El método consiste en usar el método clásico hasta que el Sudoku deje de ser resoluble mediante ese método (se ha llegado a un punto de decisión igualitaria anteriormente mencionado). En ese momento realizar *backtracking* hasta resolver el tablero por completo.

La pseudo-aleatoriedad es aplicada en varias ocasiones a lo largo del juego, en todas las ocasiones haciendo uso de la biblioteca Random de Python:

1. **Creación de tableros llenos**: para crear los tableros se parte de una lista de valores, en este caso [1, 9] y se van introduciendo en las casillas libres de manera que no infrinjan las reglas básicas. De esta manera si no están ordenadas de manera pseudo-aleatoria cada vez que se genere un Sudoku nuevo, todos serían el mismo y además, no quedaría finalizado (Ilustración 3.22).

1	2	3	4	5	6	7	8	9
4	5	6	1	2	3			
7	8	9				1	2	3
2	1	4	3	6	5	8	7	
3	6	5	2	1	4	9		
8	7		9			2	1	4
5	3	1	6	4	2		9	7
6	4	2	5	3	1			8
9		7	8			3	4	1

Ilustración 3.22: Ejemplo de generación de Sudoku sin pseudo-aleatoriedad

Sin embargo, si partimos de una lista reordenada pseudo-aleatoriamente, en vez de partir de [1, 2, 3, 4, 5, 6, 7, 8, 9], partiríamos de una distinta en cada creación de Sudoku nuevo.

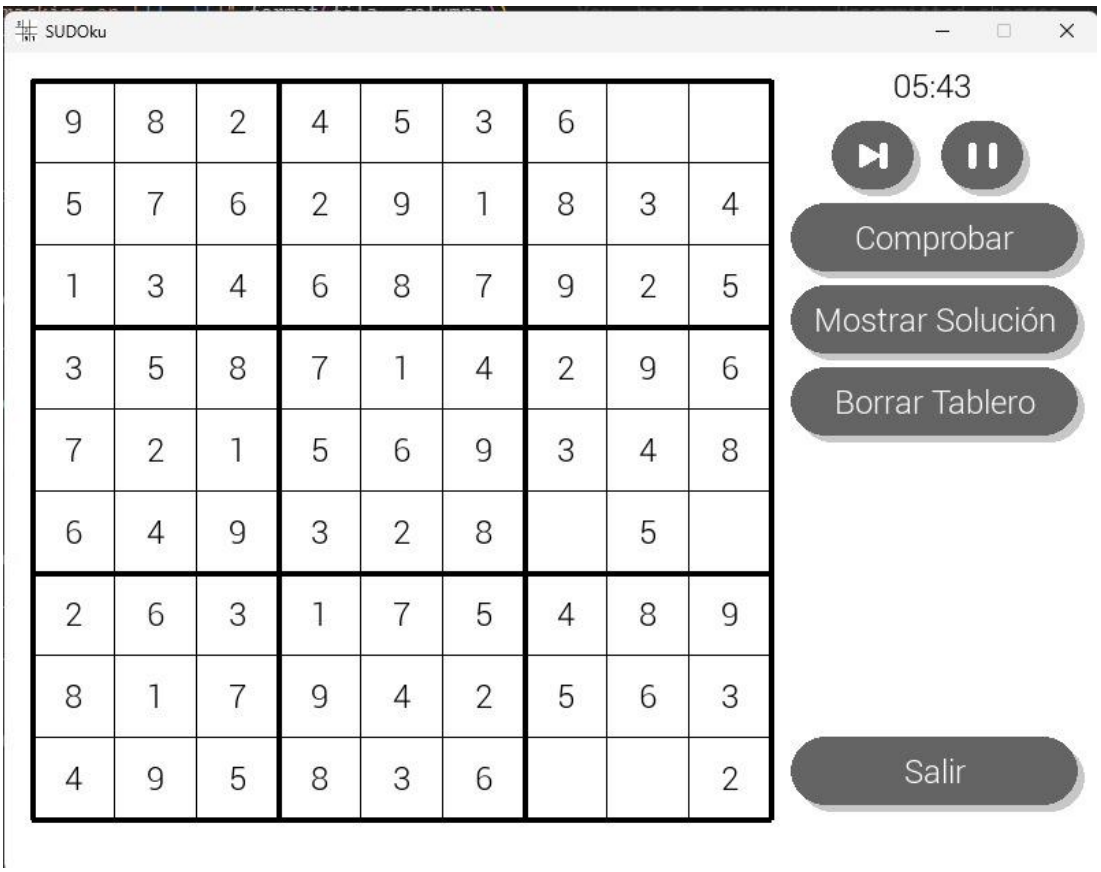
2. **Eliminación de números al crear tableros preparados para jugar**: partiendo de un tablero completamente relleno, es necesario obtener todas las casillas en una lista de celdas y reordenarlas para probar a eliminar números. Inicialmente tenemos la lista de celdas [[0,0], [0,1], [0,2], [0,3], [0,4], ... , [9,9]], probamos de manera iterativa a eliminar los números de esas celdas y comprobamos si sigue siendo resoluble el tablero en la forma actual, seguimos así hasta que intente eliminarlos todos y no pueda eliminar ningún número más. De esta manera siempre se intenta eliminar la casilla [0,0] y luego la [0,1], así sucesivamente hasta llegar a la [9,9], de forma que siempre sigue el mismo orden. Si se reordenan de forma pseudo-aleatoria, si partimos del mismo tablero relleno, nos proporcionará dos Sudokus distintos para resolver, ya que van eliminando casillas en órdenes distintos. Con esto se reduce prácticamente al máximo la posibilidad de encontrar dos tableros idénticos, uno seguido de otro.

Debido a la pseudo-aleatoriedad, en el nivel difícil es posible que se creen tableros que pueden ser resueltos mediante el método clásico por completo, debido a la eliminación pseudo-aleatoria de pistas. La probabilidad de que esto ocurra está indicada en la Tabla 3.1. El único aspecto negativo de esta probabilidad es que en los casos en los que ocurra, en vez de un Sudoku difícil el usuario tendrá un Sudoku medio, pero no afecta a ningún aspecto más, por lo que se considera un riesgo asumible con tal de permitir al usuario modificar sus partidas más difíciles.

Número de Sudokus difíciles generados	Resueltos con *Backtracking*	Resueltos con el método Clásico
100	12%	100%

Tabla 3.1: Proporción Sudokus difíciles resolubles con distintos métodos

En niveles difíciles de más de 5 puntos de dificultad puede darse el caso mostrado en la Ilustración 3.23 con muy baja probabilidad:



9	8	2	4	5	3	6		
5	7	6	2	9	1	8	3	4
1	3	4	6	8	7	9	2	5
3	5	8	7	1	4	2	9	6
7	2	1	5	6	9	3	4	8
6	4	9	3	2	8		5	
2	6	3	1	7	5	4	8	9
8	1	7	9	4	2	5	6	3
4	9	5	8	3	6			2

Ilustración 3.23: Ejemplo de Sudoku con múltiples soluciones

Como podemos ver en la Ilustración 3.23, el Sudoku tiene dos soluciones posibles igualmente válidas, puesto que en la fila 1, columna 8 puede introducirse un 1 o un 7 con la misma probabilidad y ambas opciones generarían un Sudoku resuelto igualmente válido.

3.9 Novena iteración: Versión 2 de accesibilidad, tutorial y créditos

En esta última iteración los objetivos eran los siguientes:

1. Completar los ajustes de accesibilidad, mejorar los contrastes en botones y colores de los distintos menús.
2. Añadir pantalla de *Créditos* con información relevante del proyecto.
3. Cambiar el título del proyecto y modificar el fondo en la ventana principal del menú.
4. Añadir un tutorial en formato diapositivas que realice un *tour* por el juego para nuevos jugadores.

Empezando por lo más sencillo, se modificó el fondo de la ventana principal del menú para hacer referencia al título del juego (Ilustración 3.24).

Ilustración 3.24: Nuevo nombre de juego en logo principal

Además se modificó el nombre de la ventana de juego, como se en la Ilustración 3.25.

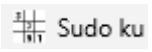


Ilustración 3.25: Logo y nombre en la novena iteración

Lo siguiente a realizar fue el tutorial en la ventana correspondiente. En las ilustraciones Ilustración 3.26 y Ilustración 3.27 se muestran las dos primeras páginas de tutorial para ver su funcionamiento. El resto de páginas se pueden encontrar en el apéndice 5.3.

Ilustración 3.26: Página 1 de tutorial

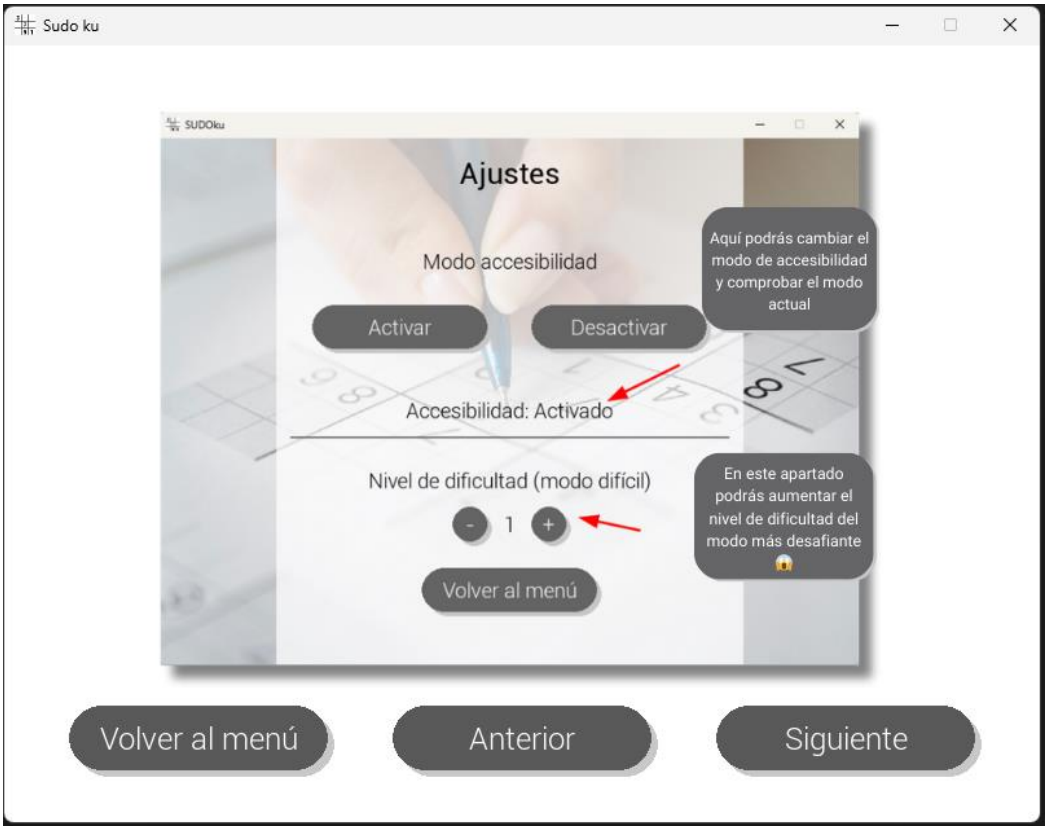


Ilustración 3.27: Página 2 de tutorial

También se modificó la ventana de créditos para que ahora mostrase más información del proyecto, incluyendo logos de las instituciones implicadas (Ilustración 3.28).

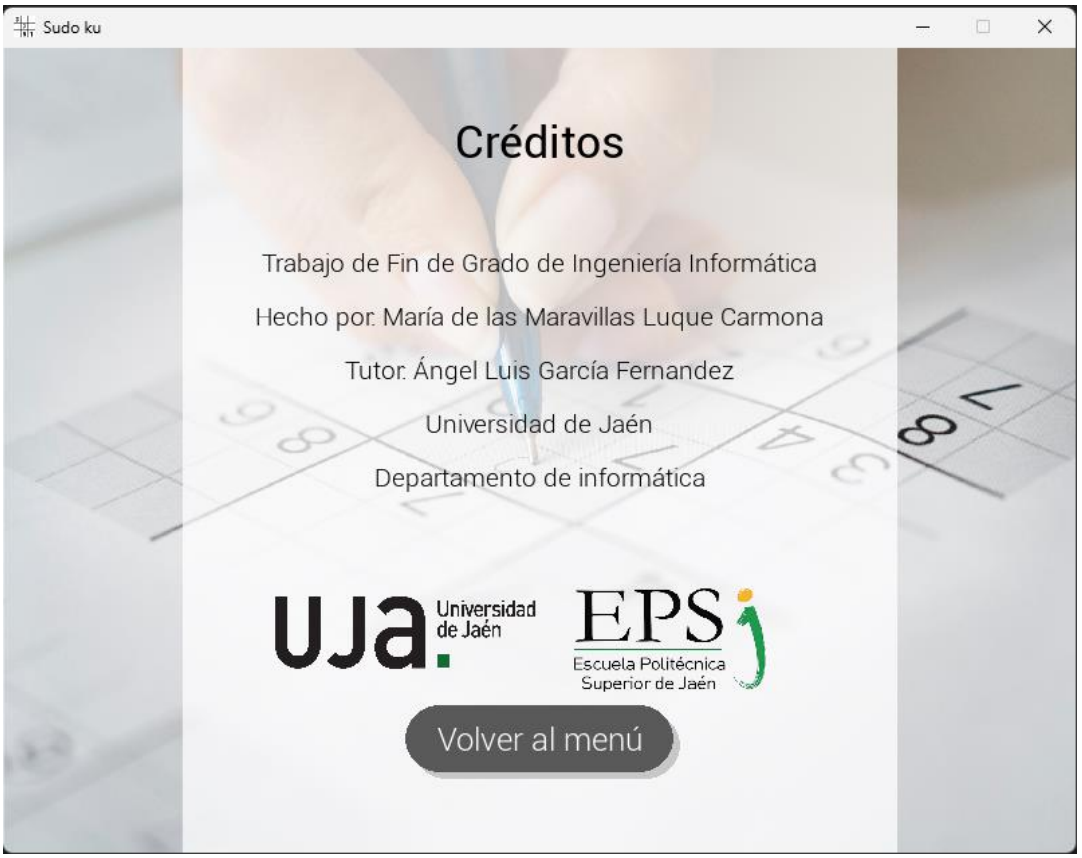


Ilustración 3.28: Ventana de Créditos en la novena iteración

Por último, se realizaron los últimos cambios respecto a accesibilidad, empezando por cambiar el color de los botones para que existiese más contraste entre la letra y el fondo, incluyendo cambios en el fondo y color de letra cuando se posiciona el ratón encima. La evolución de la apariencia de estos controles se ve en la Tabla 3.2.

Modo	Quinta iteración	Octava iteración	Novena iteración
Normal	Empezar	Empezar	Empezar
Ratón encima del botón	Empezar	Empezar	Empezar

Tabla 3.2: Comparativa avance de botones entre iteraciones

3.10 Pruebas finales

Las pruebas finales se van a desarrollar analizando varios elementos, indicados a continuación.

1. **Pruebas de verificación del sistema**, en las que se realizarán 2 pruebas a 2 usuarios externos al proyecto para verificar que el sistema se ha completado con éxito.
2. **Validación de los requisitos,** en las que se analizará si se ha cumplido con todos los requisitos definidos al inicio del proyecto, corroborando su aplicación uno a uno.
3. **Validación de la accesibilidad del sistema,** en las que se verá si se cumple con los estándares mínimos de accesibilidad definidos al principio del proyecto.

3.10.1 Pruebas de verificación del sistema

Para este apartado de pruebas se usarán como referencia pruebas de 2 usuarios distintos ajenos al desarrollo del juego. Se plantearán diversas preguntas y se analizarán sus respuestas para verificar que el producto desarrollado es completo y simple de utilizar, además de comprobar que no exista ningún problema no contemplado hasta este momento.

Preguntas realizadas:

1. ¿Sabes qué es un Sudoku?
2. ¿Sabes resolver Sudokus?
3. ¿Qué es lo primero que harías si quieres jugar una partida?
4. ¿Entiendes todo lo que puedes hacer en las diversas pantallas?
5. ¿Crees que la interfaz es cómoda y fácil de entender?
6. ¿Crees que la dificultad es adecuada?, ¿Por qué?

3.10.1.1 Primera prueba con usuarios

Al principio de la prueba se realizaron las preguntas pertinentes. A continuación se detallan las respuestas que se dieron y el análisis posterior.

1. **¿Sabes qué es un Sudoku?** Sí.
2. **¿Sabes resolver Sudokus**? Sí, cuando era pequeño hacía los Sudokus que venían en el periódico.

3. ¿**Qué es lo primero que harías si quieres jugar una partida**? Lo primero que hago cuando descargo un juego es entrar en *Ajustes* y miro toda la configuración para ver si puedo modificar algo y ponerlo a mi gusto. Tras eso paso directamente a jugar, en este caso pulsaría *Empezar.*
4. ¿**Entiendes todo lo que puedes hacer en las diversas pantallas**? Sí, en todas las pantallas entiendo lo que puedo hacer.
5. ¿**Crees que la interfaz es cómoda y fácil de entender**? Sí, es sencilla y no te pierdes por los botones, sabes en todo momento en qué punto estás, cómo volver al inicio o cómo empezar una partida.
6. ¿**Crees que la dificultad es adecuada?, ¿Por qué**? Sí, creo que es adecuada. En todo momento he podido tomar decisiones de una manera razonadamente adecuada a la dificultad que he elegido.

Este usuario entrevistado ya tenía experiencia previa en resolución de Sudokus de manera ocasional, por lo que la dificultad le ha parecido adecuada y ha agradecido poder aumentar la dificultad a su antojo en el nivel más difícil. No ha encontrado ningún problema con la interfaz y lo ha entendido todo a la perfección.

3.10.1.2 Segunda prueba con usuarios

Al igual que en la prueba anterior, al principio de la prueba se realizaron las preguntas pertinentes. A continuación se detallan las respuestas que se dieron y el análisis posterior.

1. **¿Sabes qué es un Sudoku?** Sí.
2. **¿Sabes resolver Sudokus**? Sí aunque soy muy mala para resolverlos, me llevan mucho tiempo.
3. ¿**Qué es lo primero que harías si quieres jugar una partida**? Darle a *Empezar.*
4. ¿**Entiendes todo lo que puedes hacer en las diversas pantallas**? Sí, lo entiendo.
5. ¿**Crees que la interfaz es cómoda y fácil de entender**? Sí, en los botones pone qué hace cada uno y se entienden perfectamente.

6. ¿**Crees que la dificultad es adecuada?, ¿Por qué**? Sí, lo que pasa es que yo soy muy mala haciendo Sudokus, pero dentro de lo que cabe ha sido fácil (el nivel fácil).

La segunda persona entrevistada no tenía mucha experiencia con Sudokus. Aunque conocía las reglas, alegaba que le costaba mucho tiempo resolverlos, así que únicamente ha probado el nivel fácil y medio, abandonando el difícil por falta de experiencia. La interfaz le ha parecido bonita, minimalista y simple y tampoco ha tenido ningún problema.

Por lo que se dan por concluidas las pruebas con usuarios como un éxito, ya que no han aparecido problemas y todo se ha entendido a la perfección en todo momento.

3.10.2 Validación de los requisitos

Para comprobar la validez del sistema, se hará referencia a todos los requisitos definidos en el apartado de Requisitos iniciales, analizando a su vez si se han cumplido todos y cada uno de ellos.

En vista de los resultados de las pruebas con usuarios, se demuestra que el sistema es capaz de crear una partida de Sudoku, seleccionando previamente la dificultad deseada por el jugador (**RF1**).

El juego permite la resolución de un Sudoku, permitiendo al usuario rellenar las casillas del tablero con el número deseado. También se proporciona la opción de comprobar si la solución es correcta o no (**RF2**).

En relación con (**RF1**) también, el Sudoku generado es pseudo-aleatorio de manera automática (**RF3**).

El programa valida el número introducido de forma instantánea e indica mediante colores o símbolos si es correcto o no, en función del cumplimiento de las reglas básicas del juego (**RF4**).

El juego muestra el tiempo transcurrido desde que se inicia la partida y el jugador tiene acceso al tablero de Sudoku. También se permite parar o reanudar el cronómetro (**RF5**).

El programa almacena la puntuación en un archivo CSV para después mostrarla en su ventana correspondiente (**RF6**), se puede ver en la Ilustración 3.29.

```
1   Dificultad,Puntuacion
2   media,1.4060392379760742
3   facil,26.71428918838501
4   media,7.259825706481934
5   dificil,383.300669670105
6   dificil,1262.404726266861
```

Ilustración 3.29: Contenido del archivo de puntuación tras varias partidas de prueba

Como se ha podido ver en las múltiples ilustraciones, el sistema tiene una interfaz gráfica para mostrar el tablero y permitir la interacción con el jugador (**RF7**).

El programa resuelve de manera automática e instantánea el Sudoku si se presiona el botón *Comprobar* (**RF8**).

Si analizamos los requisitos no funcionales, podemos ver también que se han cumplido todos de igual manera.

El programa genera tableros de Sudoku en menos de 4 segundos (**RNF1**), la interfaz es fácil de usar e intuitiva según demuestran las pruebas realizadas en usuarios en el apartado 3.10.1 Primera prueba con usuarios(**RNF2**). El sistema también está preparado para futuras mejoras y adiciones posteriores de funcionalidades sin comprometer el rendimiento; de hecho, está ubicado en un repositorio público que permite la interacción y las sugerencias de mejoras y correcciones de errores (**RNF3**). Por último, como se demostrará a continuación en el apartado Validación de la accesibilidad del sistema, el programa puede considerarse accesible en cuanto a colores e información visual (**RNF4**).

3.10.3 Validación de la accesibilidad del sistema

Puesto que la accesibilidad es un aspecto fundamental del diseño de todo el proyecto, se procede a enumerar las pruebas realizadas para garantizar que el producto final cumple con unos mínimos exigibles de accesibilidad para todos los usuarios.

3.10.3.1 Contraste de texto y color

En este apartado se analiza si se cumple con los requisitos de la referencia (Web Accessibility Initiative, 2019). Para ello, se usa una herramienta externa de Adobe (Adobe Color, 2024), en la que se han introducido los códigos de colores utilizados en el proyecto.

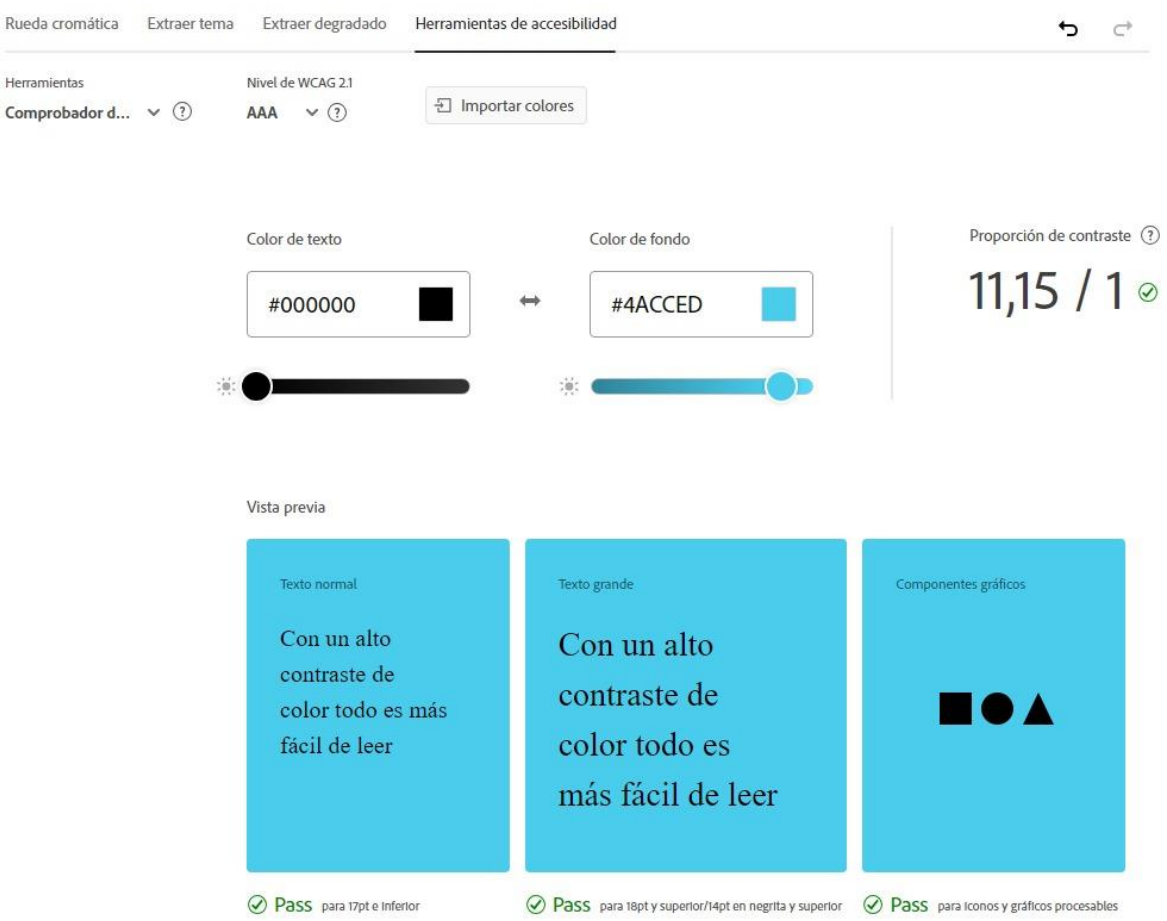


Ilustración 3.30: Comprobación de contraste de la primera combinación de colores

Como podemos ver en la Ilustración 3.30, la primera combinación de colores supera el contraste mínimo definido por WCAG, estableciéndose en 11,15 sobre 1, cumpliendo el nivel AAA de WCAG 2.1.

Ilustración 3.31: Comprobación de contraste de la segunda combinación de colores

Como podemos ver en la Ilustración 3.31, el segundo color también supera el contraste mínimo definido por WCAG, estableciéndose en 13,91 sobre 1, cumpliendo el nivel AAA de WCAG 2.1.

Ilustración 3.32: Comprobación de contraste de la tercera combinación de colores

La tercera combinación de colores también supera el contraste mínimo definido por WCAG, estableciéndose en 9,03 sobre 1, cumpliendo el nivel AAA de WCAG 2.1 (Ilustración 3.32).

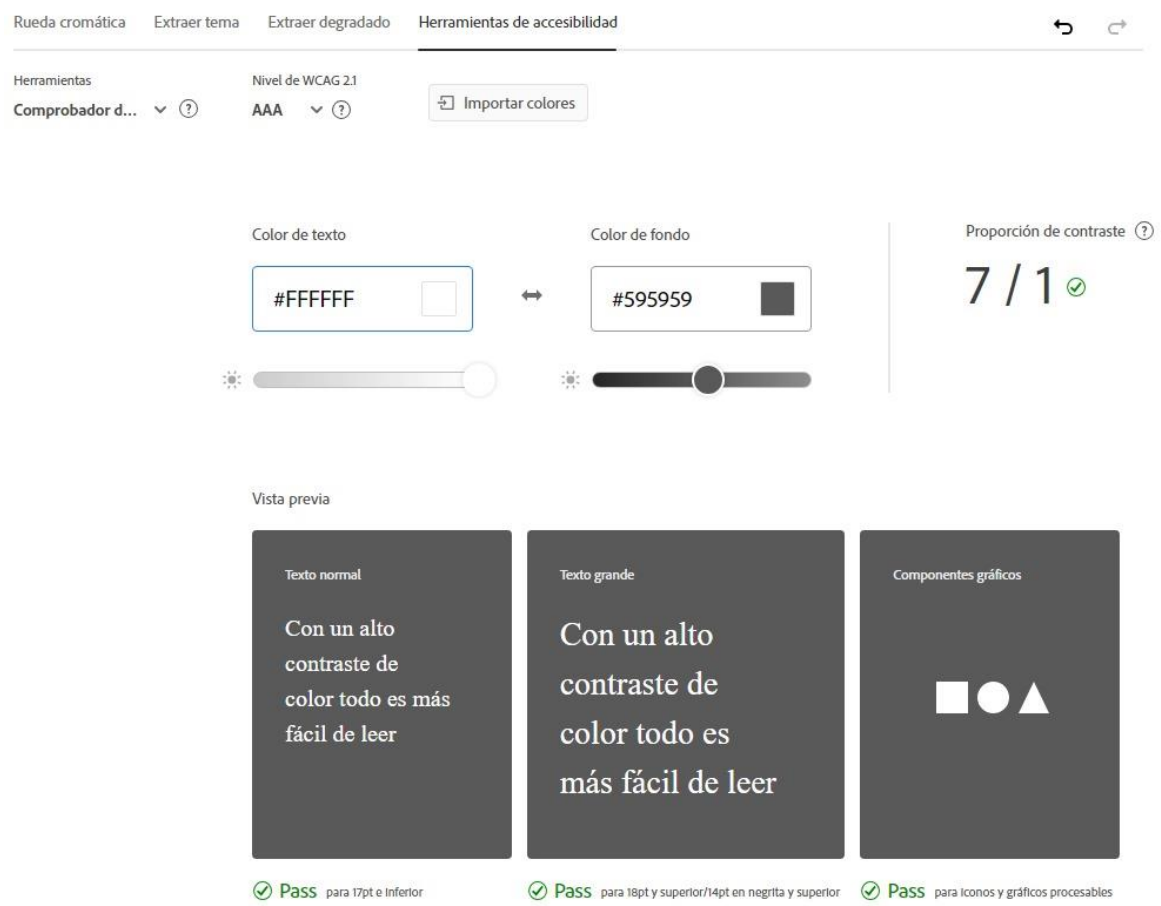


Ilustración 3.33: Comprobación de contraste de la cuarta combinación de colores

Por último, la cuarta combinación de colores (de los botones), también supera el contraste mínimo definido por WCAG, estableciéndose en 7 sobre 1, cumpliendo el nivel AAA de WCAG 2.1 (Ilustración 3.33).

Con todas las combinaciones de colores comprobadas, podemos afirmar que el contraste de colores definidos en el juego es el correcto para considerarse acesible.

3.10.3.2 Distinguibilidad de colores

En este apartado se comprobará que los colores sean totalmente distinguibles por personas con algún tipo de ceguera de color. Se hará uso una herramienta muy similar a la anterior, también desarrollada por Adobe (Adobe Color, 2024). Para ello bastará con introducir los códigos de los colores utilizados en los menús.

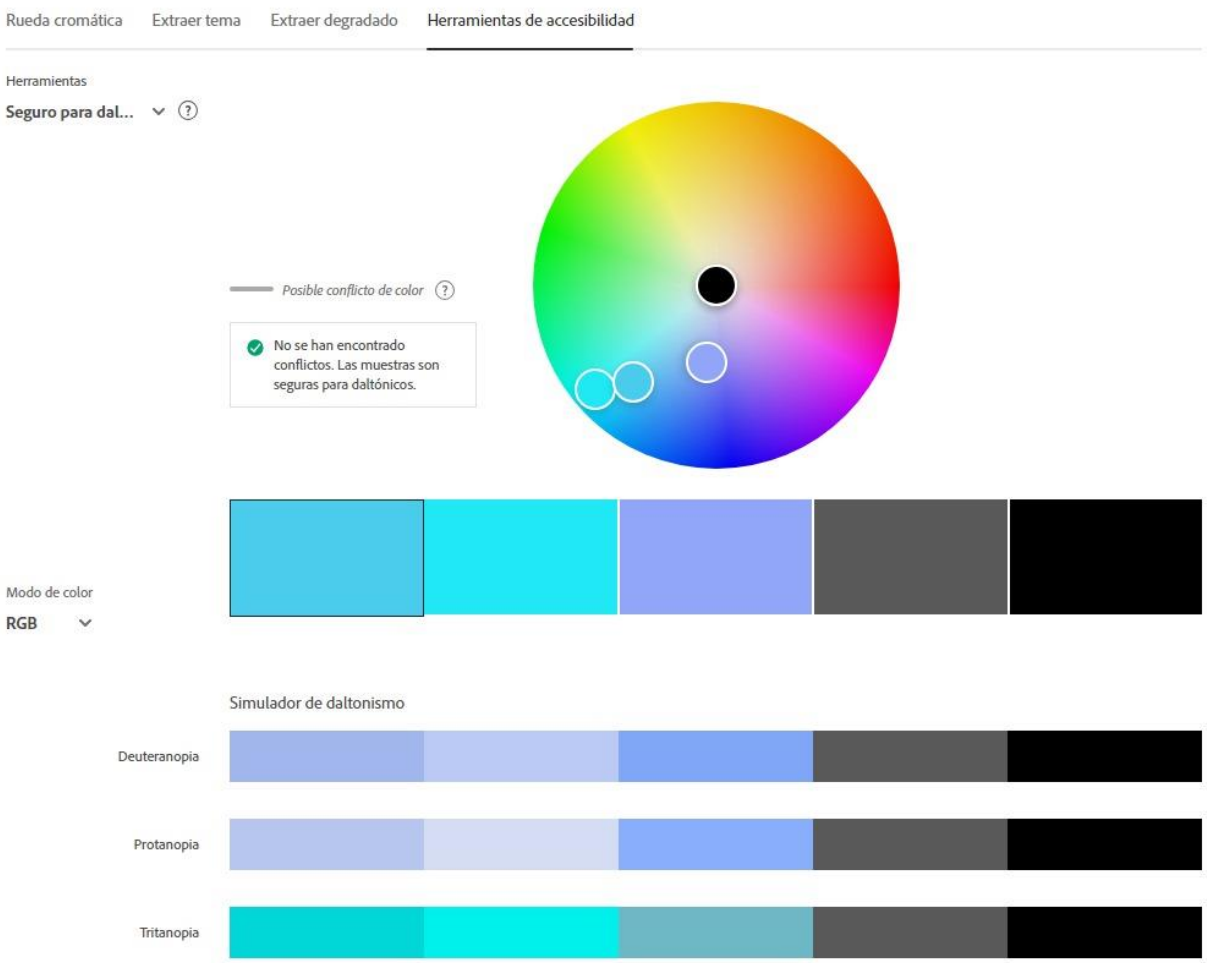


Ilustración 3.34: Comprobador de distinguibilidad de colores principales para daltonismo

Como se puede ver en la Ilustración 3.34, la herramienta considera que no existe ningún conflicto y que la muestra de colores es accesible para daltónicos.

En la Ilustración 3.35 se muestra el resultado para los colores del propio juego (celdas y números) también.

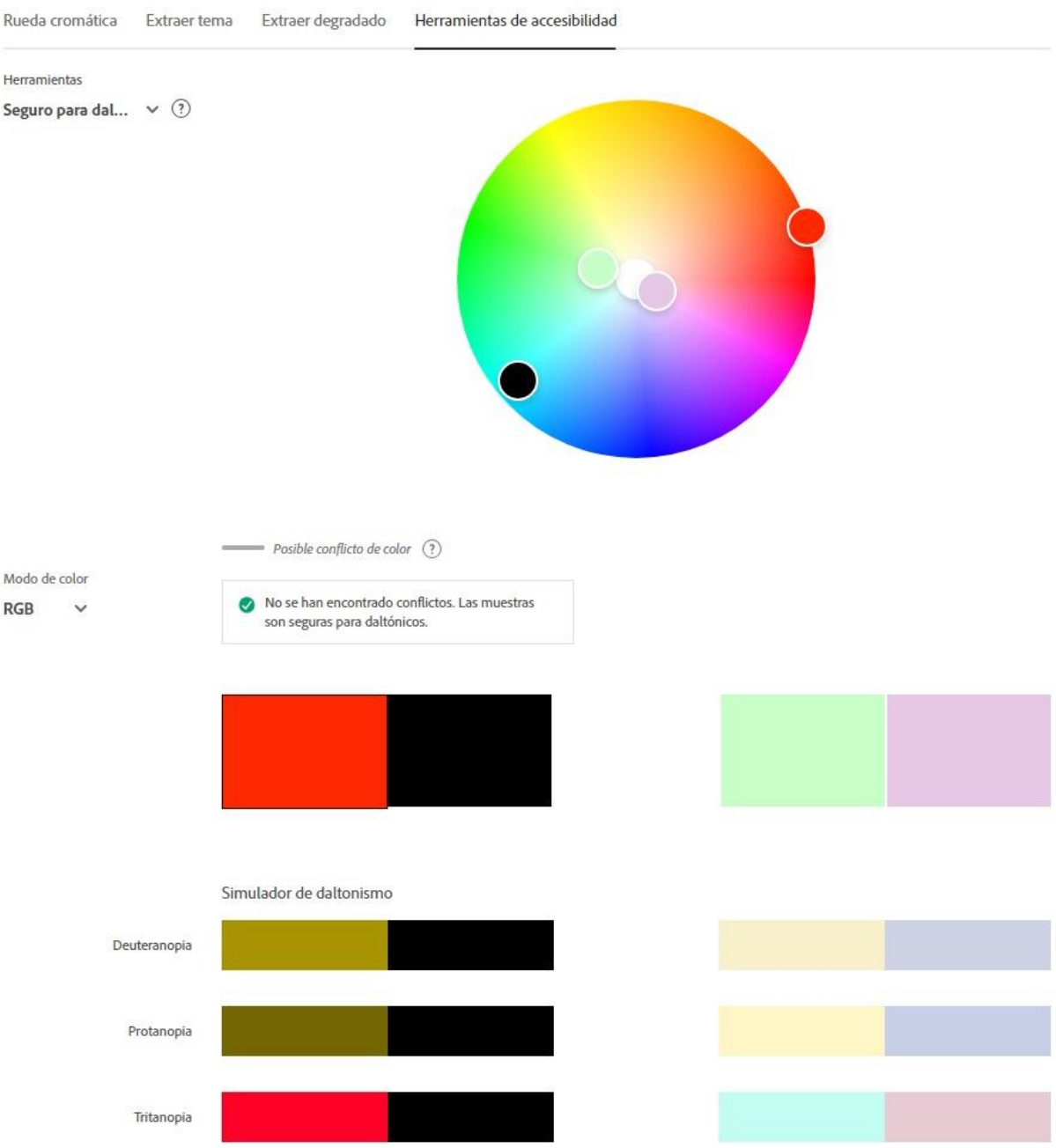


Ilustración 3.35: Comprobador de distinguibilidad de colores de juego para daltonismo

Al igual que la muestra de color anterior, esta muestra también es considerada segura para personas con daltonismo.

Por último, para finalizar estas pruebas, se decidió introducir una captura de pantalla durante el juego en un simulador de daltonismo para vivir en primera persona cómo se juega con ese tipo de condición (Colorlitelens.com, 2024). Se puede ver en las ilustraciones Ilustración 3.36 a Ilustración 3.37, que los colores y las formas son totalmente distinguibles en cualquier caso.

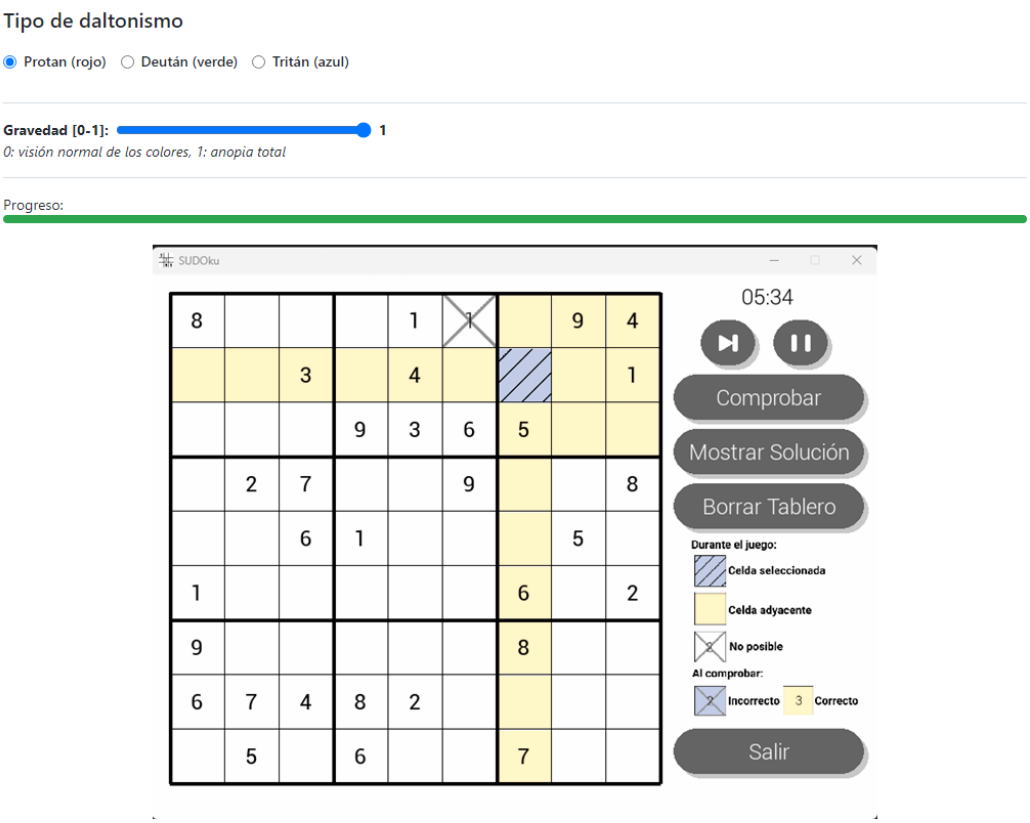


Ilustración 3.36: Simulador de daltonismo tipo Protanopia

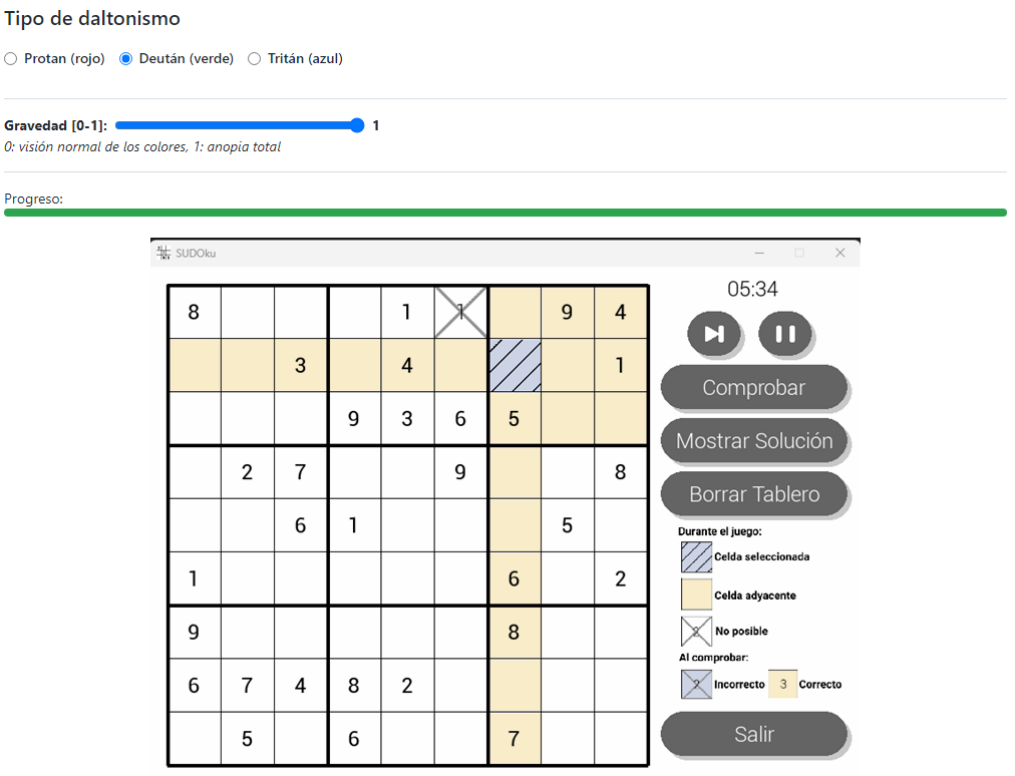


Ilustración 3.37: Simulador de daltonismo tipo Deuteranopia

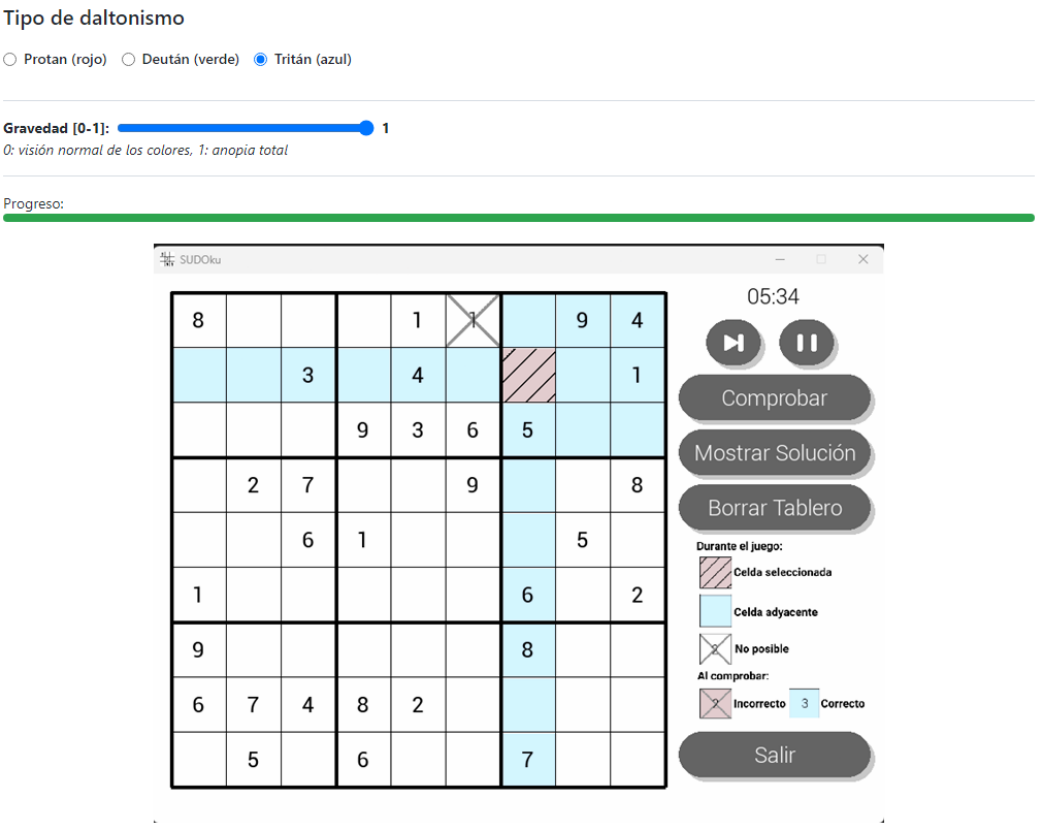


Ilustración 3.38: Simulador de daltonismo tipo Tritanopia

3.10.3.3 Explicación de uso de colores y formas

Por último, es necesario que toda la información importante no se proporcione únicamente en formato color, sino que debe ser escrita en formato texto para que todos los usuarios lo puedan leer según (Web Accessibility Initiative, 2019), por lo que se comprueba que existe una leyenda que explica el uso de formas y colores, como se puede ver en la Ilustración 3.19: Ventana de juego con leyenda para accesibilidad.

4 CONCLUSIONES Y TRABAJOS FUTUROS

Como conclusión final se puede decir que el resultado ha sido totalmente satisfactorio, ya que se ha conseguido desarrollar un juego que crea tableros de Sudoku de dificultad variable en menos de 4 segundos, todo de manera totalmente automática y añadiéndole pseudo-aleatoriedad, lo cual hace prácticamente imposible que se repitan los tableros durante una jornada de partidas seguidas. Creo que el resultado ha sido un producto software fácilmente escalable y muy útil, no contiene publicidad, no invade la privacidad de todo aquel que lo descarga y contiene infinidad de horas de diversión.

En los puntos de mejora del trabajo actual encontramos la mejora de la variedad de Sudokus a elegir, como pueden ser Sudoku Samurai (ver la Ilustración 4.1: Sudoku samurai) o Sudoku Killer (ver la Ilustración 4.2).

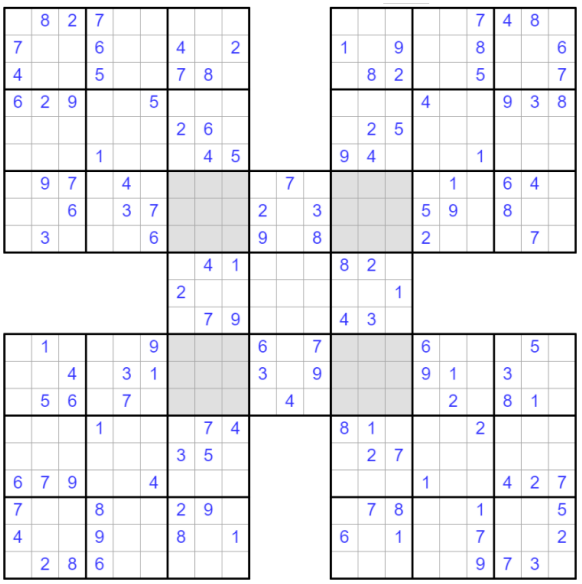


Ilustración 4.1: Sudoku samurai, via (Janko, Sudoku Samurai, 2023)

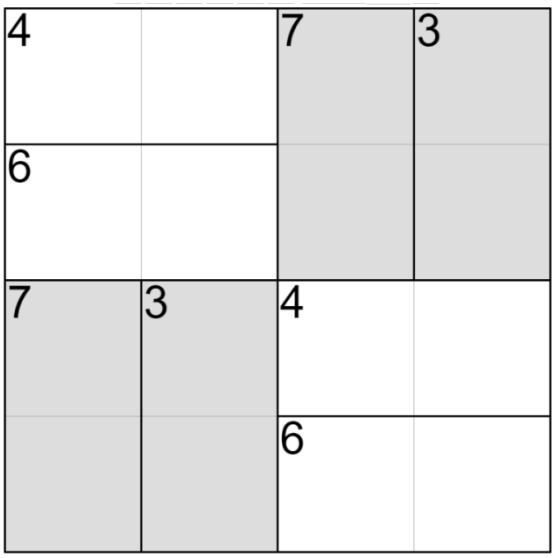


Ilustración 4.2: Sudoku killer, via (Janko, Sudoku Killer, 2023)

También sería de gran interés el hecho de añadir mayores características de accesibilidad, como pueden ser la transcripción de audio a texto, o incluir sonidos en los botones que se pulsen.

Otra mejora a tener en cuenta sería añadir un sistema de pistas que influya en la puntuación. De esta manera, el usuario podría pedir pistas cuando se atasque, a costa de perjudicar su puntuación final o añadir algún límite de pistas por nivel.

Por último, una mejora que se podría realizar es hacer que el juego sea multiplataforma y multidispositivo, lo cual ayudaría a su difusión y usabilidad de muchas personas. Para ello, únicamente faltaría hacer un ejecutable para las demás plataformas, puesto que Python ya proporciona la compatibilidad entre distintas plataformas.

5 APÉNDICES

5.1 Guía original del Trabajo Fin de Título

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

PROPUESTA DE TRABAJO DE FIN DE GRADO
GRADO EN: Grado en Ingeniería Informática **ESPECIALIDAD:** General **MENCIÓN:** General
TÍTULO DEL TRABAJO: Diseño e implementación de un juego de sudoku **Idioma:** Castellano
DESCRIPCION CORTA DEL TFG: El objetivo de este trabajo es realizar el diseño e implementación de una aplicación para generar tableros de sudoku con dificultad variable y jugar en ellos. El usuario podrá configurar las características del tablero a generar, y la aplicación implementará las reglas del juego, permitiendo que el usuario juegue, e indicándole distintas métricas (tiempo, mejores resultados, promedios, etcétera).

DATOS DEL TRABAJO FIN DE GRADO:

Modalidad del TFG

Proyecto de Ingeniería

Tipo de TFG

- ○ TFG General
- ◉ TFG Específico

TFG en equipo

- ☐ TFG en Equipo

(Debe ajuntarse memoria justificativa)

Número máximo de estudiantes

TUTOR DEL TRABAJO FIN DE GRADO:

Tutor/a

Nombre:
ANGEL LUIS GARCIA FERNANDEZ
Departamento:
Informática
A. Conocimiento:
LENGUAJES Y SISTEMAS INFORMÁTICOS

☐ **Este TFG tiene un segundo tutor**

DATOS DEL ALUMNO ASIGNADO

Alumno/a asignado/a

Nombre:
María de las Maravillas Luque Carmona
DNI:
31881177A
Dirección:
Calle Patrocinio de Biedma 2, 7ºE. 23009 Jaén
Teléfono:
675583182
Correo-E:
mmlc0007@red.ujaen.es

CONOCIMIENTOS/REQUISITOS PREVIOS RECOMENDADOS

- Conocimientos de diseño de interfaces.
- Conocimientos de Python.

OBJETIVOS DEL TFG:

- Poner en práctica las competencias y destrezas adquiridas mientras se ha estado cursando el grado.
- Demostrar el grado de madurez en la elaboración de análisis y diseño de aplicaciones.
- Crear un producto software completo y correcto, listo para su utilización por usuarios finales.

METODOLOGÍA A DESARROLLAR:

La aplicación se podrá desarrollar tanto con una metodología clásica en cascada como con una metodología ágil basada en iteraciones. La decisión sobre la metodología a utilizar se tomará al inicio del proyecto.

DOCUMENTOS Y FORMATOS DE ENTREGA:

Toda la documentación se entregará en formato digital, incluyendo.

- Código fuente de la aplicación.
- Manual de instalación.
- Manual de uso.
- Memoria del proyecto, incluyendo la documentación de análisis, diseño, implementación y pruebas en el formato correspondiente a la metodología seguida.

DOCUMENTOS ENTREGADOS:

Documentos entregados

Solicitud de propuesta de TFG cumplimenada y firmada.

5.2 Instalación y configuración del sistema

Para ejecutar el juego únicamente será necesario disponer de su ejecutable, ya que al ser un software básico no requiere de instalación ni configuración. Todo se gestiona automáticamente desde el ejecutable, creando las carpetas de usuario necesarias y los archivos de configuración que se necesitarán durante el juego.

Ilustración 5.1: Ejecutable del juego

5.3 Manuales de usuario

El manual de usuario está insertado en el propio software desarrollado, adjuntado también en las ilustraciones Ilustración 5.2 a Ilustración 5.7.

Ilustración 5.2: Primera página de manual de usuario

Ilustración 5.3: Segunda página de manual de usuario

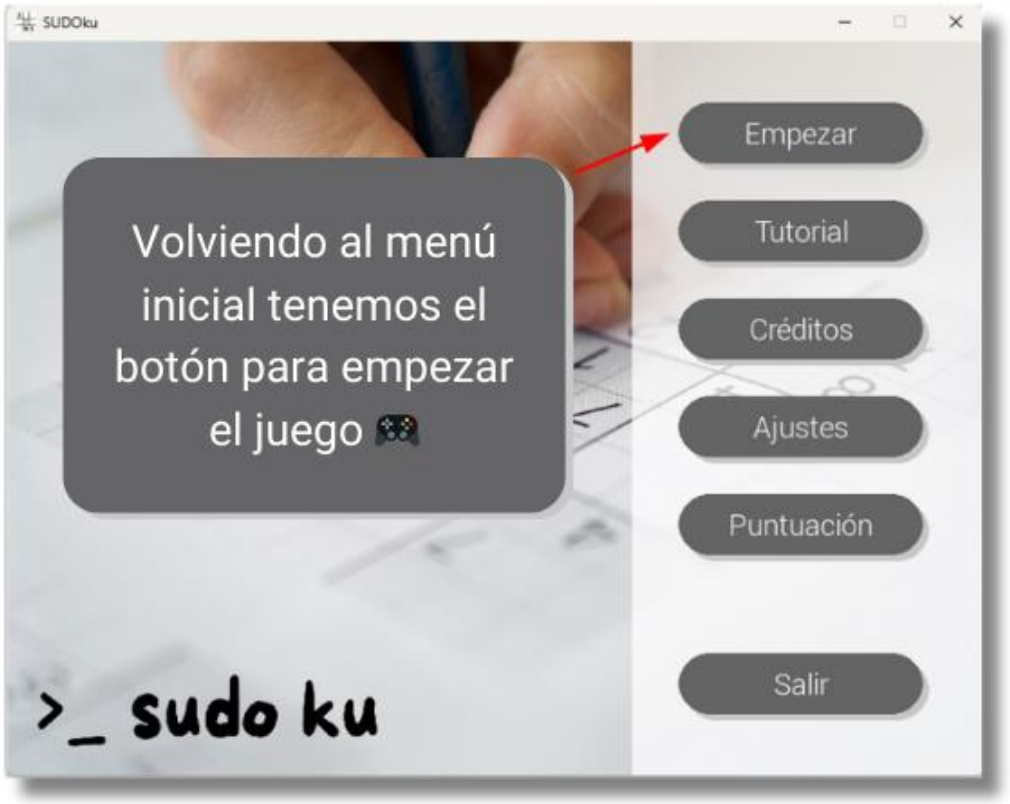


Ilustración 5.4: Tercera página de manual de usuario

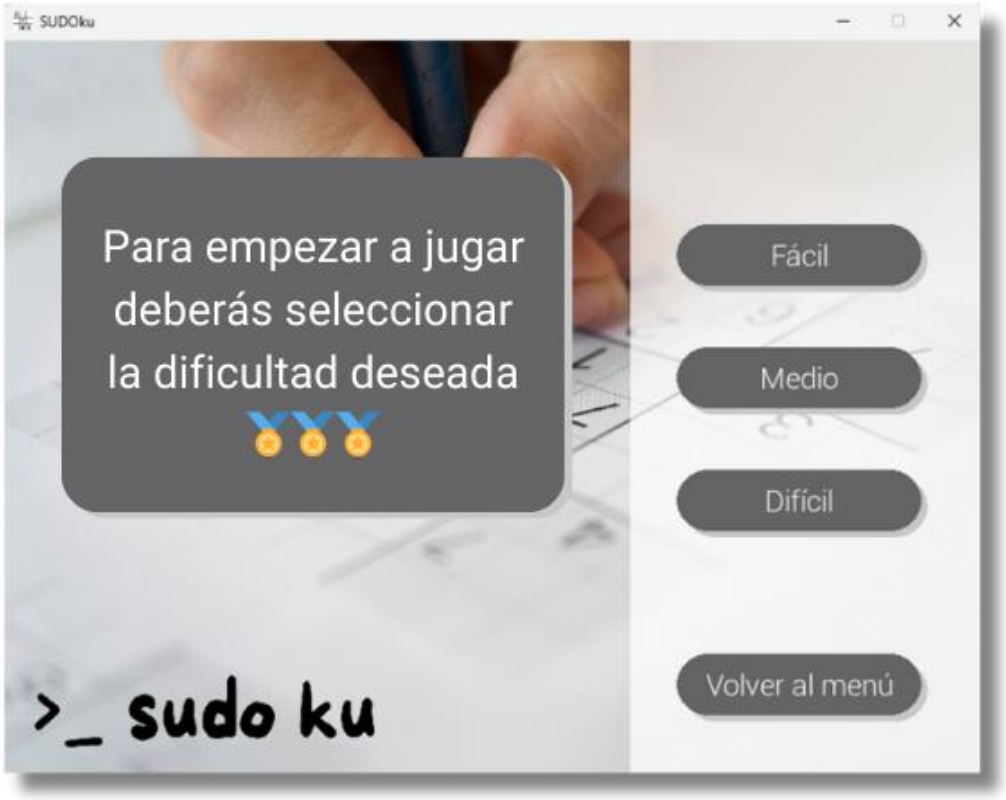


Ilustración 5.5: Cuarta página de manual de usuario

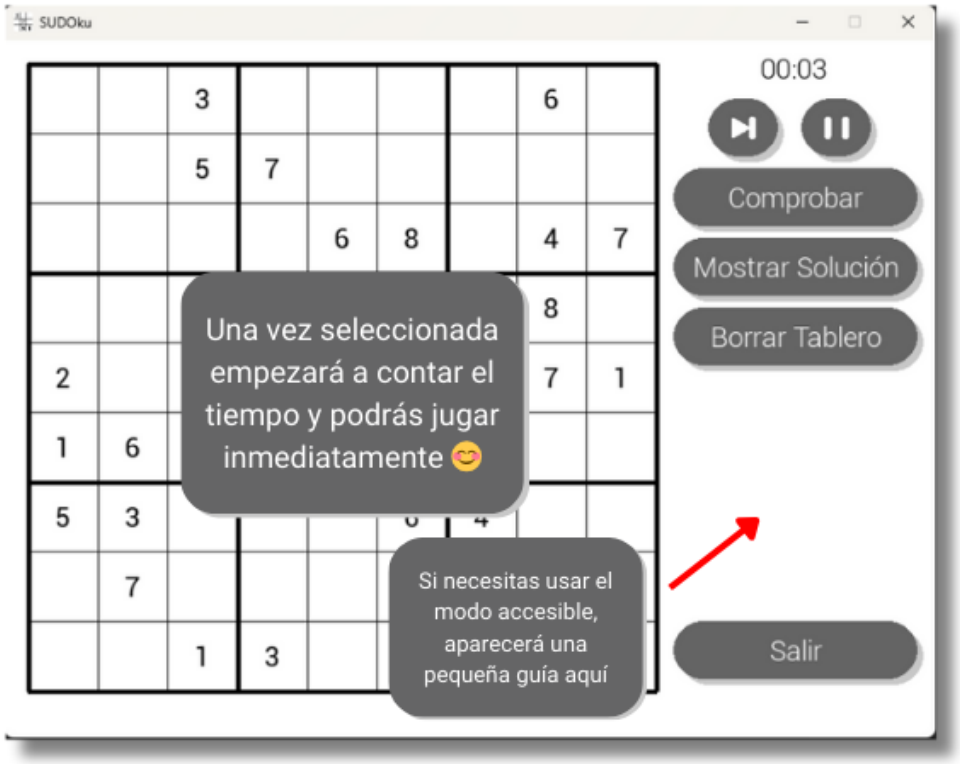


Ilustración 5.6: Quinta página de manual de usuario

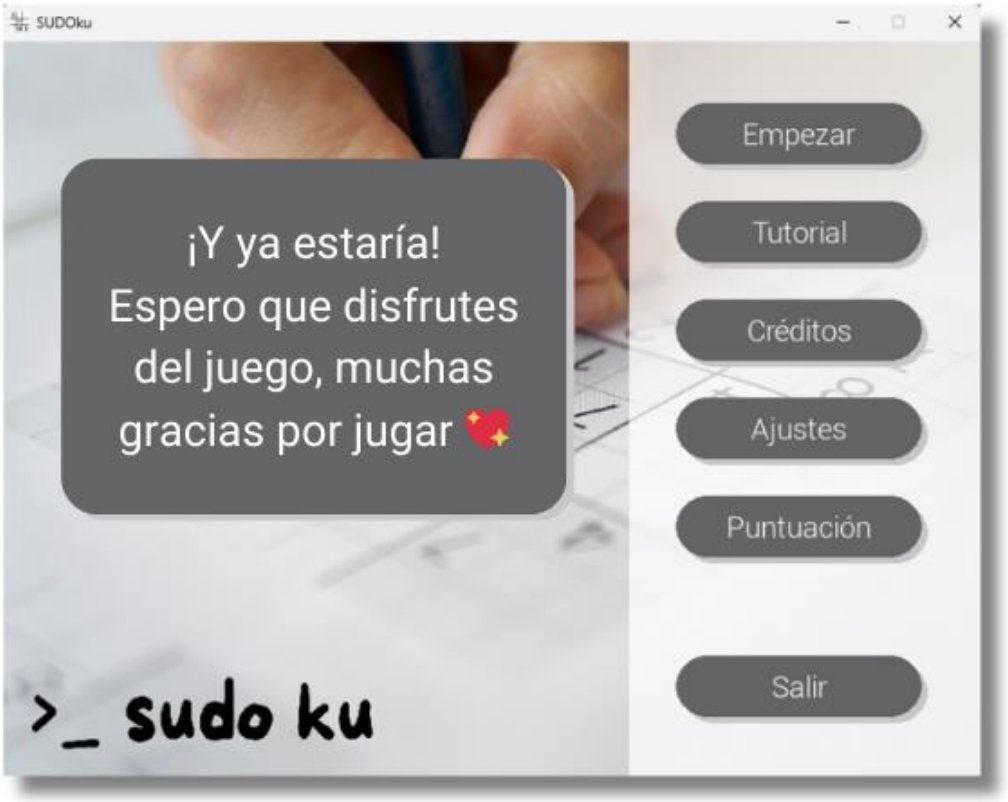


Ilustración 5.7: Sexta página de manual de usuario

6 DEFINICIONES Y ABREVIATURAS

- **Pop-up**: ventana emergente que aparece de manera repentina en la pantalla de un sitio web, aplicación o programa, tapando el contenido existente previamente hasta que éste es cerrado.
- **Juego indie:** videojuego independiente, normalmente desarrollado por un grupo reducido de individuos o una empresa de tamaño pequeño, con un presupuesto mucho menor que una gran empresa desarrolladora de videojuegos.
- **CSV**: archivo de texto en el que aparecen valores separados por comas, del inglés *"Comma Separated Values*", útil para el almacenamiento y gestión de grandes cantidades de datos.
- **COCOMO:** es un modelo matemático constructivo de costes que tiene una base empírica para la estimación de costes de software. Incluye tres submodelos, los cuales son el modelo orgánico, medio y empotrado en

función de la complejidad del proyecto y el nivel de aproximación que se requiera (Boehm, 1981).

- **Diagrama de Gantt:** herramienta de gestión de proyectos que representa gráficamente el trabajo que se ha realizado durante un tiempo concreto. (Eddie Meardon, Atlassian, 2024).
- **Backtracking**: algoritmo de búsqueda utilizado para resolución de problemas de manera recursiva, explorando todas las posibles soluciones de manera eficiente. Funciona intentando construir una solución paso a paso. Si detecta que el paso que se ha dado incumple alguna regla o restricción del sistema, vuelve hacia atrás y sigue avanzando (Bratley, 1997).

7 BIBLIOGRAFÍA

Adobe Color. (2024). *Analizador de contraste de colores*. Obtenido de https://color.adobe.com/es/create/color-contrast-analyzer

Adobe Color. (2024). *Comprobador de distinguibilidad de colores*. Obtenido de https://color.adobe.com/es/create/color-accessibility

Alonso, S. (12 de Febrero de 2024). *7 Lenguajes de programación para aprender en 2024*. Recuperado el 15 de Agosto de 2024, de https://dinahosting.com/blog/lenguajes-programacion-2024/

Boehm, B. (1981). *Software Engineering Economics.* Prentice Hall.

Bratley, G. B. (1997). *Fundamentos de Algoritmia.* Prentice Hall.

Canva. (2024). *Página principal de Canva*. Obtenido de https://www.canva.com/

Colorlitelens.com. (2024). *Simulador de daltonismo*. Obtenido de https://www.es.colorlitelens.com/images/tesztek/simulator/

Dev, L. (22 de Junio de 2023). *Top 8 Most Demanded Programming Languages in 2023*. Recuperado el 2 de Agosto de 2024, de https://www.devjobsscanner.com/blog/top-8-most-demanded-programming-languages/

Eddie Meardon, Atlassian. (2024). *¿Qué son los diagramas de Gantt?* Obtenido de https://www.atlassian.com/es/agile/project-management/gantt-chart

Fasung Design. (s.f.). *Sudoku Central, Microsoft Apps*. Recuperado el Julio de 2024, de https://apps.microsoft.com/detail/9nblggh4w4fm?hl=es-es&gl=ES

Git. (2024). *Página principal de Git*. Obtenido de https://git-scm.com/

GitHub. (2024). *Página principal de GitHub*. Obtenido de https://github.com/

Glassdoor España. (2024). *Sueldo medio para el puesto de Middle Developer*. Obtenido de https://www.glassdoor.es/Sueldos/middle-developer-sueldo-SRCH_KO0,16.htm

Glassdoor España. (2024). *Sueldos para el puesto de Disenador Grafico en España*. Obtenido de https://www.glassdoor.es/Sueldos/disenador-grafico-sueldo-SRCH_KO0,17.htm

Glassdoor España. (2024). *Sueldos para el puesto de Ingeniero De Pruebas en España*. Obtenido de https://www.glassdoor.es/Sueldos/ingeniero-de-pruebas-sueldo-SRCH_KO0,20.htm

Glassdoor España. (2024). *Sueldos para el puesto de Jefe De Proyectos en España*. Obtenido de https://www.glassdoor.es/Sueldos/jefe-de-proyectos-sueldo-SRCH_KO0,17.htm

Glassdoor España. (2024). *Sueldos para el puesto de Programador Senior en España*. Obtenido de https://www.glassdoor.es/Sueldos/programador-senior-sueldo-SRCH_KO0,18.htm

Janko, O. (18 de Mayo de 2023). *Sudoku Killer*. Obtenido de https://www.psicoactiva.com/juegos-inteligencia/area-sudoku/sudoku-killer/

Janko, O. (18 de Mayo de 2023). *Sudoku Samurai*. Obtenido de https://www.psicoactiva.com/juegos-inteligencia/sudoku-samurai/

Jauzat, L., & Jauzat, L. (19 de Diciembre de 2023). *Billiken*. Recuperado el 4 de Julio de 2024, de https://billiken.lat/interesante/commodore-64-una-computadora-para-las-masas/

McGuire, G. T. (1 de Enero de 2012). *There is no 16-Clue Sudoku: Solving the Sudoku Minimum Number of Clues Problem*. Recuperado el 8 de Mayo de 2024, de https://arxiv.org/abs/1201.0749v2

Microsoft Apps. (2024). *Microsoft Apps*. Recuperado el Julio de 2024, de https://apps.microsoft.com/home?hl=es-es&gl=ES

Microsoft. (Julio de 2024). *Visual Studio Code: Página principal*. Recuperado el 4 de Julio de 2024, de https://code.visualstudio.com/

NumPy team. (2024). Obtenido de https://numpy.org/

Pygame. (2024). *Pygame*. Obtenido de https://www.pygame.org/news

PyInstaller. (2024). *Documentación de PyInstaller*. Obtenido de de una manera realmente cómoda

Python. (Septiembre de 2024). *Documentación de la biblioteca CSV*. Obtenido de https://docs.python.org/3/library/csv.html

Python Software Foundation. (2024). *Página principal de Python*. Obtenido de https://www.python.org/downloads/

Ramirez, O. (23 de Mayo de 2024). *10 Lenguajes de programación más usados en 2024*. Recuperado el 15 de Julio de 2024, de https://bambu-mobile.com/10-lenguajes-de-programacion-mas-usados/#:~:text=%C2%BFCu%C3%A1l%20es%20el%20mejor%20lenguaje,desarrollo%20web%20hasta%20inteligencia%20artificial.

ResearchGate. (2016). *Modelo de proceso incremental*. Recuperado el 2024, de https://www.researchgate.net/figure/Figura-10-Modelo-de-proceso-incremental-Fuente_fig6_326571456

Sarah Laoyan, A. (6 de Febrero de 2024). *Qué es la metodología waterfall y cuándo utilizarla*. Recuperado el 10 de Agosto de 2024, de https://asana.com/es/resources/waterfall-project-management-methodology

sistemasoperativos.info. (Abril de 2023). *Los sistemas operativos más utilizados*. Recuperado el Agosto de 2024, de https://sistemasoperativos.info/blog/los-sistemas-operativos-mas-utilizados/

Splutterworth, D. C. (s.f.). *Juego MySudoku*. Obtenido de https://store.steampowered.com/app/3033660/My_Sudoku/

Valve Corporation. (2024). *Steam: Página principal*. Recuperado el Julio de 2024, de https://store.steampowered.com/?l=spanish

Web Accessibility Initiative. (2019). *How to Meet WCAG (Quickref Reference), Use of color*. Recuperado el 10 de Julio de 2024, de https://www.w3.org/WAI/WCAG22/quickref/?versions=2.1#use-of-color

Web Accessibility Initiative. (2019). *How to Meet WCAG, Quickref Reference, Contrast*. Recuperado el 10 de Julio de 2024, de https://www.w3.org/WAI/WCAG22/quickref/#contrast-minimum

Web Accessibility Initiative. (19 de Mayo de 2019). *Principios de accesibilidad*. Recuperado el Julio de 2024, de https://www.w3.org/WAI/fundamentals/accessibility-principles/es

Ydoate, J. (6 de Septiembre de 2021). *¿Cuáles son las plataformas para comprar videojuegos oficiales más importantes?* Recuperado el 4 de Julio de 2024, de Marketing Insider Review: https://marketinginsiderreview.com/plataformas-comprar-videojuegos/