

Diseño e implementación de “Defend Your Privacy”, un minijuego para ayudar a los adolescentes a entender lo que no deben publicar en los medios sociales

Vicente Navarro, Angel
Curs 2020-2021

Directors: Davinia Hernández-Leo
Emily Theophilou
René Lobo

GRAU EN ENGINYERIA INFORMÀTICA



Universitat
Pompeu Fabra
Barcelona

Escola
Superior Politècnica

Treball de Fi de Grau

Agradecimientos

Me gustaría dar las gracias a Davinia Hernández, Emily Theophilou, René Lobo y Roberto Sánchez no sólo por darme la oportunidad de colaborar en el proyecto Courage haciendo este minijuego, sino también por su disponibilidad y consejos que me han dado a lo largo del transcurso de este trabajo de fin de grado.

También me gustaría agradecer a mi familia por su apoyo durante estos últimos años.

Resumen

Durante los últimos años, los medios sociales han logrado hacerse un hueco en la sociedad hasta tal punto que es inimaginable pensar en la existencia de un mundo sin estas plataformas de comunicación en línea. Si bien es cierto que su objetivo radica en conectar a la gente mediante el intercambio digital de contenido, no podemos pasar por alto los riesgos a los que nos exponemos cuando hacemos uso de ellos. Uno de los riesgos principales es la exposición de nuestra privacidad, y dado que los adolescentes suelen ser los más afectados debido a su falta de conocimientos en este ámbito, me he propuesto crear un minijuego orientado a la web para enseñarles a combatir esta problemática de una forma lúdica.

En este trabajo se presentan las distintas fases seguidas para la creación de este minijuego gráfico, partiendo de su diseño hasta su implementación con el framework de VUE JS. Finalmente, para validar si se han logrado los objetivos propuestos, se hará una evaluación con un grupo de voluntarios.

Resum

Durant els últims anys, els mitjans socials han aconseguit fer-se un forat en la societat fins al punt que és inimaginable pensar en l'existència d'un món sense aquestes plataformes de comunicació en línia. Si bé és cert que el seu objectiu rau en connectar a la gent mitjançant l'intercanvi digital de contingut, no podem passar per alt els riscos a què ens exposem quan fem ús d'ells. Un dels riscos principals és l'exposició de la nostra privacitat, i atès que els adolescents solen ser els més afectats per la seva falta de coneixements en aquest àmbit, m'he proposat crear un mini-joc orientat a la web per ensenyar-los a combatre aquesta problemàtica d'una forma lúdica.

En aquest treball es presenten les diferents fases seguides per a la creació d'aquest mini-joc gràfic, partint del seu disseny fins a la seva implementació amb el framework de VUE JS. Finalment, per validar si s'han assolit els objectius proposats, es farà una valuació amb un grup de voluntaris.

Abstract

Over the last few years, social media have carved out a niche for themselves in society to such an extent that it is unimaginable to think of a world without these online communication platforms. While it is true that their purpose is to connect people through the digital exchange of content, we cannot overlook the risks we expose ourselves to when we use them. One of the main risks is the exposure of our privacy, and since teenagers are often the most affected due to their lack of knowledge in this area, I have proposed to create a web-oriented mini-game to teach them how to combat this problem in a playful way.

This paper presents the different phases followed for the creation of this graphic mini-game, from its design to its implementation with the VUE JS framework. Finally, to validate whether the proposed objectives have been achieved, an evaluation will be carried out with a group of volunteers.

ÍNDICE

Agradecimientos.....	ii
Resumen	iv
Resum	iv
Abstract	v
Lista de figuras	x
Lista de tablas	xii
Glosario	xiii
1. INTRODUCCIÓN	1
1.1 Problema principal	1
1.2 Solución propuesta	1
1.3 Cronograma	2
2. ESTADO DEL ARTE	5
2.1 Medios sociales.....	5
a) Redes sociales	6
b) Blogs.....	6
c) Multimedia	7
2.2 ¿Como está afectando a la sociedad?.....	8
2.3 Huella digital.....	9
2.4 Privacidad	10
2.5 Juegos educativos existentes	11
a) A Clockwork Brain	11
b) Bad News	12
c) Cyber Defense Quiz	13
d) Cyber Five	14
3. REQUERIMIENTOS	17
3.1 Requerimientos funcionales	17
3.2 Requerimientos no funcionales	17
4. ANTECEDENTES TÉCNICOS.....	21
4.1 Software para la programación web.....	21
a) HTML.....	21
b) CSS	21
c) JavaScript	22
4.2 Librerías JS	23
a) jQuery	23

b) Node.js	24
4.3 Frameworks JS	24
a) Angular	24
b) React	25
c) VueJS	25
4.4 Creación de assets para el juego.....	26
a) Piskel	26
b) Tiled.....	27
c) Figma.....	28
5. SISTEMA CENTRAL DEL JUEGO.....	31
5.1 Personaje / Jugador	31
a) ¿Cómo es el personaje que protagoniza el juego?	31
b) Reglas de acción	36
c) Reglas de estado	36
5.2 Reglas de control	36
5.3 Mundo del juego.....	37
a) Componentes del mundo del juego	37
b) Diseño de niveles	41
c) Punto de vista y cámara	42
6. DISEÑO.....	47
6.1 Wireframes.....	47
6.2 Diagrama de flujo.....	53
7. IMPLEMENTACIÓN.....	55
7.1 Sintaxis de VUE	55
a) Directivas.....	55
b) Componentes	55
c) Vuex.....	56
7.2 Estados de juego	56
7.3 Intro Stage	59
7.4 Language Stage.....	60
7.5 Play Stage.....	62
7.6 Juego principal	66
a) Mapas.....	66
b) Jugador.....	69
c) Obtáculos.....	73

d) Recursos	74
e) Bucle de juego.....	74
7.6 Final Stage.....	81
8. Fase de validación	85
8.1 Resultados encuesta.....	86
9. CONCLUSIONES Y TRABAJO FUTURO	89
9.1 Conclusiones.....	89
9.2 Trabajo futuro.....	90
REFERENCIAS.....	91
ANEXO	95
Encuesta	95

Lista de figuras

Figura 1: Cronograma del trabajo realizado	3
Figura 2: Tiempo promedio utilizado en los medios sociales.....	5
Figura 3: Medios sociales más utilizados	8
Figura 4: Wheel of Colors	12
Figura 5: Chase The Numbers	12
Figura 6: Bad News pantalla de juego	13
Figura 7: Cyber Defense Quiz pantalla de juego.....	14
Figura 8: Cyber Five, animación a la izquierda y pregunta quiz a la derecha.....	14
Figura 9: PixelFed logo	18
Figura 10: PixelFed interfaz	19
Figura 11: Ejemplo documento HTML.....	21
Figura 12: Ejemplo documento HTML estilizado con CSS	22
Figura 13: Página web utilizando CSS (izquierda) vs misma página web sin utilizar CSS (derecha)	22
Figura 14: jQuery logo	23
Figura 15: node.js logo	24
Figura 16: Angular logo.....	24
Figura 17: React logo.....	25
Figura 18: VueJS logo	25
Figura 19: Piskel	27
Figura 20: Tiled.....	28
Figura 21: Figma Interfaz.....	29
Figura 22: Chico básico Spritesheet	32
Figura 23: Chica básica spritesheet.....	33
Figura 24: El Sabio spritesheet	34
Figura 25: Punky spritesheet	34
Figura 26: Chica de hielo	35
Figura 27: Narrador	37
Figura 28: Obstáculo ID	38
Figura 29: Obstáculo Contraseña	38
Figura 30: Obstáculo Tarjeta de crédito	39
Figura 31: Obstáculo datos bancarios.....	39
Figura 32: Obstáculo Vacaciones	40
Figura 33: Obstáculo Dirección de casa	40
Figura 34: Obstáculo Número de teléfono.....	40
Figura 35: Recurso Vida	41
Figura 36: Recurso Quiz	41
Figura 37: Representación nivel 1.....	42
Figura 38: Representación nivel 2.....	42
Figura 39: Representación nivel 3.....	42
Figura 40: Super Mario Bros (NES) 1985, Nintendo	43
Figura 41: Terraria 2011, Re-logic	43
Figura 42: The Legend of Zelda (NES) 1986, Nintendo	44
Figura 43: Sonic the Hedgehog 1991, Sega	44
Figura 44: Scramble 1981, Konami.....	45
Figura 45: Pantalla inicial.....	47
Figura 46: Pantalla selección de idioma.....	48
Figura 47: Pantalla selección de niveles	48
Figura 48: Pantalla selección de personajes.....	49
Figura 49: Pantalla información detallada personaje.....	49
Figura 50: Pantalla juego principal	50
Figura 51: Pantalla pregunta quiz	50

Figura 52: Pantalla menú pausa	51
Figura 53: Pantalla menú obstáculos	51
Figura 54: Pantalla de resultados.....	52
Figura 55: Pantalla de ranking	52
Figura 56: Pop Up narrador	53
Figura 57: Diagrama de flujo transición entre pantallas.....	54
Figura 58: VUE Ejemplo componentes	55
Figura 59: Diagrama estructura del código	57
Figura 60: Definición estados de la store	58
Figura 61: Definición mutaciones de la store	58
Figura 62: HTML Game	59
Figura 63: Resultado final Pantalla inicial.....	60
Figura 64: Definición Lenguajes.....	60
Figura 65: Listar idiomas en HTML	61
Figura 66: Resultado final pantalla selección idioma.....	61
Figura 67: Definición Play Stage en HTML	62
Figura 68: Definición Nivel	62
Figura 69: Mostrar niveles HTML versión simplificada	63
Figura 70: Resultado final pantalla selección de niveles	64
Figura 71: Selección de personajes en HTML	65
Figura 72: Detalle personaje por delante (izquierda) y por detrás (derecha)	65
Figura 73: Definición mapa	66
Figura 74: Representación del tilemap.....	67
Figura 75: Función drawMap	68
Figura 76: Función drawSection	68
Figura 77: Definición personaje	69
Figura 78: Clase Player	70
Figura 79: Función mouseDown de Player	71
Figura 80: Función moveOnMouseDown de Player	71
Figura 81: Movimiento personaje	72
Figura 82: Representación partículas flotantes	72
Figura 83: Clase Obstacle	73
Figura 84: Función update de los Obstáculos	74
Figura 85: Función para generar nuevos obstáculos	76
Figura 86: Visualización del juego.....	77
Figura 87: Definición caja de colisión	78
Figura 88: Función para detectar una colisión	78
Figura 89: Colisión con obstáculo	79
Figura 90: Definición pregunta quiz	79
Figura 91: Pregunta quiz estándar (izquierda) y pregunta con respuesta descartada (derecha)	80
Figura 92: Menú de pausa (izquierda) y lista de obstáculos (derecha).....	81
Figura 93: JSON puntuación.....	81
Figura 94: Función para actualizar el ranking	82
Figura 95: FinalStage simplificado HTML.....	83
Figura 96: Pantalla de resultados (izquierda) y ranking (derecha).....	84
Figura 97: Cuánto tiempo les dedican a los medios sociales de media al día	86
Figura 98: Identificación de los obstáculos sin la necesidad de consultar su definición	87
Figura 99: Nivel de dificultad de las preguntas quiz	88
Figura 100: Aprendizaje en el juego.....	88

Lista de tablas

Tabla 1: Comparativa juegos educativos	15
Tabla 2: Características de los personajes	35

Glosario

AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	HyperText Markup Language
JS	JavaScript
JSON	JavaScript Object Notation
W3C	World Wide Web Consortium

1. INTRODUCCIÓN

1.1 Problema principal

Cuando navegamos por los medios sociales estamos acostumbrados a encontrarnos con contenido tóxico, falso o simplemente que pretende herir los sentimientos de los demás. Pero hay otro factor clave que a menudo es pasado por alto y supone un grave riesgo para todos los usuarios: nuestra privacidad. Es por esta razón que nosotros, como usuarios, hemos de ir con mucho cuidado con todo lo que publicamos en la red ya que de caer en las manos equivocadas nos podría hacer pasar un muy mal rato.

1.2 Solución propuesta

Con el fin de concienciar a los jóvenes sobre lo que uno debe y no debe publicar en los medios sociales bajo ningún concepto, me propuse crear un minijuego educativo mediante el cual puedan aprender sobre este tema a la vez que se lo pasan bien. La idea de este minijuego gráfico es la siguiente:

En el juego, nosotros controlamos a un personaje que siempre está corriendo hacia delante, pero a medida que avanza se va encontrando con distintos obstáculos que tendrá que evitar desplazándose a la derecha o a la izquierda (en realidad el jugador siempre está en la misma posición y son los obstáculos los que se desplazan, así se crea la ilusión de que es el personaje el que se está moviendo). Estos obstáculos serán una representación gráfica de aquellas cosas que el usuario no debe publicar en la red. Cuanto más avance el jugador sin chocarse, más puntos ganará.

De vez en cuando aparecerán unos objetos que si cogemos nos darán la opción de ganar puntos adicionales si resolvemos una pregunta tipo quiz sobre conceptos sobre qué se debe publicar y que no. Por ejemplo:

- “De las siguientes opciones cuál no debería ser publicada?” Y se mostrarán distintas opciones dónde sólo una sería válida.

- O al contrario “¿Cuál si publicarías?” de esta manera el usuario tendría que estar más atento al juego ya que las preguntas no siempre serían del mismo estilo.

Este tipo de preguntas tendrían un límite de tiempo para ser respondidas, por consiguiente, se ejercitará la rapidez mental puesto que cuando navegamos por la red hay que saber reaccionar a tiempo.

El juego estará formado por distintos niveles, donde en cada uno podremos controlar a una serie de personajes con distintas habilidades, de tal manera que el juego no se hará repetitivo. Mas adelante veremos con más detalle las distintas fases que supone crear un juego de este estilo, partiendo de su diseño (donde se detallaran todas las funcionalidades y mecánicas) hasta su implementación.

1.3 Cronograma

Este trabajo se ha dividido principalmente en 4 fases:

- **Fase de documentación:** durante el primer mes me centré en documentarme sobre los distintos problemas que nos podemos encontrar cuando utilizamos los medios sociales. Una vez encontré una temática que suponía un riesgo para los adolescentes (la temática de la privacidad) empecé a diseñar los primeros esbozos del minijuego.
- **Fase de formación:** una vez tuve claro de que trataría el minijuego, empecé a documentarme sobre las diferentes tecnologías que utilizaría para implementarlo. Cómo se especifica en el apartado 4 (antecedentes técnicos), el juego ha sido implementado utilizando el framework de VueJS y dado que nunca lo había utilizado, tuve que aprender a utilizarlo para poder sacarle el máximo rendimiento posible.
- **Fase de implementación:** en este periodo me centré en definir todos los elementos que componen el juego, diseñar cada una de las pantallas y elementos que aparecen en él y también la parte más importante,

implementar el juego. Durante todo el proyecto he ido documentando cada uno de los pasos seguidos.

- **Fase de validación:** esta última fase del proyecto consiste en realizar una validación del juego (una vez se haya finalizado su implementación) con un grupo de voluntarios para ver si se han cumplido los objetivos que me propuse al inicio de este trabajo.

En la figura 1 se puede observar el cronograma seguido.

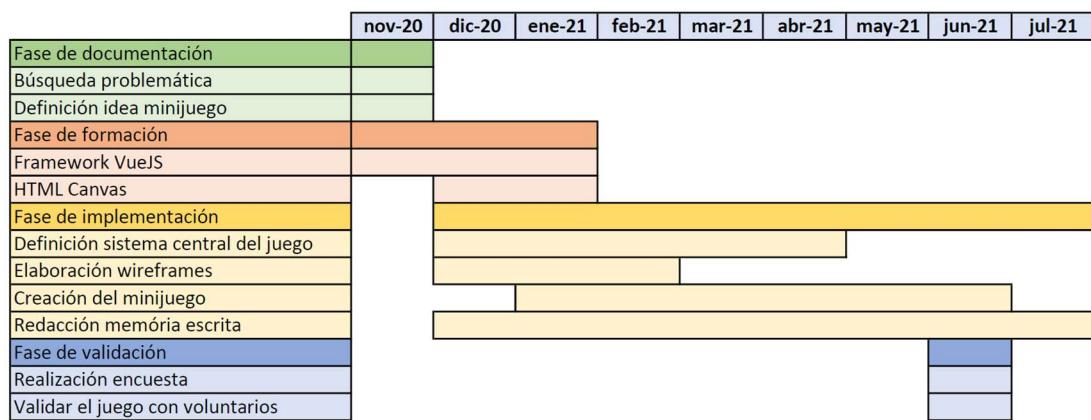


Figura 1: Cronograma del trabajo realizado

2. ESTADO DEL ARTE

Durante los últimos años, los medios sociales han logrado hacerse un hueco en nuestra sociedad hasta tal punto que es inimaginable pensar en la existencia de un mundo sin estas plataformas de comunicación en línea.

Según el estudio “Digital in 2021” publicado por “We Are Social” en colaboración con “Hootsuite” [1], en enero de 2021 cerca del 60% de la población mundial utilizaron Internet (4'6 billones de personas), de los cuales el 91'3% utilizaban los medios sociales (4'2 billones de personas), lo que equivale a un 53% de la población mundial. En este estudio se estimó que un usuario dedica una media de 2h y 35minutos al día a estar conectado a los medios sociales. En la figura 2 podemos observar el tiempo promedio utilizado cada día en los medios sociales.

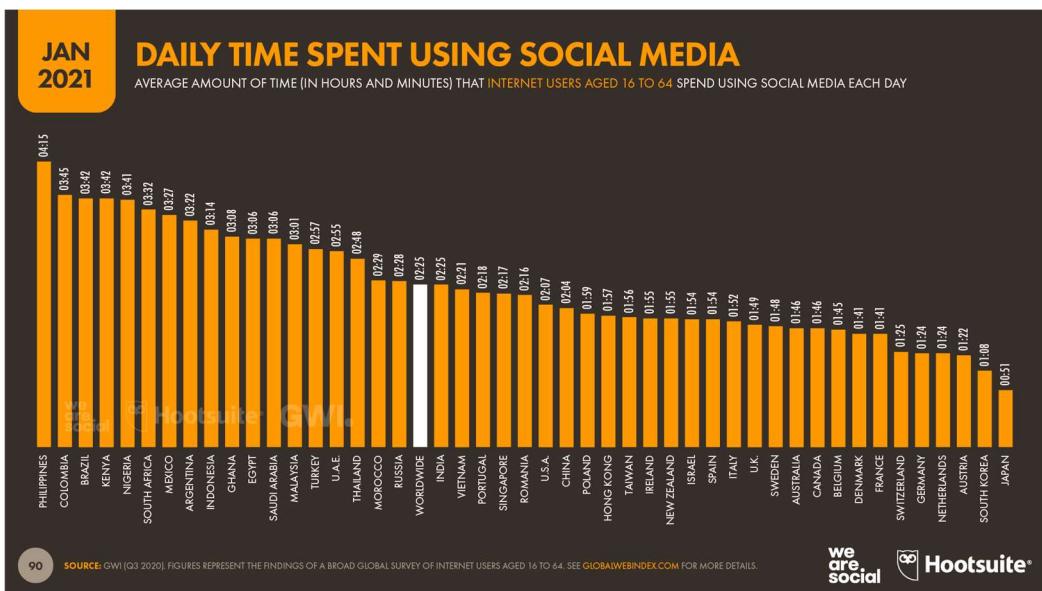


Figura 2: Tiempo promedio utilizado en los medios sociales

2.1 Medios sociales

Hemos visto que cada día millones de personas alrededor del mundo utilizan los medios sociales, pero ¿qué son exactamente y porque dotan de tanta popularidad?

Andreas M. Kaplan y Michael Haenlein las definen como “un grupo de aplicaciones basadas en Internet que se desarrollan sobre los fundamentos ideológicos y tecnológicos de la Web 2.0, y que permiten la creación y el intercambio de contenidos generados por el usuario” [2]. Es decir, son

aplicaciones o plataformas de comunicación virtual donde sus usuarios pueden interactuar en tiempo real entre ellos ya sea mediante el intercambio de mensajes, realizando publicaciones (de texto o imágenes) o editando su contenido conjuntamente.

Hoy en día encontramos una gran variedad de medios sociales, entre los que destacan:

a) Redes sociales

Las redes sociales son una plataforma web online que permiten a sus usuarios crear relaciones sociales con otros usuarios que comparten mismas aficiones o intereses.

“Facebook” y “Twitter” serían dos ejemplos de redes sociales.

b) Blogs

Un blog es una página web donde el autor expone temas de su interés generalmente escritos de manera informal y siguiendo un estilo de diario personal [3]. A cada uno de estos temas se les conoce como: “Post” y es habitual que estos aparezcan ordenados de manera cronológica descendente (el último post que se ha publicado sería el que aparece primero y el primer post publicado sería el último en aparecer) para que los lectores vean los posts más recientes nada más entrar en la página. Una de las principales características de los blogs es que permiten a sus lectores realizar comentarios sobre los posts y a su vez, estos comentarios pueden recibir otros comentarios, consiguiendo así crear una comunicación entre los lectores y el autor.

Las plataformas más conocidas mundialmente para la creación de blogs son: “Blogger” y “WordPress” (originalmente fue creada para publicar blogs, pero hoy en día soporta una amplia variedad de contenido web [4]).

c) Multimedia

Este tipo de plataforma permite a sus usuarios compartir imágenes, vídeos y retransmisiones en directo. Actualmente aparte de usuarios, también podemos encontrar a empresas que promocionan sus productos en este tipo de medio puesto que es de los medios sociales de mayor popularidad del mercado y por consecuencia, sería más sencillo llegar a potenciales clientes en comparación con otros medios [5].

Es habitual que mucha gente confunda las redes sociales con los medios multimedia ya que comparten muchas similitudes, pero, mientras la base de las redes sociales es compartir texto u opiniones con otros usuarios (incluyen la opción de compartir imágenes para reforzar ese texto), los medios multimedia están pensados para compartir imágenes o videos y, opcionalmente, se les puede añadir una pequeña descripción. Eso sí, las dos permiten la comunicación entre sus usuarios ya sea mediante comentarios en las publicaciones o por mensajería dentro de la plataforma.

Algunos ejemplos de este tipo de medio social serían: Instagram, YouTube o Snapchat.

Ahora que ya hemos visto aquellos tipos de medios más predominantes, ha llegado el momento de ver que plataformas han sido las más utilizadas a escala mundial [1], para ello, en la figura 3 se muestran los medios sociales más utilizados hasta enero de 2021:

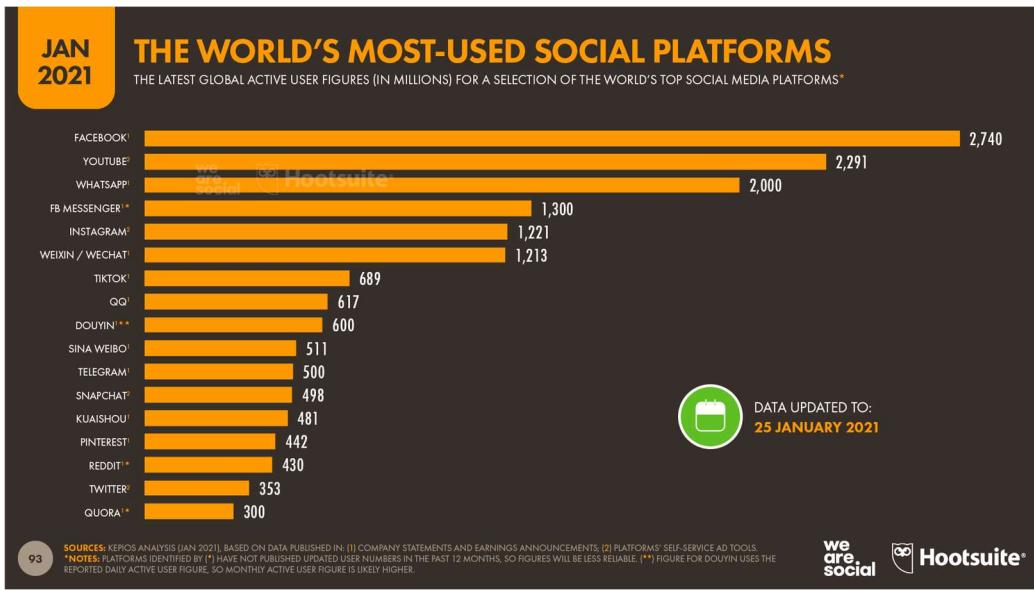


Figura 3: Medios sociales más utilizados

Como podemos ver, la red social Facebook lidera la lista de plataforma social más utilizada, seguida por la plataforma multimedia Youtube y la red de mensajería WhatsApp.

2.2 ¿Como está afectando a la sociedad?

Los medios sociales nacieron con un propósito: mantener conectada a la gente, ya sea con amigos, familiares o desconocidos mediante el intercambio digital de mensajes (compartiendo sus ideas a través de texto o imágenes). Esto ayuda a sus usuarios a mejorar sus capacidades de socialización y comunicación [6].

Por otro lado, utilizar los medios sociales también puede suponer un riesgo, sobre todo para los adolescentes debido a su falta de conocimiento y experiencia en este campo. Los peligros a los que están más expuestos son:

- Comunicación de persona a persona: cuando hablamos o chateamos con otros usuarios no podemos estar seguros al 100% de que esa persona sea quien dice ser. A través de la ingeniería social (conjunto de técnicas para engañar a otros usuarios), los cibercriminales pueden elaborarse un perfil falso con la finalidad de conseguir que otras personas hagan lo que ellos quieren ya que, por lo general, el ser humano suele confiar en

los demás. Todo ello tiene como objetivo la extracción de datos personales, chantajes o delitos de grooming [7].

- Contenido inapropiado: controlar todo lo que aparece en internet no es sencillo, por esta razón, es muy habitual que los niños se encuentren con contenido inapropiado para sus edades que les puede causar malestar o dejarlos confundidos. Con contenido inapropiado nos referimos a material [8]:
 - Con contenido sexual.
 - Fomentando el vandalismo, la delincuencia, el terrorismo, el sexism o el racismo.
 - Incitando a los juegos de apuestas.
 - Promocionando salas de chat sin restricciones (no exigen un mínimo de edad para ingresar y sin supervisión sobre lo que se comparte).
- Influencias externas: cada vez son más las empresas que utilizan estos medios para promocionar sus productos. Por un lado, incitan a todo aquel que ve sus anuncios a que los comprén, pero a la vez, pueden llegar a cambiar la manera en que perciben la realidad ya que estos suelen expresar un mundo ideal [6].
- Problemas de privacidad: dar mucha información personal o publicar información sobre otros sin su consentimiento. Si bien es cierto que hay adolescentes a los que le preocupa su privacidad, a veces lo que desean mantener en privado suele diferir a lo que un adulto cree que se debería mantener para uno mismo [9].

2.3 Huella digital

Cuando utilizamos Internet, todo lo que hacemos, por más simple que sea, deja rastro. Ya sea visitar una página web, crear una página o nuevos contenidos, comunicarnos con otra gente, reaccionar ante un post o una publicación (por ejemplo, dar un me gusta o realizar un comentario), todo ello quedará marcado

en la red por mucho tiempo y puede llegar a causar problemas de reputación, credibilidad o confianza a esos individuos que realizaron una acción indebida o mal vista por la sociedad [9].

Existen dos formas de recopilar las huellas digitales:

- Recopilación pasiva: el usuario no está al corriente de que se le están extrayendo estos datos. Estas huellas suelen rastrear la dirección IP del dispositivo en el que se encuentra el usuario, así como el momento en que se crearon y de donde provienen.
- Recopilación activa: el usuario acepta que la página en la que se encuentra le pueda recopilar sus datos. Por ejemplo, al realizar un comentario en una red social quedara registrado que usuario lo realizó.

En ambos casos, las huellas digitales se pueden almacenar en bases de datos en línea o bien en archivos locales dentro del propio dispositivo [10].

Todo ello permite a las empresas controlar las acciones que realizan sus usuarios. En el ámbito de las redes sociales, estos datos se podrían utilizar para reconstruir la personalidad o perfil de un usuario gracias al análisis de sus intereses, aficiones, amistades, localización, tipo de publicaciones realizadas o las interacciones con otras cuentas. Por otro lado, hay empresas que antes de contratar a alguien, deciden investigar sus acciones en Internet con el fin de conocer cómo es realmente en su ámbito más personal, de ahí la importancia de disponer una buena huella digital. Este proceso es conocido como “investigación cibernética”.

2.4 Privacidad

Toda información que compartimos en los medios sociales, ya sean, comentarios en publicaciones, imágenes o datos personales, pueden ser vistos y compartidos por más gente de la que uno se imagina. Por esta razón, nunca deberíamos compartir nuestro número de teléfono, direcciones, información más personal o cualquier dato que comprometa nuestra privacidad puesto que no sabemos

quién lo estará viendo, y lo más preocupante, que harán con esta información [11].

También es muy importante revisar los ajustes de privacidad de estos medios antes de aceptarlos para saber que hacen estas empresas con nuestra información ya que podrían utilizarlos para fines propios o incluso venderlos a terceros.

2.5 Juegos educativos existentes

Cuando un juego además de ofrecer entretenimiento está diseñado con la intención de educar a sus jugadores se le conoce como: Juego educativo.

A continuación, se muestran una serie de juegos educativos que encontramos actualmente en el mercado:

a) A Clockwork Brain

Este juego creado por Total Eclipse (disponible para iOS, Android y PC), nos presenta una colección de puzzles para entrenar nuestro cerebro. En él, se nos presentan distintos retos para fortalecer nuestra memoria, atención, destreza, lenguaje y razonamiento [12].

Su seña de identidad está marcada tanto por la originalidad de sus rompecabezas como por su arte (basado en la época Victoriana y en la comunidad Maya). A su vez, para lograr crear una conexión entre el jugador y el juego, utilizan un personaje con forma de robot que actúa como narrador.

Veamos alguno de sus desafíos:



Figura 4: Wheel of Colors

Para mejorar la atención, nos encontramos con un círculo que contiene un diseño en particular, tal y como muestra la figura 4. El objetivo de este rompecabezas será averiguar cuál de los patrones de abajo representa correctamente el diseño del círculo.



Figura 5: Chase The Numbers

En la figura 5 podemos ver otro de los desafíos que nos proponen para ejercitarnos la memoria. Inicialmente nos muestran 12 números, pero estos desaparecen al cabo de unos segundos. Nuestro objetivo será desvelar la posición de cada número en el orden correcto (primero el 1, después el 2 y así hasta desvelar todos).

b) Bad News

Bad News es un juego creado por el estudio holandés “Gusmanson” en colaboración con la Universidad de Cambridge. En él nos ponemos en la piel de un propagandista que pretende crear un ejército de usuarios online utilizando las redes sociales con el propósito de difundir conspiraciones para influir en las opiniones públicas [13]. En la figura 6 podemos observar una de las pantallas de juego. En ella, se nos muestra un mensaje y el usuario ha de escoger si decide publicarlo o no en función de si cree que ese mensaje podrá influir en las opiniones públicas o no. Dependiendo de sus acciones, su número de seguidores aumentará, se mantendrá igual o decrementará. Y, a la vez, el

índicador de credibilidad variará en función de si el mensaje publicado se considera que podría ser real o muy ficticio.

Con este juego se pretende educar a sus jugadores sobre los peligros que conlleva creerse toda información que leemos cada día en las redes sociales sin antes haberla contrastado.

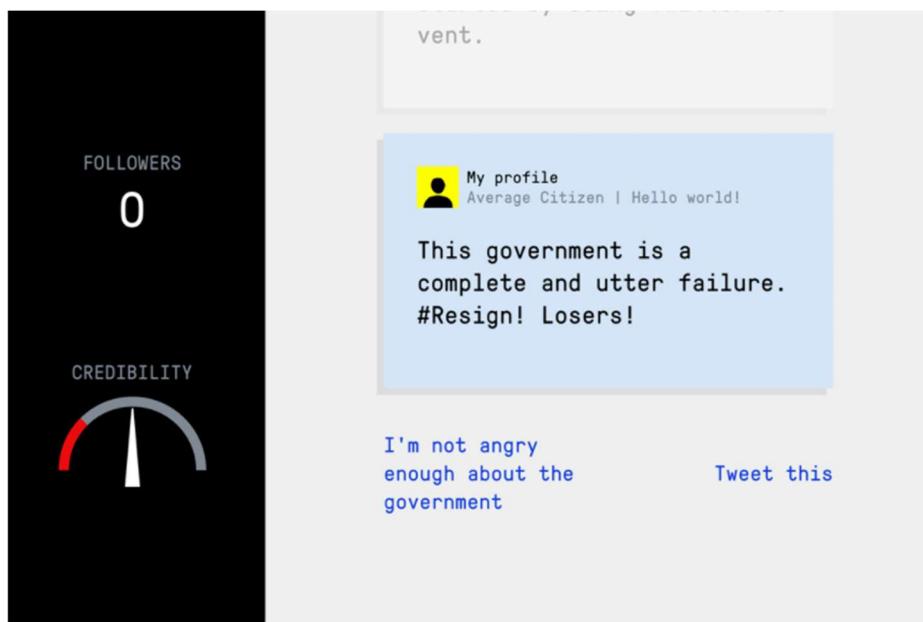


Figura 6: Bad News pantalla de juego

c) Cyber Defense Quiz

Este juego creado por la Universidad Carnegie Mellon (localizada en Pittsburgh, Estados Unidos) tiene como objetivo enseñar a los niños sobre peligros relacionados con el mundo de la ciberseguridad utilizando una temática de superhéroes [14]. Para ello, cómo se puede observar en la figura 7, el juego se basa en realizar preguntas quiz sobre estos temas y a medida que el jugador va acertando preguntas, irá desbloqueando nuevos personajes. El objetivo final es conseguir desbloquear a todos los superhéroes.



Figura 7: Cyber Defense Quiz pantalla de juego

d) Cyber Five

Cyber Five es un juego desarrollado por la compañía especializada en juegos educativos: ABCya. Este juego empieza con una pequeña animación dónde nos presentan a dos amigos con forma de animales (un hipopótamo y un erizo) que están aprendiendo sobre las normas que hay que seguir para navegar por internet de forma segura [15]. A la izquierda de la figura 8 podemos ver una parte de esta animación, y la derecha vemos una de las preguntas quiz que aparece una vez finalizada la animación para ver si los niños han aprendido algo nuevo.



Figura 8: Cyber Five, animación a la izquierda y pregunta quiz a la derecha

A continuación, en la tabla 1 se muestra una comparativa entre estos cuatro juegos educativos:

Juego	A Clockwork Brain	Bad News	Cyber Defense Quiz	Cyber Five
Puntos fuertes	Gran variedad de rompecabezas Jugable en más de un idioma	Dispone de una versión para adultos y otra adaptada a niños para que sea adecuado para todas las edades. Te pone en la perspectiva del enemigo	Gran variedad de preguntas. Se tratan varios temas sobre ciberseguridad, cómo ahora: contraseñas, ciber crimen y correos peligrosos.	Combina una parte de animación con preguntas quiz. Se puede regular la velocidad de la animación. Permite rehacer el quiz una vez terminado.
Puntos débiles	No disponible en versión web	Se podrían utilizar más imágenes para acompañar al texto	Jugable en un solo idioma. No explica porque una respuesta es incorrecta	Jugable en un solo idioma.
Nivel de dificultad	Medio-Alto (dependiendo del desafío)	Bajo	Medio	Bajo
Elementos que podría adaptar a mi juego	Perfil de un narrador que acompañe al jugador	Estética visual	Variedad en las preguntas quiz (enfocarlas a distintos temas relacionados sobre la privacidad)	Rehacer las preguntas quiz en caso de haberlas fallado

Tabla 1: Comparativa juegos educativos

3. REQUERIMIENTOS

Cuando se desarrolla un proyecto o sistema de software, un punto muy importante es definir los requerimientos funcionales y no funcionales.

3.1 Requerimientos funcionales

Los requerimientos funcionales se encargan de detallar las funcionalidades que ofrecerá el sistema.

El juego ha de disponer las siguientes funcionalidades:

- Ha de permitir seleccionar el idioma con el que se desea jugar.
- No se necesita un sistema de registro para ingresar al juego.
- Orientado a un solo jugador.
- Habrá distintos niveles, donde cada uno tendrá sus propias características para evitar que el juego sea repetitivo.
- Para cada nivel, el jugador podrá escoger con qué personaje desea jugar (la selección se hará por medio de una lista de personajes).
- Se ha de poder reiniciar el juego sin necesidad de cerrar y abrirlo de nuevo.
- Al finalizar la partida se ha de mostrar la puntuación obtenida junto a un ranking local con las mejores puntuaciones de partidas anteriores.

3.2 Requerimientos no funcionales

Los requerimientos no funcionales definen las restricciones y características del sistema.

Dado que la idea de elaborar este juego surge de la mano del proyecto Courage, hay algunos requerimientos que han de ir ligados con sus requisitos.

Courage, es un proyecto internacional llevado a cabo a por varias universidades con el objetivo de desarrollar un compañero virtual que se encargue de educar y apoyar a los adolescentes sobre cómo deben actuar frente a las amenazas que se encuentran cuando hacen uso de los medios sociales, como ahora, temas

relacionados con la discriminación racial o de género, discursos de odio, noticias falsas, acoso o contenido tóxico, entre otros [16]. Para ello, una de las actividades que se están realizando, es la elaboración de un aplicativo que permita la inserción de minijuegos educativos en una red social con el fin de concienciar a los usuarios sobre los peligros que uno se puede encontrar mientras navega por estos medios sociales a la vez que les ayuda a fortalecer su autoprotección ante esas amenazas. Este aplicativo, funcionará sobre PixelFed.



Figura 9: PixelFed logo

PixelFed (su logo correspondiente se encuentra en la figura 9) es una plataforma de código abierto¹ para compartir imágenes con otros usuarios, por lo que se trata de un medio social multimedia [17].

A diferencia de otros medios, este se caracteriza por:

- No disponer de anuncios.
- Sus publicaciones aparecen ordenadas de manera cronológica, nada de algoritmos que influyan en la manera en la que se muestran.
- No realizar un seguimiento de los datos privados de sus usuarios ni análisis de estos por parte de terceros.

En la figura 10 se muestra como es la interfaz de PixelFed:

¹ El código fuente es publicado bajo una licencia que permite a los usuarios la utilización, modificación y distribución del software para cualquier finalidad [18]

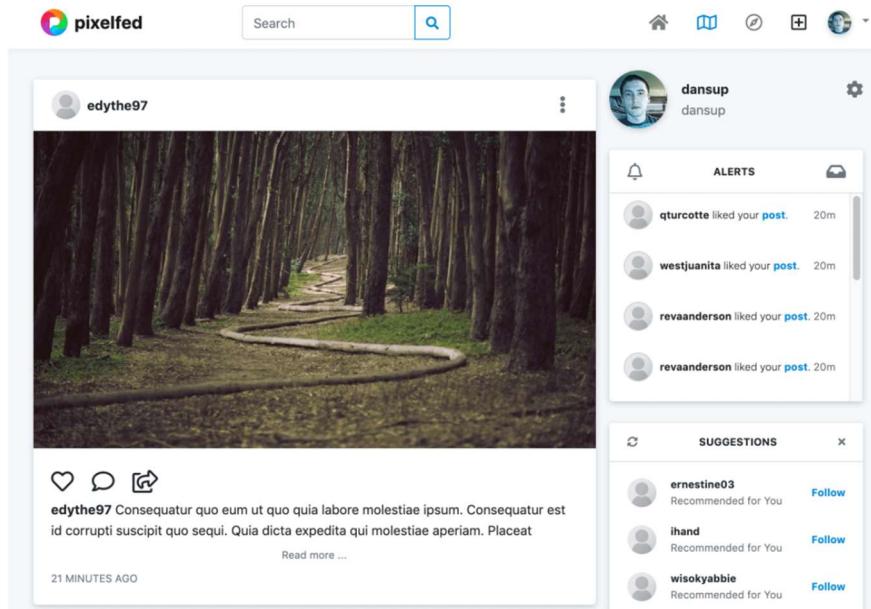


Figura 10: PixelFed interfaz

Otros requerimientos no funcionales que se han de tener en cuenta:

Relacionados con la **portabilidad** (cómo variará el sistema al ejecutarse en un entorno u otro):

- Puesto que el juego va a estar diseñado para ejecutarse en la web, debe funcionar tanto en versión de escritorio como en la versión móvil.
- Todos los elementos que aparecen en pantalla deben escalarse acorde con las dimensiones de la pantalla del dispositivo donde se está ejecutando (para evitar la distorsión de los elementos).

Relacionados con el **rendimiento** (cómo de rápido reaccionará el sistema):

- El tiempo de respuesta entre que el usuario realiza una acción y el sistema le responde, debe ser lo más rápido posible ya que de lo contrario, el usuario se podría cansar de esperar y por conseciente, abandonaría el juego.
- Todos los recursos audiovisuales que aparecen en el juego deben ser previamente cargados (en una fase de carga) para que cuando se ejecute el juego, se pueda hacer uso de ellos al instante.

Relacionados con la **fiabilidad** (cómo se comportará el sistema ante fallos):

- El sistema no debe de disponer de ningún error, pero en caso de que hubiera alguno, tendría que estar preparado para solventarlo en el menor tiempo posible y no repercutir en el flujo del juego.

Relacionados con la **usabilidad** (cómo de difícil es utilizarlo):

- La interfaz de usuario ha de ser intuitiva para evitar a toda costa que el usuario no sepa qué hacer en algún momento.

4. ANTECEDENTES TÉCNICOS

4.1 Software para la programación web

Antes de empezar con el diseño del minijuego, vamos a ver qué conjunto de herramientas necesitamos para desarrollar un proyecto orientado para la web.

a) HTML

El lenguaje HTML (Hyper Text Markup Language) es utilizado para la elaboración de páginas web. Hay que dejar claro que HTML no es un lenguaje de programación, sino que es un lenguaje de marcas. Es decir, combina datos y etiquetas (“tag”) que los marcan, ofreciendo así una información adicional sobre la presentación de cada uno de los datos que componen el documento [19].

Este tipo de lenguaje se caracteriza por dar una estructura organizada al contenido de la página web. En la figura 11 tenemos un pequeño ejemplo de un documento escrito en HTML.

```
<html>
  <head>
  </head>
  <body>
    <div>
      <p>Esto es un párrafo</p>
    </div>
  </body>
</html>
```

Figura 11: Ejemplo documento HTML

b) CSS

En 1996, la organización W3C (World Wide Web Consortium) desarrolló un nuevo lenguaje gráfico, conocido como CSS (Cascading Style Sheets). Este lenguaje permitió a los desarrolladores estilizar los elementos de un documento estructurado, como sería el caso de HTML [20]. Por ejemplo, al utilizar CSS podemos modificar el tipo de letra que se va a mostrar en la página web, cambiar los colores del fondo o de cada uno de los elementos presentes o incluso se podría llegar a crear animaciones con dichos elementos.

Hoy en día, es impensable no utilizar CSS cuando se decide trabajar con HTML y JavaScript, ya que, al fin y al cabo, el apartado visual de una página web es lo que más capta la atención del usuario. Para estilizar un elemento del documento escrito en HTML debemos seguir la estructura marcada en la figura 12.

```
body {  
    font: 'Arial';  
    background: blue;  
    margin: 0;  
}
```

Figura 12: Ejemplo documento HTML estilizado con CSS

En la figura 13 podemos apreciar una comparativa visual entre utilizar CSS y no utilizarlo:

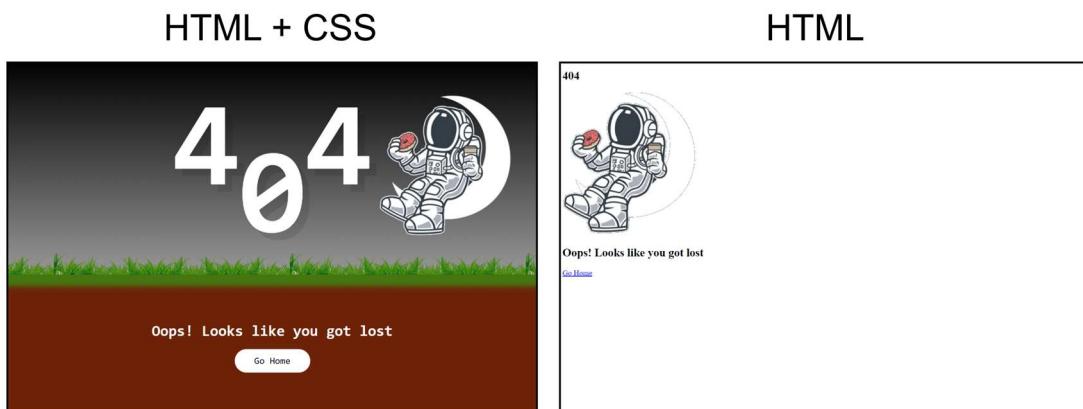


Figura 13: Página web utilizando CSS (izquierda) vs misma página web sin utilizar CSS (derecha)

c) JavaScript

JavaScript (JS) es un lenguaje de programación pensado para ser ejecutado en una página web. Su sintaxis es muy similar al lenguaje “C” pero a diferencia de este, no hay que indicar el tipo de las variables, por lo que es responsabilidad del programador controlar los valores que se asignan a cada una. Desde sus inicios, su nombre ha generado confusiones debido a su parecido con “Java”, pero en realidad no están relacionados.

JS es un lenguaje interpretado, es decir, el intérprete lee el código línea por línea y reacciona a ellas una vez ha llegado a dicha línea. Debido a esto, el código no

es tan rápido como si fuese compilado, pero nos ofrece algunas ventajas como: utilización de tipos dinámicos, puede ejecutarse en cualquier sistema operativo o permite modificar el código mientras se está ejecutando.

Al igual que otros lenguajes de programación, hay varias maneras de utilizar JS. Por un lado, podemos utilizar solamente JS, también conocido como “Vanilla JS”. En este caso, todo se programaría sin hacer uso de librerías externas, solamente utilizando JS. Esta técnica nos permite tener más control sobre nuestro código, pero a la vez nos hace tener que dedicarle más tiempo ya que todo lo tendremos que programar por nuestra cuenta, por lo que a la larga es menos eficiente y más costoso. Por otro lado, al igual que en el resto de los lenguajes de programación, existen librerías o “frameworks” (conjunto de herramientas) creados por otros usuarios o empresas que al añadirlas a nuestro proyecto nos permiten poder utilizar sus funciones dentro de nuestro código, ahorrándonos mucho tiempo y haciendo el código más eficiente y entendible para otros usuarios [21].

4.2 Librerías JS

Algunas de las librerías de JS más utilizadas a nivel mundial son:

a) jQuery



Figura 14: jQuery logo

Lanzado en 2006 por John Resig. La principal finalidad de jQuery (cuyo logotipo aparece en la figura 14) es hacer más fácil utilizar JavaScript dentro de la web. Para ello, jQuery contiene una inmensa cantidad de funciones que nosotros podemos llamar en cualquier parte de nuestro código ya sea para seleccionar elementos del DOM² (simplifica la sintaxis para encontrar, seleccionar y manipular dichos elementos), capturar eventos al realizar ciertas acciones, crear animaciones o simplemente para crear aplicaciones con Ajax³ [22]. En resumen, todo se logra escribiendo mucho menos en comparación de si

² DOM (“Document Object Model”): estructura en forma de árbol que contiene la representación de todos los elementos de una página web.

³ Ajax (“Asynchronous JavaScript and XML”): conjunto de técnicas para realizar aplicaciones orientadas a la comunicación cliente-servidor de manera asíncrona.

usáramos solamente JavaScript. De ahí su eslogan: “Escribe menos, haz más” (“write less, do more”).

De hecho, en enero de 2021, un 77'3% de todas las páginas web a nivel mundial estaban utilizando jQuery [23].

b) Node.js



Figura 15: node.js logo

Lanzado en el 2009 por Ryan Dahl. Node.js (su logotipo aparece en la figura 15) está orientado a crear aplicaciones para la red, pero del lado del servidor. De manera que es el servidor quien ejecuta las comandas escritas con JS para crear una página web dinámica antes de que esta sea mandada al usuario que la está solicitando des de su navegador.

4.3 Frameworks JS

En el ámbito del desarrollo de software, con “framework” nos referimos a un entorno de trabajo con una estructura predefinida para facilitar el desarrollo de una aplicación.

A continuación, vamos a ver los “frameworks” de JS más conocidos para ver cuál se adapta mejor a nuestras necesidades:

a) Angular



Figura 16: Angular logo

Angular (cuyo logotipo aparece en la figura 16) fue lanzado en el 2010 por Google y se caracteriza por permitir crear una aplicación donde todos sus componentes son encapsulados en una sola página, también conocido como “Single Page Application”. De tal manera que, al clicar en enlaces de la página principal, la página en si no se recargará completamente, sino que actualizará los datos en la propia página. Permitiendo así crear contenidos dinámicos y rápidos como si fuera una aplicación de escritorio en vez de una página web. Al ser un “framework” flexible,

es muy sencillo de integrar con otras librerías y cada una de sus características pueden ser modificadas para que se adapten mejor a nuestro proyecto [24].

b) React

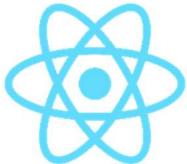


Figura 17: React logo

Lanzado en el 2013 por Facebook, React (con su respectivo logotipo en la figura 17) permite crear interfaces de usuario interactivas complejas a partir de la composición de distintos componentes (pequeños trozos de código independientes). Es muy similar a Angular, pero React utiliza el DOM virtual, de manera que el rendimiento del framework mejora considerablemente con respecto al uso del DOM real [25].

c) VueJS



Figura 18: VueJS logo

A diferencia de los frameworks Angular y React, Vue (con su logotipo en la figura 18) no fue desarrollado por una empresa sino por un ex ingeniero de Google llamado “Evan You” durante el 2014. Al ser de los frameworks más recientes, no dispone de tanta popularidad entre los programadores como el resto, pero cada vez son más los que lo utilizan ya que facilita mucho el desarrollo para los Front End a la vez que mejora algunas funcionalidades de React y Angular. Al igual que React, utiliza el DOM virtual. Uno de los puntos fuertes de Vue, es que dispone de una inmensa variedad de librerías y una gran comunidad que lo hacen todo más fácil. Al fin y al cabo, Vue nació con la idea de ayudar a los desarrolladores a crear interfaces de usuario de manera muy sencilla, pero con un resultado increíble. Y hay que decir que es muy fácil de aprender y utilizar [26].

Dado que para este proyecto se estaba buscando un framework para crear una “Single Page Application” donde el usuario pudiera interactuar con la página, se ha optado por utilizar VueJS por las siguientes razones además de lo comentado previamente [27]:

- El DOM virtual ofrece un mayor rendimiento
- Utiliza vanilla JS

- Permite crear un código muy bien estructurado
- En comparación con React y Angular ocupa muy poco espacio por lo que reduce significativamente los tiempos de carga y velocidad en la web, y esto para un juego es un aspecto fundamental.

4.4 Creación de assets para el juego

Todo juego necesita un apartado estético que lo represente ante sus competidores, es ahí donde entran los assets. Con “asset” nos referimos a todos los recursos audiovisuales que utiliza un videojuego ya sean imágenes, videos, audios o sprites [28].

Si nos limitamos a utilizar assets ya existentes, nuestro juego no solo acabará siendo muy similar a otros juegos (habrá más gente que los habrá utilizado), sino que estaremos muy limitados al contenido que haya creado un diseñador puesto que combinar assets de distintos creadores es muy probable que acabe por romper la coherencia estética de nuestro juego. Es por esta razón que es mejor crear nuestros propios recursos, ya que de esta manera disponemos de total libertad para crear cualquier elemento que necesitemos y a la vez nos aseguramos de que todos ellos tengan un estilo muy similar.

Para elaborar los distintos personajes, objetos y el resto de los elementos que aparecen en el juego se han utilizado las siguientes herramientas:

a) Piskel

Piskel o también conocido como PiskelApp, es un editor web de sprites que nos permite crear sprites para videojuegos al más puro estilo píxel art (imágenes editadas píxel a píxel) de los juegos de la década de los 80 [29].

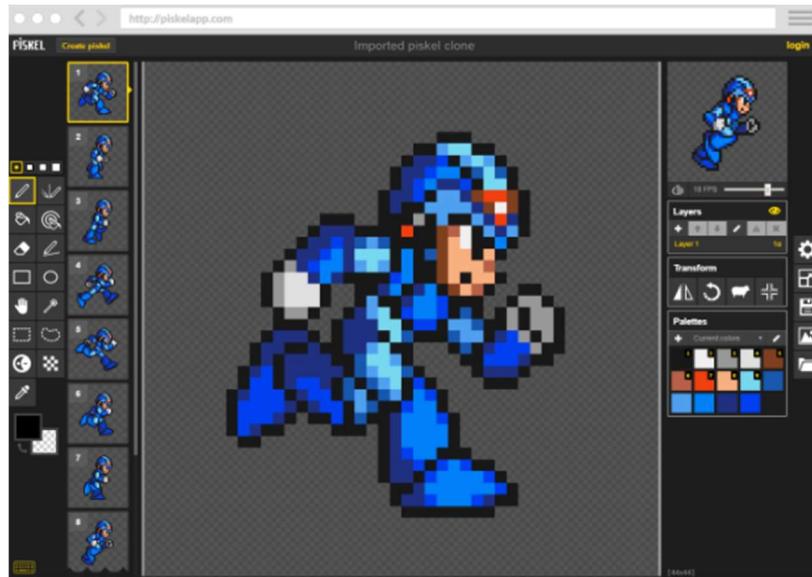


Figura 19: Piskel

En la figura 19 se muestra la interfaz de la página Piskel. En ella vemos que en la parte izquierda están las herramientas para editar la imagen que aparece en la parte central, seguido de todo el conjunto de imágenes que componen la animación.

b) Tiled

Tiled es un editor de niveles para videojuegos 2D donde el mapa es generado a partir de celdas [30]. Si observamos la figura 20 podemos ver en la parte izquierda el mapa dividido en celdas (10 celdas de altura x 10 celdas de anchura) donde cada una de estas celdas corresponde a una de las imágenes que aparecen en la parte inferior derecha. A este conjunto de imágenes se le conoce como “tileset”.

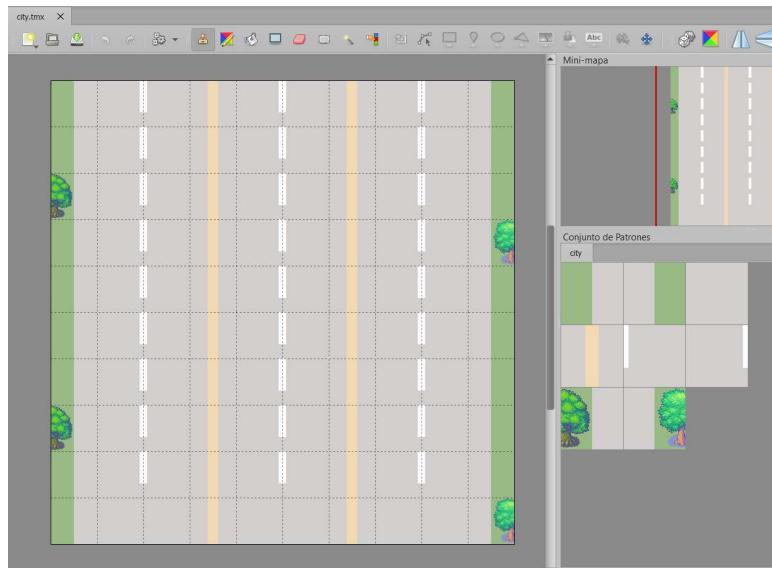


Figura 20: Tiled

c) Figma

Figma es una aplicación web para diseñar interfaces de usuario (a partir de la edición de gráficos vectoriales), crear esquemas de páginas (también conocidos como “wireframes”, son unas guías que representan la estructura visual de una página web) y prototipados (permite visualizar como quedarían los diseños en una página web tanto en versión de escritorio como de dispositivo móvil).

Una de las funcionalidades por las que destaca es que permite la colaboración grupal en tiempo real (más de un usuario puede editar a la vez sobre un mismo proyecto y el resto verían los cambios al instante) [31].

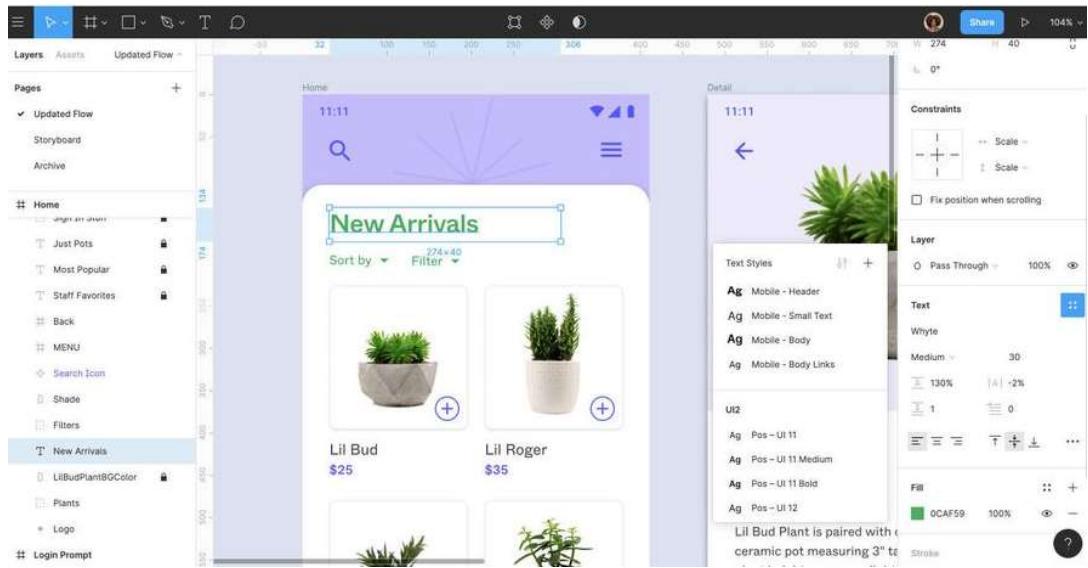


Figura 21: Figma Interfaz

En la figura 21 podemos ver la interfaz de Figma, donde en la parte izquierda encontramos todos los componentes que forman parte del wireframe que se está diseñando en la zona central y en la derecha tenemos otro panel para editar las propiedades del componente seleccionado.

5. SISTEMA CENTRAL DEL JUEGO

5.1 Personaje / Jugador

a) ¿Cómo es el personaje que protagoniza el juego?

El juego no estará limitado a un solo personaje, sino que habrá varios y será el propio jugador el que elija con cuál de ellos quiere jugar al nivel seleccionado. Cada uno de estos personajes representará a un tipo de usuario que podemos encontrar en los medios sociales y a su vez, dispondrá de unas ciertas habilidades que le otorgarán funcionalidades extras en el juego.

Si en un juego se tiene todo desbloqueado desde un principio sí que es verdad que al inicio tiene mucha emoción ya que puedes hacer lo que quieras, al fin y al cabo, ya lo tienes todo. Pero se pierde ese factor que hace que el usuario siga jugando para desbloquear nuevos niveles, personajes o recompensas. Así que, por esta razón, no todos los personajes estarán disponibles des del principio, sino que se irán desbloqueando poco a poco, a medida que el usuario vaya progresando a la vez que va aprendiendo puesto que los personajes más avanzados representan a un usuario con mayor conocimiento (a mayor conocimiento, mejores serán las habilidades extras del personaje).

A continuación, se muestran los distintos personajes junto a una representación visual de cada uno de ellos:

- **Chico básico:** este será el personaje masculino más básico de todos, por lo que no dispondrá de ninguna habilidad especial ya que su finalidad principal será que el jugador aprenda los controles básicos del juego. Con él se pretende representar a aquel usuario inexperto que entra por primera vez en un medio social y no dispone de conocimientos suficientes sobre lo que debe y no debe publicar en la red puesto que desconoce los riesgos que supone公开其私人信息 to the exterior world and the effects that this could have in the hands of incorrect individuals.



Figura 22: Chico básico Spritesheet

En la figura 22 podemos ver el conjunto de imágenes con el que este personaje será representado en el juego. Cada una de las filas de imágenes que componen la figura corresponden a un conjunto de animaciones que recrean una acción realizada por el propio personaje. En la primera fila encontramos la animación que realizaría el personaje al caminar de frente, en la segunda hacia la izquierda, en la tercera hacia la derecha y finalmente en la cuarta lo vemos de espaldas.

- **Chica básica:** puesto que el juego está orientado a todos los públicos, es necesario incluir personajes de todos los géneros con tal evitar cualquier tipo de discriminación. Así que Chica básica será la versión femenina de Chico básico por lo que compartirán las mismas características y funcionalidades, en la figura 23 tenemos su representación visual.



Figura 23: Chica básica spritesheet

- **El Sabio** (Figura 24): a diferencia de los personajes anteriores, esté poseerá la habilidad especial del conocimiento. De manera que ayudará al usuario en las preguntas tipo quiz descartando una de las 3 opciones incorrectas, así el usuario sólo tendrá que escoger entre 3 opciones y no 4. Simulando que el jugador ya es un veterano de los medios sociales y controla mejor sobre lo que debe y no debe publicar, así que dispone de un mayor conocimiento. En el apartado 6 “Diseño”, veremos con más detalle en que consiste este tipo de preguntas y cuándo aparecerán.



Figura 24: El Sabio spritesheet

- **Punky** (Figura 25): muy similar al anterior, este nuevo personaje nos volverá a ayudar en el quiz, pero ahora, en vez de descartar una de las opciones, nos ofrecerá 10 segundos extra. De manera que el usuario dispondrá de mayor tiempo para resolver la pregunta propuesta.



Figura 25: Punky spritesheet

- **Chica de Hielo** (Figura 26): este personaje tendrá la habilidad especial de poder ralentizar el tiempo durante un breve periodo de tiempo de tal manera que el usuario dispondrá de más tiempo para analizar el entorno puesto que todo lo que suceda en el mundo del juego verá su velocidad reducida. Con este personaje se pretende representar a un usuario ya más experto en el mundo de los medios sociales.



Figura 26: Chica de hielo

A continuación, en la tabla 2 se muestra a modo de resumen las habilidades de cada uno de los personajes junto al número de vidas máximo del que puede disponer (los personajes más sencillos tendrán un mayor número de vidas mientras que aquellos que disponen de alguna habilidad tendrán menos).

Nombre personaje	Habilidad	Número máximo de vidas
Chico básico	Ninguna	5
Chica básica	Ninguna	5
El sabio	Descartar una opción en el quiz	3
Punk	10 segundos extras en el quiz	3
Chica de hielo	Ralentizar velocidad del juego	3

Tabla 2: Características de los personajes

b) Reglas de acción

En este apartado se detallan todas las acciones posibles del personaje (reglas de acción-efecto):

- Movimiento: el personaje se podrá mover por el mapa utilizando el ratón o si estamos en un dispositivo móvil, pulsando la pantalla. Si clicamos a una zona que esté situada a la derecha del personaje, este se moverá en esa dirección, y si clicamos en una zona a la izquierda, el personaje se moverá a la izquierda. El movimiento estará limitado al eje de las 'x', así que solo se podrá mover hacia los lados.
- Colisión con los obstáculos: si el jugador colisiona con un objeto, su vida se verá afectada.
- Colisión con los recursos: si el jugador recoge algún recurso, se realizará una acción especial acorde con el recurso obtenido (se detalla en el apartado 5.3).

c) Reglas de estado

En este apartado se detallan todas las variables de estado:

- Vida: el jugador dispondrá de un número limitado de vidas que se verá reducido en una unidad cada vez que colisione con un obstáculo. Dependiendo del personaje que estemos utilizando, el número de vidas máximo del que se podrá disponer será mayor o menor.
- Puntuación: el objetivo del juego es lograr la máxima puntuación posible. Cada vez que el jugador logre esquivar un obstáculo, ganará un punto. También, como veremos más adelante, será posible obtener puntos adicionales si se responde correctamente a la pregunta quiz que aparecerá en determinados momentos del juego.

5.2 Reglas de control

Si el jugador colisiona con muchos obstáculos y su vida llega a 0, morirá y el juego habrá terminado. En este momento, el juego nos conducirá a la pantalla

de resultados dónde se mostrará los puntos que ha obtenido (tanto en el juego principal como en los bonus) y el tiempo que ha estado jugando. Una vez leídos los resultados, el jugador podrá reiniciar el nivel y volver a jugar, o si lo prefiere, será redireccionado a la selección de niveles para que escoja un nuevo nivel.

5.3 Mundo del juego

Ya hemos visto quien protagonizará nuestro juego, ahora nos queda por ver el resto de los elementos que componen el mundo del juego, es decir, los personajes no jugadores (PNJs), obstáculos, recursos y escenarios disponibles.

a) Componentes del mundo del juego



Figura 27: Narrador

Siguiendo en la línea de los personajes, encontramos a un PNJ que actuará como **narrador** del juego. Su función será muy simple, indicar al jugador lo que ha de hacer en cada una de las pantallas de juego para que este no se bloquee y no sepa que hacer. En la figura 27 podemos ver el aspecto que tendrá en el juego.

Como se ha comentado previamente, el objetivo del juego es que el jugador evite ciertos **obstáculos** que se irá encontrando. Estos obstáculos, serán una representación visual de todo aquello que un usuario no debería publicar en los medios sociales bajo ningún concepto, ya que, de hacerlo, estaría revelando información personal de gran valor ante millones de personas.

A continuación, se detallan cada uno de los obstáculos junto a su representación visual dentro del juego y a los peligros que se expondría el usuario de publicarlos [32]:

Identificación personal:

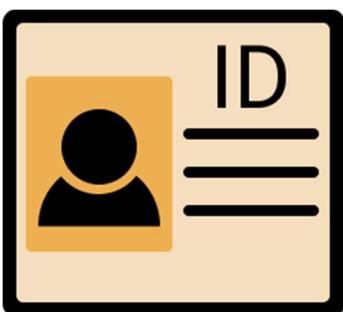


Figura 28: Obstáculo ID

Cualquier tipo de identificación personal, ya sea el DNI, el permiso de conducir o pasaporte nunca deberían publicarse puesto que contienen información personal que uno debería guardarse para sí mismo. Este tipo de documentos suelen incluir el nombre completo del usuario, fecha de nacimiento y la dirección de su casa. Entre los

delitos más frecuentes que se realizan una vez se obtiene esta información, encontraríamos el acoso o la suplantación de identidad. Por ejemplo, un ciberdelincuente podría crearse un perfil falso en una red social con nuestros datos y hacerse pasar por nosotros. En la figura 28 podemos ver la representación visual de este obstáculo, simulando una tarjeta de identificación.

Contraseña:



Figura 29: Obstáculo Contraseña

Las contraseñas se utilizan para bloquear el acceso libre a una cuenta, de manera que solo su propietario pueda acceder a ella.

Para evitar que descubran nuestras contraseñas se recomienda utilizar un mínimo de 8 caracteres con una combinación de números, letras (minúsculas y mayúsculas) y caracteres especiales (como ahora “-”, “\$”, “%”), y evitando utilizar datos personales como la fecha de nacimiento o el nombre de la mascota, entre otras. En caso de incumplir estas recomendaciones, la contraseña sería catalogada como una contraseña débil, y esto se lo pondría fácil a los ciberdelincuentes para intentar descubrirla utilizando la fuerza bruta (probar todas las combinaciones posibles) o haciendo uso de diccionarios que contengan una lista larga con las contraseñas más utilizadas.

Por tanto, para evitar que alguien se haga con el acceso de nuestras cuentas, se debería utilizar una combinación fuerte de caracteres y evitar compartirla con más gente ya que es habitual que se utilice la misma contraseña en distintas páginas, así que si alguien logra acceder a una de nuestras cuentas es muy

probable que acceda al resto. De ahí la importancia de mantenerlas siempre en secreto.

Como muestra la figura 29, este obstáculo será representado con un candado junto a los asteriscos típicos que aparecen al escribir una contraseña.

Tarjeta de crédito:

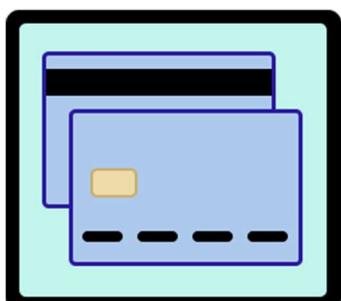


Figura 30: Obstáculo Tarjeta de crédito

Algunos adolescentes al obtener su primera tarjeta de crédito deciden compartirlo en las redes sin tener en cuenta que todo aquel que tenga su número de la tarjeta y la fecha de caducidad pueden comprar muchas cosas por internet sin su consentimiento. Así que para evitar que nos vacíen todo el dinero de nuestra cuenta bancaria, es mejor que mantengamos en secreto todas nuestras tarjetas de crédito. En la figura 30 podemos apreciar la representación de una de estas tarjetas por ambas caras.

Datos bancarios:



Figura 31: Obstáculo datos bancarios

En la línea de las tarjetas de crédito, no se debe publicar cuánto dinero tiene uno en su cuenta ya que podría convertirse en una víctima para los ladrones o ciberdelincuentes que quieran hacerse con su cuenta. En la figura 31 vemos que para representar los datos bancarios se ha utilizado el símbolo del dólar (\$) junto a un gráfico de barras (simulando los gastos e ingresos).

Información detallada sobre tus vacaciones:



Figura 32: Obstáculo Vacaciones

Cuando alguien se va de vacaciones, es muy habitual que cuelgue imágenes en las redes sociales para presumir ante sus seguidores sobre lo que está haciendo. Podríamos pensar que estas imágenes solo las ven sus seguidores, pero en realidad no sabemos a quién está llegando esta información, así que si alguien sabe dónde vives, podría aprovechar tu ausencia para entrar a robar a tu casa. Una forma sencilla de evitar este riesgo sería esperar a regresar de tu periodo de vacaciones para colgar dichas imágenes. Como muestra la figura 32, las vacaciones han sido representadas por una sombrilla ya que es la clara expresión de verano, y arriba a la izquierda aparece la "i" de información.

Dirección de casa:



Figura 33: Obstáculo Dirección de casa

Muy ligado al anterior. Al compartir la dirección de casa, se les está abriendo la puerta a ladrones. Puesto que podrían aprovechar nuestra ausencia para entrar a robar. Pero no solo eso, sino que también podría hacer que los acosadores se acerquen a la vivienda.

Para representar este obstáculo se ha utilizado el icono de ubicación junto a la silueta de una casa (Figura 33).

Números de teléfonos:



Figura 34: Obstáculo Número de teléfono

Compartir nuestro número de teléfono en Internet puede tener más consecuencias de las que uno se imagina. Por un lado, podríamos sufrir spam telefónico (llamadas continuas de personas desconocidas o empresas con las que no queremos hablar, suelen llamar para intentar vender sus productos o intentar estafarnos). Pero también podría llevar a la suplantación de

identidad. En la figura 34 podemos ver que este obstáculo será representado por el icono de un teléfono junto a los puntos equivalentes a los diez dígitos que pueden aparecer en un número de teléfono.

Por otro lado, también habrá unos objetos o **recursos**, que, de ser recolectados por el jugador, se realizaran ciertas acciones especiales:

Vida:



Figura 35: Recurso Vida

Tal y como indica su nombre, este recurso (Figura 35) servirá para otorgar una vida extra al jugador. Por lo que, al ser recogido, incrementará en una unidad el número de vidas del jugador (siempre y cuando no exceda del número máximo de vidas del que puede disponer cada uno de los personajes).

Quiz:



Figura 36: Recurso Quiz

Cuando el jugador recoja este recurso (Figura 36), se mostrará una pantalla con una pregunta de tipo de quiz. Si se acierta la pregunta se obtendrán puntos adicionales, de lo contrario, se quedará como estaba (en el apartado de “Diseño” se explicará con más detalle cómo será la pantalla encargada de mostrar estas preguntas).

b) Diseño de niveles

Como se ha comentado previamente, habrá distintos niveles donde cada uno presentará un reto para el usuario:

- **Nivel 1 Clásico:** será el nivel más simple de todos puesto que se pretende que el usuario aprenda los fundamentos básicos del juego. En él los obstáculos se moverán en línea recta y podrán aparecer en 3 puntos del mapa, tal y como muestra la figura 37.

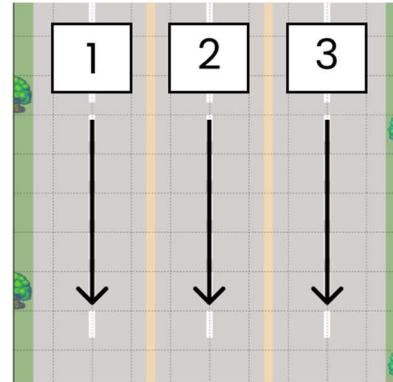


Figura 37: Representación nivel 1

- **Nivel 2 Oscilando:** presenta una estructura similar al nivel 1 pero ahora los obstáculos seguirán un movimiento oscilatorio con el fin de aumentar la dificultad con respecto al nivel anterior y podrán aparecer en 2 puntos, cómo muestra la figura 38.

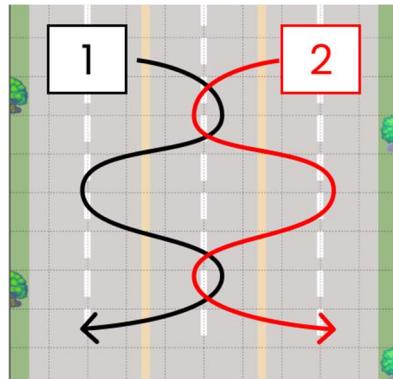


Figura 38: Representación nivel 2

- **Nivel 3 Más obstáculos:** se añade una columna más por donde pueden venir los obstáculos, lo que implica mayor número de obstáculos y un incremento de la dificultad. En la figura 39 tenemos la representación de este nivel.

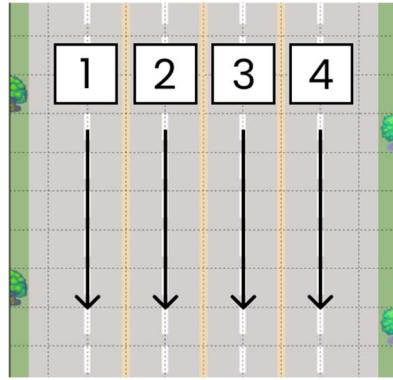


Figura 39: Representación nivel 3

c) Punto de vista y cámara

Escoger el tipo de cámara que se va a utilizar es uno de puntos más importantes a la hora de hacer un juego ya que utilizar una cámara u otra puede cambiar por completo la sensación que percibe el jugador. Para evitar escoger una cámara

incorrecta, veamos algunas de las técnicas que más se han utilizado para desarrollar juegos en 2D [33]:

- **Cámara bloqueada en el eje horizontal:** la cámara se mueve siguiendo al personaje en el eje de las “x”. Un claro ejemplo de juego que utiliza esta técnica es “Super Mario Bros” (Figura 40):



Figura 40: Super Mario Bros (NES) 1985, Nintendo

- **Cámara bloqueada en ambos ejes:** la cámara sigue al personaje en los ejes “x” e “y”, de manera que el jugador siempre permanece en el centro de la pantalla. En el juego “Terraria” (Figura 41), dado que el personaje es muy pequeño y puede moverse en todas direcciones a gran velocidad, esta cámara funciona muy bien y a la vez ofrece una amplia visibilidad para que el jugador visualice el entorno.



Figura 41: Terraria 2011, Re-logic

- **Cámara fijada en una casilla:** la cámara se mueve por una cuadrícula cuando el jugador desaparece de su campo de visión. Por ejemplo, en “The Legend of Zelda” (Figura 42), el mapa está compuesto por casillas/regiones y la cámara enfoca a la región en la que se encuentra el

personaje. Este puede moverse libremente por esta región, sin embargo, cuando desaparece de este campo de visión, la cámara se moverá hacia la celda a la que se ha movido.

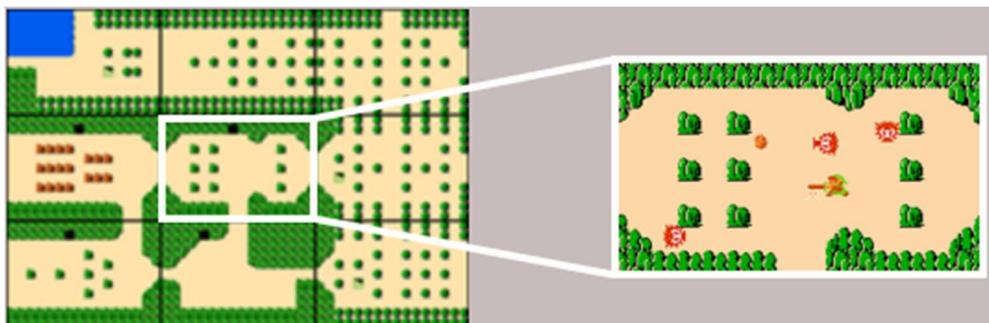


Figura 42: The Legend of Zelda (NES) 1986, Nintendo

- **Cámara controlada por una ventana:** el jugador está rodeado por una ventana transparente y al colisionar con uno de los bordes, es cuándo la cámara se mueve en esa dirección. En todos los juegos de “Sonic” (Figura 43) utilizan esta técnica:



Figura 43: Sonic the Hedgehog 1991, Sega

- **Cámara de desplazamiento automático:** el jugador no tiene control sobre la cámara, esta se mueve de manera constante y automática hacia un eje puesto que ha de simular que el entorno se va moviendo. Suele posicionar al jugador en un extremo dejando una amplia visibilidad sobre lo que se acerca hacia él por delante. Por ejemplo, en el juego “Scramble” (Figura 44), la nave espacial puede moverse libremente por todo el campo de visión, pero la cámara se va desplazando a velocidad constante hacia la derecha.



Figura 44: Scramble 1981, Konami

Dado que para mi juego estaba buscando crear una escena donde el personaje simule que está corriendo hacia delante y a medida que avanza se va encontrando con una serie de obstáculos, la cámara que mejor se adaptaba era la de desplazamiento automático.

6. DISEÑO

6.1 Wireframes

Una vez tenemos claro los elementos que aparecerán en el juego, ha llegado el momento de empezar a diseñarlo. Para ello, se han elaborado una serie de “wireframes” para esquematizar el diseño de cada una de las pantallas que componen el juego sin entrar en detalle en el aspecto visual dado que su principal objetivo reside en mostrar la funcionalidad, el comportamiento y la jerarquía de contenidos que aparece en cada una de las pantallas [34].

Pantalla inicial



La figura 45 expone la primera pantalla del juego, en ella se mostrará una imagen de fondo del juego junto al título de este. Con el fin de lograr captar más la atención del usuario, irá acompañada de una música de fondo.

Al pulsar el botón de “JUGAR”, el jugador será redirigido a la siguiente pantalla para que pueda empezar a jugar al juego.

Figura 45: Pantalla inicial

Selección de idioma

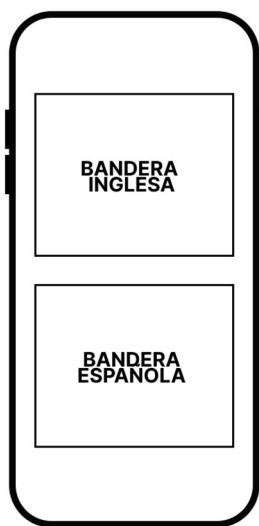


Figura 46: Pantalla selección de idioma

La pantalla representada en la figura 46 servirá para seleccionar el idioma con el que el usuario deseará jugar al juego.

Habrá 2 idiomas disponibles: inglés y español. Cada uno de ellos estará representado por una imagen de la bandera de uno de los países dónde se habla este idioma. En el caso del inglés, se utilizará la bandera de Reino Unido y para el caso del español, la de España. Además, el nombre del idioma aparecerá sobre la bandera.

Al pulsar sobre una de las banderas, el idioma quedará seleccionado y a continuación, el usuario será redirigido a la siguiente pantalla.

Selección de niveles



Figura 47: Pantalla selección de niveles

En la pantalla mostrada en la figura 47, se presentarán los distintos niveles en forma de “slider”⁴.

Cada nivel será representado por una especie de carta que contendrá una imagen representativa del nivel junto a una breve descripción de este. Para cambiar de nivel, el usuario tendrá 2 opciones:

- Deslizar las cartas hacia el lado deseado.
- Clicar en los botones de la parte inferior.

Al clicar en el botón “SELECCIONAR”, el nivel será seleccionado y se pasará a la siguiente pantalla: la selección de los personajes.

⁴ Elemento que muestra varios contenidos, ya sean imágenes o textos que se van alternando entre ellos

Selección de personajes

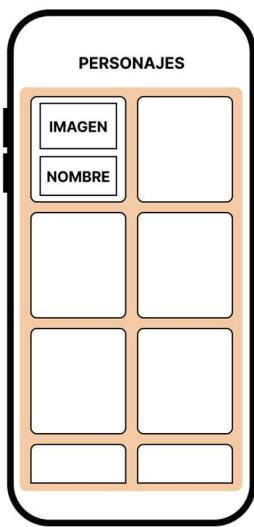


Figura 48: Pantalla selección de personajes

La figura 48 presenta la pantalla donde se mostrarán los distintos personajes con los que se podrá jugar.

Cada una de las casillas hará referencia a un personaje y en ella se mostrará una imagen del personaje junto a su nombre.

Al clicar sobre una de estas casillas, se pasará a la siguiente pantalla con el fin de ver información adicional sobre este personaje.

Información detallada sobre el personaje

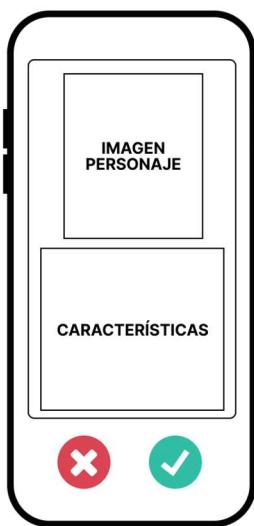


Figura 49: Pantalla información detallada personaje

La pantalla de la figura 49 actuará como si fuese un Pop Up⁵ y nos mostrará la siguiente información sobre el personaje:

- Qué tipo de usuario (que podemos encontrar en los medios sociales) representa.
- Habilidades especiales de las que dispone que nos ayudarían en el juego.

Para descartar el personaje y volver a la pantalla anterior (selección de personajes), el usuario deberá clicar en el botón con una cruz, y en caso de que desee seleccionar el personaje y proceder con el juego, deberá

clicar al botón con la marca de verificación.

⁵ Ventana emergente que aparece encima del contenido que estamos visualizando

Juego principal

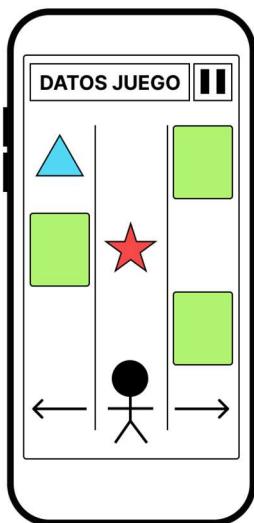


Figura 50: Pantalla juego principal

Una vez el nivel y el personaje han sido seleccionados, pasamos a la pantalla del juego principal, representada en la figura 50. En la parte superior izquierda de esta encontramos información sobre la partida (se muestran los puntos obtenidos y las vidas restantes que nos quedan) y en la parte derecha encontramos un botón que de ser pulsado nos redirigirá a la pantalla con el menú de pausa.

Debajo de la barra de información y del botón, será donde sucederá la acción del juego. Por un lado, tendremos a nuestro personaje situado en la parte inferior de la pantalla que deberá sortear los distintos obstáculos que se le van aproximando, para ello deberá moverse hacia uno de los lados. En la figura, estos obstáculos están representados por los cubos de color verde. Además de los obstáculos, también habrá unos objetos especiales que de ser recogidos darán la oportunidad al jugador de ganar puntos adicionales.

Pregunta Quiz

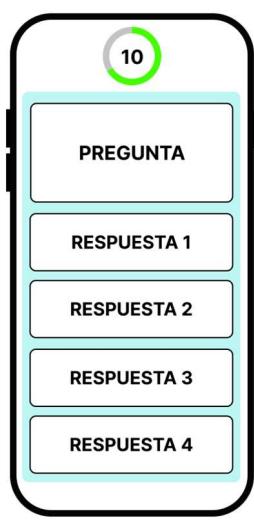


Figura 51: Pantalla pregunta quiz

Llegará un momento en el juego en el que el jugador cogerá un objeto que lo redirigirá a esta pantalla (Figura 51). Aquí se mostrará una pregunta junto a 4 posibles respuestas donde solo una de ellas será la correcta. Dado que se pretender que el jugador mejore su agilidad mental, habrá un límite de tiempo para responder a esta pregunta. En caso de no contestarla dentro del límite establecido, se contará como respuesta incorrecta.

Para seleccionar una respuesta, simplemente se tendrá que clicar sobre ella. A continuación, se volverá a la pantalla del juego y en caso de acertar la pregunta, se aumentarán los puntos del juego.

Menú de pausa

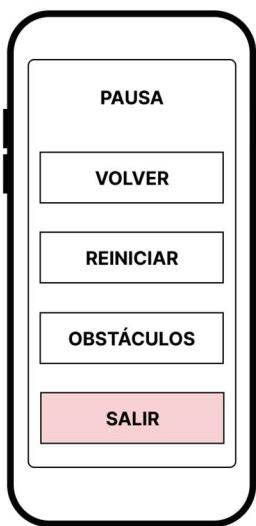


Figura 52: Pantalla menú pausa

Cuando en el juego principal se pulse el botón de pausa, se nos redirigirá a esta pantalla (Figura 52) y el juego quedará pausado.

En la parte central habrá 4 botones, cada uno con una funcionalidad distinta:

- Volver: volver al juego y retomar la partida donde la dejamos antes de acceder al menú de pausa.
- Reiniciar: reiniciar el nivel en las mismas condiciones con las que llegamos a esta pantalla.
- Obstáculos: acceder a un panel donde se muestran todos los obstáculos que aparecen en el juego.
- Salir: dar por finalizado el juego y proceder a la pantalla de resultados.

Lista de obstáculos

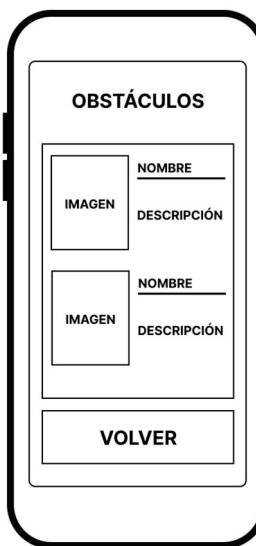


Figura 53: Pantalla menú obstáculos

Al clicar en el botón de “Obstáculos” del menú de pausa seremos redirigidos a este nuevo panel (Figura 53), donde podremos observar una lista con todos los obstáculos que aparecen en el juego. Estos se mostrarán con la imagen que los representa junto a su nombre y una breve descripción del peligro que le supondría al usuario en caso de que los publicara en los medios sociales.

Al clicar al botón de “Volver” regresaremos al menú de pausa.

Resultados



Figura 54: Pantalla de resultados

Una vez terminado el juego, aparecerá la pantalla representada por la figura 54. En ella se podrá ver el tiempo activo de juego, la puntuación obtenida tanto en el juego principal como en la parte de bonus y la puntuación final (será la suma de las dos puntuaciones anteriores).

En la parte inferior habrá 2 botones:

- Reiniciar: permitirá reiniciar el nivel en las mismas condiciones con las que se ha jugado, es decir, mismo nivel y personaje.
- Volver: volver a la selección de niveles para escoger un nuevo nivel y personaje.

En la parte superior derecha habrá un botón que de ser pulsado nos conducirá a la pantalla de Ranking.

Ranking

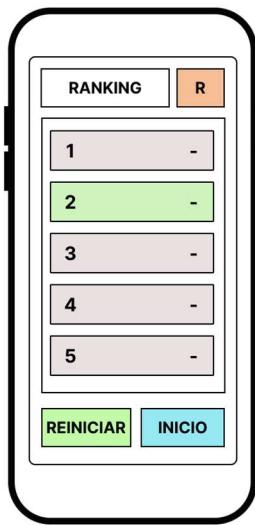


Figura 55: Pantalla de ranking

Esta pantalla será muy similar a la de los Resultados, pero cambiará su contenido central (Figura 55). En él se nos mostrará los puntos que se han obtenido en partidas anteriores ordenados de manera descendente (solo se mostraran los 5 primeros). En caso de que el jugador haya logrado una marca que supere a algunos de los 5 mejores, aparecerá en este ranking. Para facilitar al usuario localizar la posición en la que ha quedado respecto al ranking, su casilla se marcará de un color distinto al resto.

Los botones que aparecen en esta pantalla tendrán las mismas funcionalidades que los de la pantalla de resultados, con la única diferencia, que ahora al pulsar el botón superior derecho, seremos redirigidos de nuevo a la pantalla anterior, es decir, a la de resultados.

Instrucciones por parte del narrador



Figura 56: Pop Up narrador

El “wireframe” expuesto en la figura 56 no representa a una pantalla en concreto sino más bien a un elemento que aparecerá sobre las pantallas anteriores:

- Selección de niveles
- Selección de personajes
- Juego principal

Su finalidad será poner en contexto la pantalla a la que precede, de manera que el usuario entienda lo que habrá de hacer en esa pantalla. Es aquí donde entra en juego el PNJ Narrador.

En la parte izquierda se mostrará la imagen del narrador para que el jugador entienda quien le está dando las instrucciones, las cuáles aparecerán en la parte derecha. Al clicar al botón de Cerrar, la ventana emergente se cerrará y se mostrará la pantalla del fondo.

6.2 Diagrama de flujo

El diagrama de flujo de la figura 57 nos muestra las distintas acciones que se han de realizar para acceder a cada una de las pantallas previamente definidas.

Donde la primera pantalla en mostrarse seria la “Pantalla inicial” y la última la “Pantalla de resultados” o “Pantalla de ranking”.

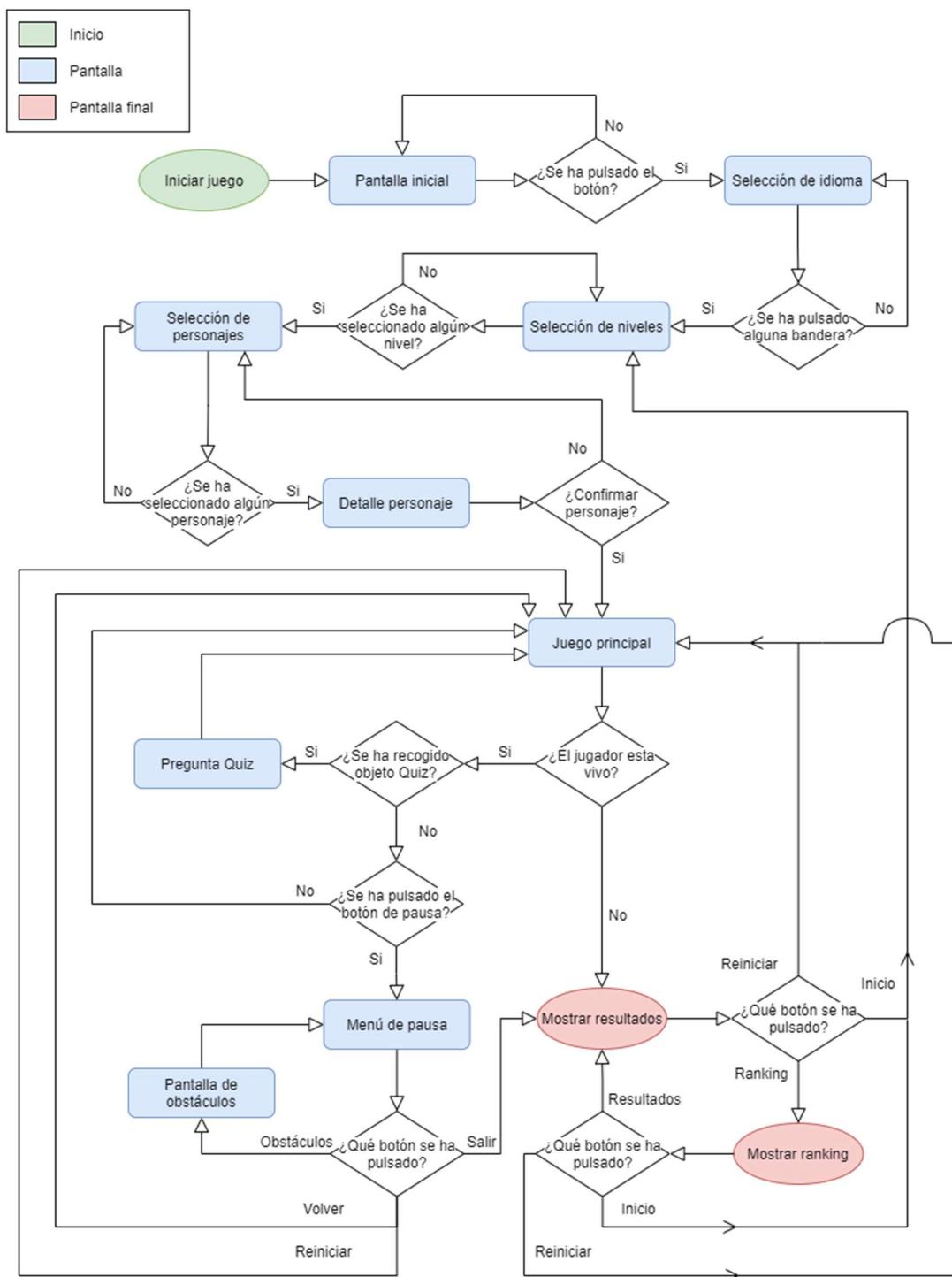


Figura 57: Diagrama de flujo transición entre pantallas

7. IMPLEMENTACIÓN

En este apartado se detalla cómo se ha implementado el juego utilizando tanto el “framework” de VUE como JS.

7.1 Sintaxis de VUE

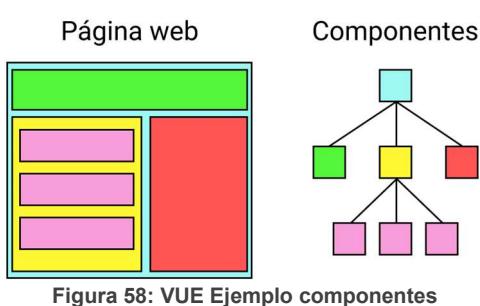
[26] VUE nos permite renderizar datos directamente en el DOM de nuestra página de manera que los datos y el DOM están relacionados, por lo que, si en algún momento decidimos modificar uno de estos datos, el DOM reaccionará y lo actualizará en pantalla. Este proceso se conoce como “Renderización declarativa”.

a) Directivas

Son unos atributos que permiten hacer reaccionar al DOM de una manera especial, estos atributos empiezan con el prefijo “v-“. Por ejemplo, si añadimos la directiva “v-for” en algún elemento de nuestro HTML, se crearán tantos elementos como nos indique el for. Sería el equivalente a ejecutar un bucle “for” en los lenguajes de programación. Otra directiva que destacar sería el “v-if”, la cual nos permite mostrar o no un elemento.

b) Componentes

Son los elementos más importantes de Vue ya que nos permiten separar nuestra página web en distintos bloques o secciones donde cada una ellas encapsularán su propio HTML, CSS y JS de tal forma que podemos reutilizar estas secciones en otras partes de nuestra página sin necesidad de duplicar código.



En la figura 58 podemos ver que la página web (color azul) está formada por los componentes: encabezado (verde), área lateral derecha (rojo) y área lateral izquierda (amarillo). Y este último, a la vez está formado por otros 3 subcomponentes (rosa) que

presentan las mismas características por lo que solo se han declarado una vez, pero han sido reutilizados.

c) Vuex

[35] Es una librería adicional de VUE que nos permite definir datos globales para que los tengamos a disposición en cada uno de nuestros componentes. También podemos declarar métodos para modificar dichos datos. Para ello, tendremos que crearnos una instancia de tipo “Store” que guardara estos datos globales. La “store” está formada principalmente por:

- State (estado): es un objeto que se encarga de guardar los datos globales.
- Mutations (mutaciones): para evitar que los componentes puedan modificar libremente los valores del estado, se deben crear unas funciones específicas para la manipulación de estos datos.

Estas funciones reciben como primer argumento el estado y opcionalmente como segundo parámetro pueden recibir un “payload” (un conjunto de datos encapsulados en un objeto). Una vez se haya modificado el valor del estado, todos los componentes lo verán actualizado al instante.

Para llamar a una mutación hay que usar la sintaxis:
`store.commit('nombreFuncion', 'payload')`

7.2 Estados de juego

Teniendo una clara idea de cómo funciona VUE, ha llegado el momento de empezar con la implementación del juego. Con el fin de tener un código más organizado y limpio, el juego se ha separado en 4 secciones diferentes: “Intro Stage”, “Language Stage”, “Play Stage” y “Final Stage”. De esta manera conseguimos no mezclar todas las secciones del juego en un único archivo, ya que cada una de estas secciones representará una parte del juego.

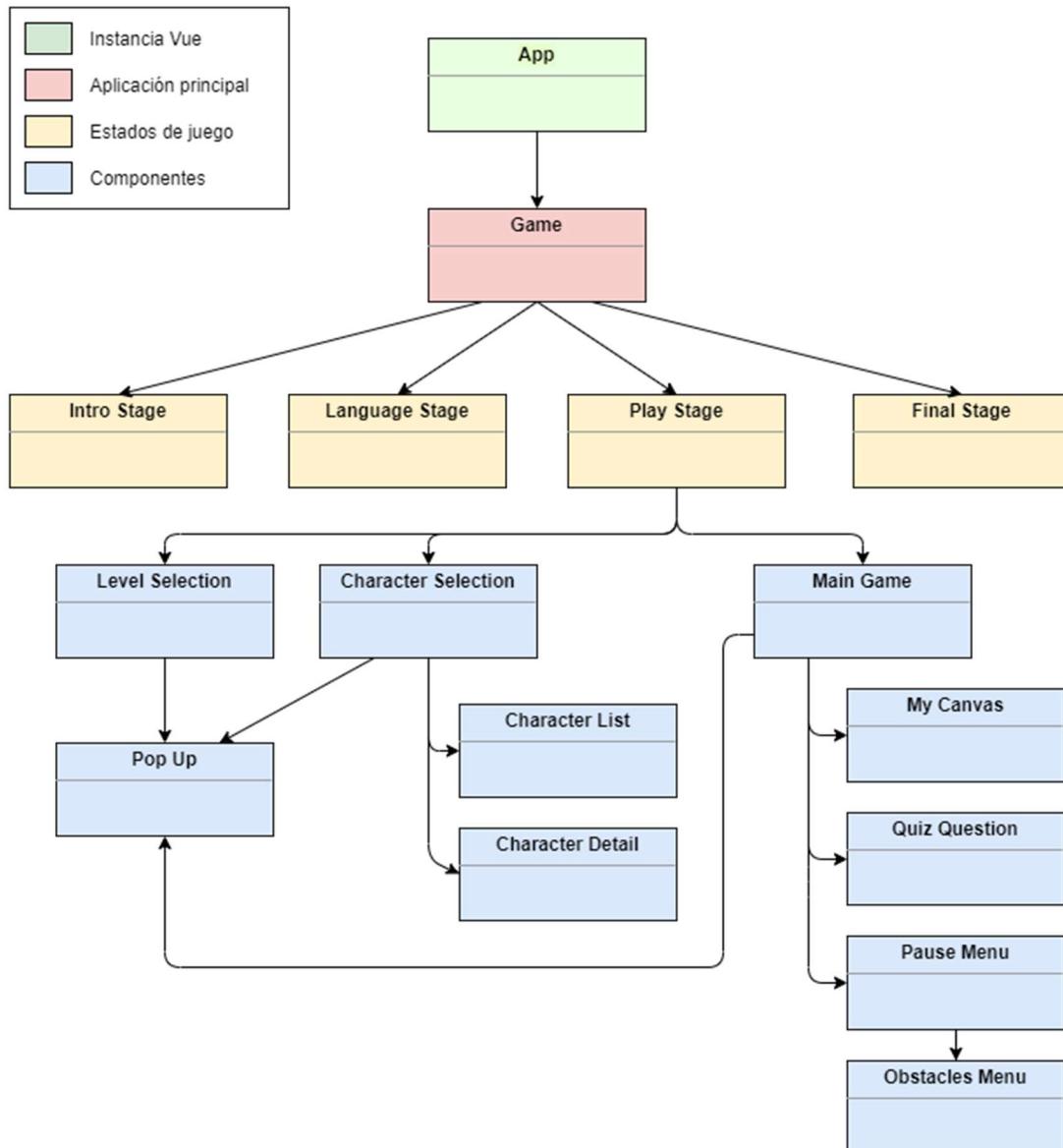


Figura 59: Diagrama estructura del código

Antes de entrar en detalle con cada una de las secciones de juego, veamos la definición de los estados de la “store” (Figura 60):

```

state: {
    //Definición de los distintos estados del juego
    stage: [
        { name: 'Stage', id: STAGE, change: false, change_to: 0 },
        { name: 'IntroStage', id: INTRO_STAGE, change: false, change_to: 0 },
        { name: 'LanguageStage', id: LANGUAGE_STAGE, change: false, change_to: 0 },
        { name: 'PlayStage', id: PLAY_STAGE, change: false, change_to: 0 },
        { name: 'FinalStage', id: FINAL_STAGE, change: false, change_to: 0 }
    ],
    //Estado en el que nos encontramos (inicialmente estaremos en la introducción)
    current_stage: { name: 'IntroStage', id: 1 },
    //Lenguaje que se está utilizando (por defecto aparecerá en inglés)
    language: { name: "English", id: 0 },
    //Mostrar o no los mensajes del narrador
    enablePopUp: true,
    //Ranking local
    local_standings: [
        { level: 0, standings: [
            { pos: 1, points: 0 }, { pos: 2, points: 0 }, { pos: 3, points: 0 },
            { pos: 4, points: 0 }, { pos: 5, points: 0 } ] },
        //Lo mismo para el resto de niveles (cada uno guardará su propio ranking)
    ]
}

```

Figura 60: Definición estados de la store

Y en la figura 61 vemos sus mutaciones más relevantes:

```

mutations: {
    //Cambiar de estado de juego
    changeState( state, payload ) {
        //payload = { index, destination }
        state.stage[payload.index].change = true;
        state.stage[payload.index].change_to = payload.destination;
        state.current_stage.id = payload.destination;
        state.current_stage.name = state.stage[payload.destination].name;
    },
    //Cambiar de lenguaje
    changeLanguage( state, lan_id ) {
        for ( var i = 0; i < LANGUAGES.length; i++ ) {
            if ( LANGUAGES[i].id == lan_id ) {
                state.language.id = lan_id; state.language.name = LANGUAGES[i].name
                break;
            }
        }
    }
}

```

Figura 61: Definición mutaciones de la store

Como muestra la figura de 59, la instancia App llama al componente “Game”, el cual se encargará de decidir qué sección de juego se muestra utilizando el atributo “current_stage” de la “store”:

```
<div v-if="current_stage.id == STAGE"><Stage/></div>
<div v-if="current_stage.id == INTRO_STAGE"><IntroStage/></div>
<div v-if="current_stage.id == LANGUAGE_STAGE"><LanguageStage/></div>
<div v-if="current_stage.id == PLAY_STAGE">
    <PlayStage ref="playStage"
        :prevCharacterID="prevCharacterID"
        :prevLevelID="prevLevelID"
        @gameIsOver="gameIsOver"/></div>
<div v-if="current_stage.id == my_stages.FINAL_STAGE">
    <FinalStage
        :score_obtained="score_obtained"
        @resetGame="resetGame"/></div>
```

Figura 62: HTML Game

Cada vez que se quiera cambiar de sección, se deberá llamar a la mutación “changeState” de la “store”, y acto seguido, “Game” reaccionará mostrando el componente correspondiente, tal y como muestra la figura 62.

7.3 Intro Stage

Esta sección nos servirá para introducir el juego al usuario. Corresponde a la “pantalla inicial” declarada en el apartado de diseño.

En ella podemos apreciar el título del juego junto a una pequeña animación de uno de los personajes jugables que aparecen en el juego (Figura 63).

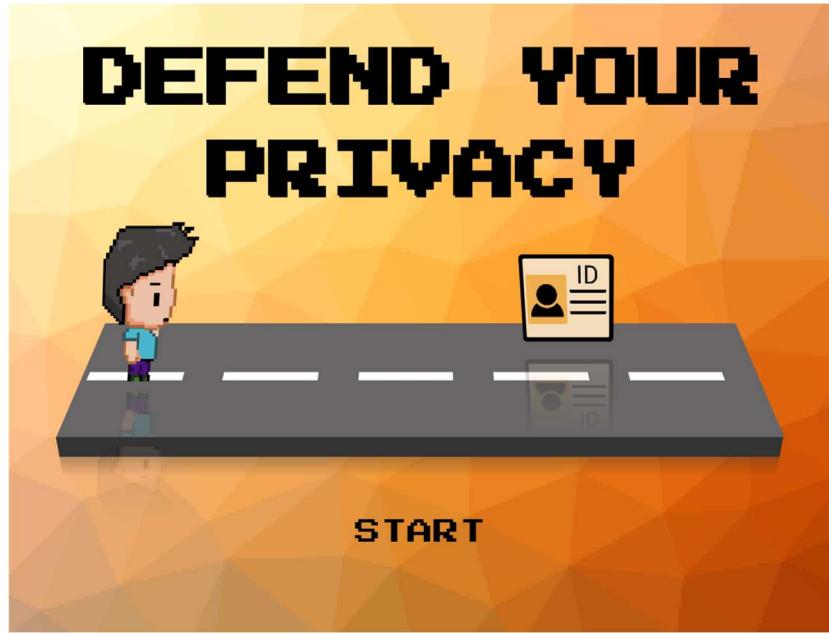


Figura 63: Resultado final Pantalla inicial

Al pulsar el botón de START, se comunicará a la “store” que queremos realizar un cambio de estado, pasando a la sección de “Language Stage”.

7.4 Language Stage

Corresponde con la pantalla de “Selección de idioma”. Cada uno de los idiomas estará definido cómo el objeto que nos muestra la figura 64:

```
[  
  /* DEFINICIÓN DE LOS LENGUAJES */  
  { name: "English", id: 0 , screen_name : "English" ,  
    image: "assets/languages/english.png" },  
  { name: "Spanish", id: 1 , screen_name : "Español" ,  
    image: "assets/languages/spain.png" }  
]
```

Figura 64: Definición Lenguajes

Dónde para cada idioma guardamos su nombre, un identificador, el nombre con el que aparecerá en pantalla y la bandera que lo representa. Con estos datos y utilizando la directiva “v-for” de VUE, podemos representar la lista de idiomas sin preocuparnos del número de elementos por el que este compuesto (Figura 65):

```

<div class="languages">
  <button v-for="(language, index) in languages" :key="index"
    class="btn btn-pointer" @click="changeLanguage( language.id )"
    :style="{ backgroundImage: 'url(`@/${ language.image }`)' }">
    {{language.screen_name}}
  </button>
</div>

```

Figura 65: Listar idiomas en HTML

Para cada idioma declarado dentro de la lista de idiomas, se creará un botón con la imagen de fondo de la bandera y el nombre de este en el centro. En la figura 66 podemos ver el resultado:



Figura 66: Resultado final pantalla selección idioma

Una vez escogido un idioma, todo el texto que aparezca en el juego estará en ese lenguaje y pasaremos a la siguiente sección.

7.5 Play Stage

En esta fase nos encargaremos de seleccionar un nivel, un personaje y jugar al juego. En la figura 67 se muestra como se ha definido esta sección en HTML:

```
<div class="playstage-div">
    <!-- Si el nivel no ha sido seleccionado -->
    <LevelSelection v-if="levelIsNotSelected"
        @levelIsSelected="levelIsSelected"/>

    <!-- Si el personaje no ha sido seleccionado -->
    <CharacterSelection v-if="isThereACharacter()"
        @characterIsSelected="characterIsSelected"/>

    <!-- Una vez tenemos el nivel y el personaje, empezamos la partida -->
    <MainGame v-if="canStartTheGame()"
        :characterID="characterID"
        :levelID="levelID"
        @gameIsOver="gameIsOver"/>
</div>
```

Figura 67: Definición Play Stage en HTML

Inicialmente al no haber ningún nivel seleccionado, se mostrará el componente “LevelSelection”. Este componente corresponde a la pantalla “Selección de niveles” y cómo bien indica su nombre, servirá para mostrar todos los niveles disponibles para que el usuario escoja el que desee. Cada uno de los niveles de juego vendrá definido como el objeto mostrado por la figura 68:

```
{
    /* INFORMACIÓN SOBRE CADA NIVEL */
    id: 0 ,                      // Identificador
    name: [
        "LEVEL 1: CLASSIC",      // Nombre del nivel en inglés
        "NIVEL 1: CLÁSICO"       // Nombre del nivel en español
    ],
    description: [
        "",                      // Descripción en inglés
        ""                       // Descripción en español
    ],
    isActive: true,                // ¿ Está activo ?
    difficulty: "easy" ,          // Nivel de dificultad
    icon: "icon.png" ,            // Dirección imagen ícono
    map: [ 0 ]                     // Identificador del mapa que utiliza
}
```

Figura 68: Definición Nivel

Como podemos ver, las variables “name” y “description” son un array de cadenas de caracteres, es aquí donde entra en juego el identificador del idioma. Por ejemplo, si el idioma que se escogió fue el inglés (cuyo identificador era el 0) leeríamos la posición 0 del array, es decir, el nombre del nivel en inglés.

Para visualizar los niveles, se ha utilizado la misma técnica que en la pantalla de selección de idiomas, con la diferencia de que ahora, se mostrará el nivel activo en el centro y por detrás se mostraran el nivel anterior y el posterior. En la figura 69 se puede ver una implementación en HTML simplificada (el nivel anterior y posterior utilizan la misma sintaxis que el central):

```
<!-- Mostrar los niveles en forma de slider -->
<div class="slideshow-container">
  <div v-for="(level, index) in levels" :key="index">
    <!-- Si el nivel esta activo -->
    <div v-if="level.isActive" class="level-container">
      <!-- Nivel actual en el centro -->
      <div class="card">
        
        <div class="card-content">
          <h1 class="card-header">{{ level.name[language_id] }}</h1>
          <p class="card-text">{{ level.description[language_id] }}</p>
        </div>
        <button class="card-btn btn-pointer"
          @click="levelIsSelected(current_level)"
          {{ buttons.BUTTON_SELECT[language_id] }}>
        </button>
      </div>

      <!-- Siguiente nivel a la derecha -->
      <div v-if="index >= 0 && (index+1) < levels.length" class="card card-right">
        <!-- Utilizando levels[index+1] en vez de level y sin el botón-->
      </div>

      <!-- Nivel anterior a la izquierda -->
      <div v-if="index < levels.length && (index-1) >= 0" class="card card-left">
        <!-- Utilizando levels[index-1] en vez de level y sin el botón -->
      </div>
    </div>
  </div>
</div>
```

Figura 69: Mostrar niveles HTML versión simplificada

Cada nivel se representará con una especie de tarjeta que contendrá el ícono del nivel en la parte superior, el título del nivel en la parte central y su breve descripción debajo de esto. El nivel que este en el centro, además dispondrá de un botón para seleccionarlo. También habrá dos botones en los laterales para cambiar de nivel, o si se prefiere, se puede pulsar a uno de los círculos que aparecen abajo y nos conduciría directamente a su nivel (el primer círculo corresponde al nivel 1, el segundo al 2 y así). En la figura 70 podemos ver el resultado final:



Figura 70: Resultado final pantalla selección de niveles

Una vez se seleccione el nivel, pasaremos a ejecutar el componente “CharacterSelection”, correspondiente a la pantalla “Selección de personajes”.

Este componente, estará formado por dos subcomponentes: “CharacterList” mostrará la lista completa de todos los personajes disponibles, mientras que “CharacterDetail” se mostrará encima de la lista una vez se haya clicado sobre un personaje, mostrando una descripción detallada sobre este. En la figura 71 podemos ver como se ha implementado esta selección en HTML:

```

<div class="selection-div">
  <CharacterList
    @setCharacterID="setCharacterID"/>

  <transition name="fade">
    <CharacterDetail
      v-if="showDetail"
      :characterID="characterID"
      @closeDetail="closeDetail"
      @characterIsSelected="characterIsSelected"/>
  </transition>
</div>

```

Figura 71: Selección de personajes en HTML

Para la lista de personajes, simplemente deberemos recorrer dicha lista de igual forma que se hizo con los idiomas.

Una vez pulsado a un personaje, se activará el componente “CharacterDetail”, el cual recibe el identificador de este personaje. Este nos mostrará una tarjeta como si fuese un PopUp. Utilizando la directiva “v-if” y “v-else”, se mostrará la parte frontal de la tarjeta o la trasera. Al pulsar sobre el botón superior izquierdo, la tarjeta realizará una animación como si voltease y mostrará el otro contenido. En la figura 72 podemos ver las dos caras de esta tarjeta.

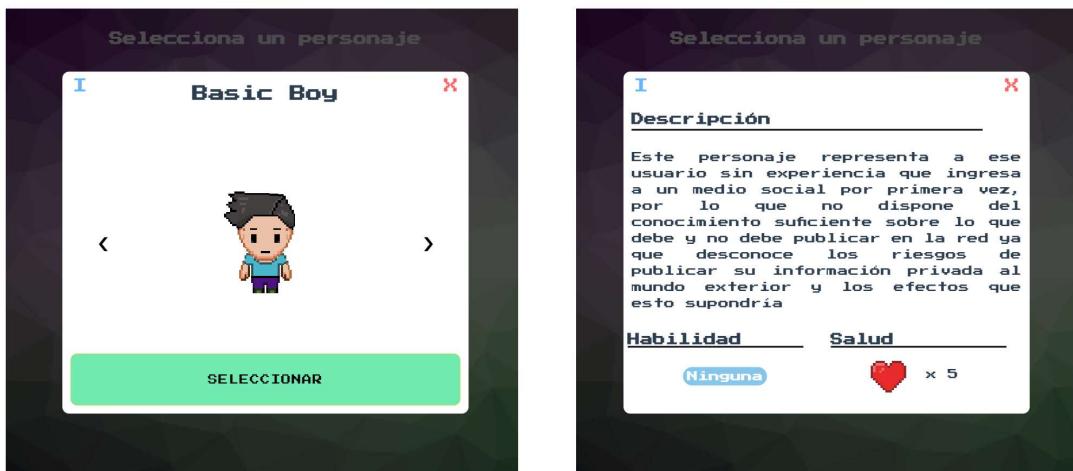


Figura 72: Detalle personaje por delante (izquierda) y por detrás (derecha)

7.6 Juego principal

Para elaborar el juego principal se ha utilizado el elemento “canvas” de HTML, el cual nos permite dibujar gráficos en una página web utilizando JS. A continuación, se detalla cómo se han incorporado cada uno de los elementos del juego:

a) Mapas

Cada uno de los mapas (actuarán como fondo) que aparecen en el juego estará definido como un objeto siguiendo la estructura marcada por la figura 73:

```
{  
    /* INFORMACIÓN SOBRE CADA MAPA DE JUEGO */  
    id: 0,                                // Identificador  
    map :  
        [1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         7, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 8,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         7, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 6, 5, 4, 6, 5, 4, 6, 5, 2,  
         1, 3, 3, 4, 3, 3, 4, 3, 3, 8],  
    tileset: "tileset.png" // Dirección imagen tileset  
}
```

Figura 73: Definición mapa

En la variable “map” se guarda qué porción de la imagen “tileset” corresponde a cada celda del mapa. De esta manera, para representar el mapa en pantalla hemos de recorrer este vector y en función del valor de cada posición, pintamos una sección u otra del “tileset”. En la figura 74, se puede ver de forma más visual. Por ejemplo, la posición “map[0]” contiene un 1, así que para representar esta casilla tenemos que ir a la imagen “tileset” y coger la porción de imagen correspondiente al 1. Si hacemos lo mismo para cada una de las posiciones del vector, obtenemos la imagen “tilemap”.

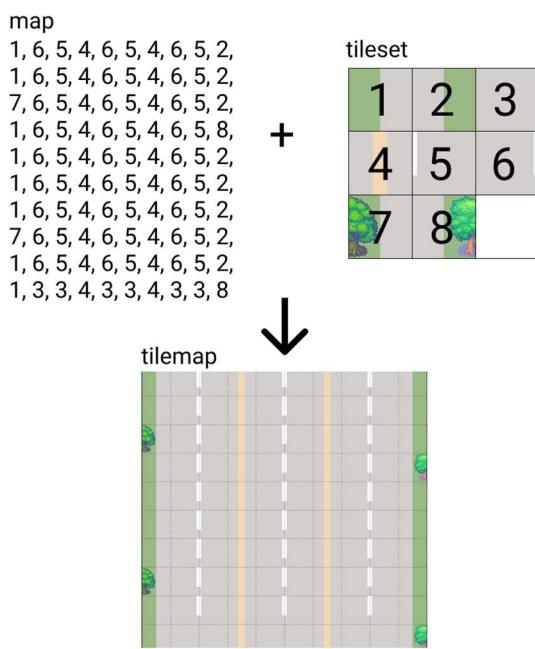


Figura 74: Representación del tilemap

De esta manera ya tenemos representado el fondo para nuestro juego, pero se ve muy estático. Para lograr dar la sensación de movimiento, se ha utilizado la técnica conocida como “Fondo de desplazamiento infinito”, muy utilizada en juegos de arcade interminables. Esta técnica consiste en pintar la misma imagen de fondo dos veces de manera consecutiva para posteriormente desplazarlas en la dirección del eje deseado, cuándo una de ellas desaparece del campo de visión, se reemplaza la segunda imagen con la primera. Este proceso se repite de manera infinita.

A nivel de código, la función encargada de actualizar y pintar el mapa quedaría como muestra la figura 75:

```

drawMap(gamespeed, ctx) {
    if (this.y + gamespeed >= this.height + this.tile_height) {
        this.y = -this.height - this.tile_height + gamespeed;
    } else {
        this.y += gamespeed;
    }

    if (this.y2 + gamespeed >= this.height + this.tile_height) {
        this.y2 = -this.height - this.tile_height + gamespeed;
    } else {
        this.y2 += gamespeed;
    }

    //Fondo 1
    this.drawSection(this.y, ctx);

    //Fondo 2
    this.drawSection(this.y2, ctx);
}

```

Figura 75: Función drawMap

Donde lo primero que se hace es comprobar si la primera imagen está en nuestro campo de visión teniendo en cuenta la velocidad a la que se desplaza, si está dentro, la desplaza, en caso contrario se reinicia su posición para que aparezca arriba de la segunda imagen. Lo mismo para la segunda imagen. Y una vez ya tenemos las posiciones actualizadas, llamamos a la función encargada de pintarlas en pantalla (Figura 76):

```

drawSection(y_section, ctx) {
    for(var x = 0; x < this.map_width; x++) {
        for (var y = 0; y < this.map_height; y++) {

            var t = this.map[ (y*this.map_width) + x ];
            if( t == 0 ) // Casilla vacía
                continue;

            var sx = TILE[t].x * this.tile_width;
            var sy = TILE[t].y * this.tile_height;

            // No dibujar si esta fuera de la pantalla
            var final_y = y_section + y*this.tile_height;
            if (final_y >= - this.tile_height && final_y <= this.height +
                this.tile_height) {
                ctx.drawImage(this.tileset, sx, sy, this.tile_width,
                    this.tile_height, x*this.tile_width, final_y,
                    this.tile_width, this.tile_height)
            }
        }
    }
}

```

Figura 76: Función drawSection

Donde “map_width” y “map_height” equivalen a las dimensiones del mapa que queremos representar (en este caso, 10 casillas x 10 casillas), “tile_width” y “tile_height” a las dimensiones en pixeles de cada casilla, y “TILE” contiene la columna (y) y fila (x) a la que la casilla corresponde con la imagen del “tileset”.

b) Jugador

Del mismo modo que con los mapas, cada uno de los personajes estará definido cómo el objeto que nos muestra la figura 77:

```
{
    /* INFORMACIÓN SOBRE CADA PERSONAJE */
    name: "Character Name",           // Nombre del personaje
    id: 0,                            // Identificador
    icon: "icon.png" ,                // Dirección imagen icono
    data: [
        "",                           // Descripción en inglés
        ""                           // Descripción en español
    ],
    has_ability: false,               // ¿ Tiene alguna habilidad ?
    ability: [
        "None" ,                     // Nombre habilidad en inglés
        "Ninguna"                    // Nombre habilidad en español
    ],
    health: 5 ,                      // Número máximo de vidas
    sprite: "spritesheet.png",       // Dirección imagen spritesheet
    sprite_size_x: 128,              // Ancho sprite (en pixeles)
    sprite_size_y: 165 ,             // Altura
    frames: 4,                       // Número de frames de los que se compone
                                    //     cada animación
    animation: [                      // Animaciones que componen el spritesheet
        // Delante
        [ { x: 0, y: 0 }, { x: 1, y: 0 }, { x: 2, y: 0 }, { x: 1, y: 0 } ],
        // Izquierda
        [ { x: 0, y: 2 }, { x: 1, y: 2 }, { x: 2, y: 2 }, { x: 1, y: 2 } ],
        // Derecha
        [ { x: 0, y: 1 }, { x: 1, y: 1 }, { x: 2, y: 1 }, { x: 1, y: 1 } ],
        // Detrás
        [ { x: 0, y: 3 }, { x: 1, y: 3 }, { x: 2, y: 3 }, { x: 1, y: 3 } ]
    ]
}
```

Figura 77: Definición personaje

A partir de esta estructura de datos y dependiendo del personaje que el usuario haya escogido, crearemos al jugador, el cual se genera a partir de la **clase Player** (Figura 78):

Player				
- sprite: Image	- sprite_x: Float	- sprite_y: Float	- sprite_width: Float	- sprite_height: Float
- animation: List of Vector2 (x: Float, y: Float)		- animation_set: List of Vector2 (x: Float, y: Float)		
- speed: Float	- isMoving: Boolean	- canMove: Boolean		
- lives: Integer	- initial_lives: Integer			
- pos: {x: Float, y: Float}	- target_pos: {x: Float, y: Float}	- offset: {x: Float, y: Float}		
- frames: Integer	- current_frame: Integer	- frames_done: Integer		
- SPRITE_SIZE_X: Float	- SPRITE_SIZE_Y: Float	- FRONT: Integer	- BACK: Integer	- RIGHT: Integer
- LEFT: Integer				
- particles_array: List of Particle	- max_particles: Integer	- particle_color: String		
+ Player (sprite: Image, _animation: List of Vector2 , frames: Integer, SPRITE_SIZE_X: Float, SPRITE_SIZE_Y: Float, health: Integer)				
+ update ()				
+ setAnimation (anim: Integer)				
+ moveOnMouseDown (elapsed_time: Float, gamespeed: Float)				
+ isValidPosition (x: Float)				
+ draw (ctx: Canvas 2D context)				
+ mouseDown (x: Float, y: Float)		+ onMouseUp()		
+ reset ()				
+ handleParticle(ctx: Canvas 2D context)	+ changeParticleColor(color: String)			
+ updateLive()	+ increaseLive()			

Figura 78: Clase Player

Con esta clase controlaremos todo lo relacionado con el jugador:

- **Player:** será el constructor de la clase. Nos permitirá inicializar las variables del jugador utilizando la información del personaje seleccionado.
- **update:** actualizar el “frame” de la animación que se está mostrando.
- **setAnimation:** cambiar de animación.
- **mouseDown:** encargada de cambiar la animación que se está mostrando y de indicar si el jugador se va a mover o no. Por ejemplo, si se pulsa a la izquierda del jugador, se tendrá que cambiar su animación para indicar que se mueve en esa dirección. En la figura 79 podemos ver la definición de esta función.

```

mousedown(x, y) {
    //input -> x,y = coordenados del canvas dónde se ha pulsado
    //output -> true = se ha de mover | false = se queda quieto

    //Donde estan estas coordenadas respecto al player ?
    if ( x < this.pos.x + this.offset.x ) {           //Antes
        this.setAnimation(this.LEFT);
    } else if ( x > this.pos.x + this.offset.x ) {   //Después
        this.setAnimation(this.RIGHT);
    } else {   //Justo encima
        this.setAnimation(this.BACK);
        return false;
    }

    //Actualiza la posición a la que se quiere llegar
    this.isMoving = true;
    this.target_pos = { x: x , y: y };

    return true;
}

```

Figura 79: Función mouseDown de Player

- **onMouseUp**: dejar de moverlo.
- **isValidPosition**: comprobar que la nueva posición a la que hay que moverlo sea una posición valida (con el fin de evitar que desaparezca de nuestro campo de visión).
- **moveOnMouseDown**: si se ha pulsado a la pantalla, actualiza su posición con respecto donde se ha pulsado (Figura 80).

```

moveOnMouseDown( elapsed_time, gamespeed ) {
    if(!this.canMove) return;

    if ( this.target_pos.x - (this.SPRITE_SIZE_X / 4) < this.pos.x +
        this.offset.x ) {
        var target_x = this.pos.x - 0.25 * this.speed * elapsed_time *
            gamespeed;
        this.isMoving = true;
        if( this.isValidPosition(target_x) ) { this.pos.x = target_x; }
    } else if ( this.target_pos.x - (this.SPRITE_SIZE_X / 2) >
        this.pos.x + this.offset.x ) {
        var target_x = this.pos.x + 0.25 * this.speed * elapsed_time *
            gamespeed;
        this.isMoving = true;
        if( this.isValidPosition(target_x) ) { this.pos.x = target_x; }
    } else {
        this.setAnimation(this.BACK);
        this.mouseIsDown = false;
        this.isMoving = false;
    }
}

```

Figura 80: Función moveOnMouseDown de Player

En la figura 81 podemos ver el resultado de aplicar las funciones “`mouseDown`” y “`moveOnMouseDown`” (al pulsar sobre la pantalla, el jugador se mueve hacia esa dirección).

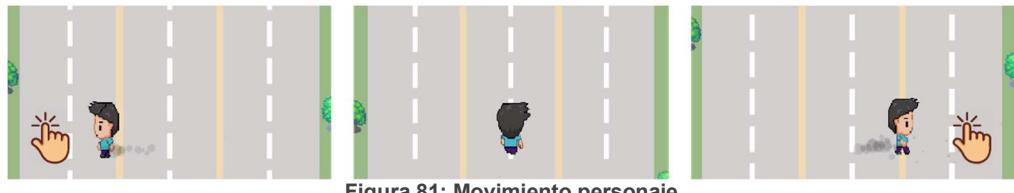


Figura 81: Movimiento personaje

- **`draw`**: representarlo en el canvas.
- **`reset`**: reiniciar todos sus datos.
- **`handleParticle`**: encargada de representar unas partículas flotantes (pequeños círculos) cuando el jugador se mueve. Todas las partículas se generarán donde este el jugador, pero cada una de ellas se moverá en una dirección y tendrán dimensiones distintas (su radio se irá reduciendo con el tiempo hasta que desaparezcan). En la figura 82 se puede ver la animación de estas partículas, siendo la imagen superior izquierda el punto de partida y la imagen inferior derecha el punto final.

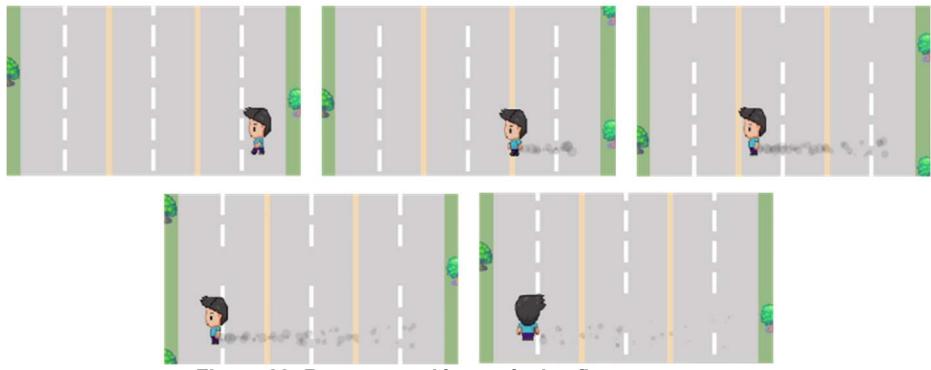


Figura 82: Representación partículas flotantes

- **`changeParticleColor`**: cambiar el color de las partículas que emite.
- **`updateLive`**: decrementar el número de vidas y devuelve “`true`” si sigue con vida o “`false`” en caso contrario.
- **`increaseLive`**: incrementa el número de vidas.

c) Obtáculos

Para crear los obstáculos que van apareciendo se ha utilizado la **clase Obstacle** (Figura 83):

Obstacle					
- sprite: Image					
- x: Float	- y: Float				
- width: Float	- height: Float	- swidth: Float	- sheight: Float		
- speed: Float	- type: Integer	- level: Integer			
- isHit: Boolean	- show: Boolean	- isEnd: Boolean			
- sx: Float	- spx: Float				
+ Obstacle(sprite: Image, x: Float, y: Float, width: Float, height: Float, speed: Float, type: Integer, spx: Float, level: Integer)					
+ update(gamespeed: Float, canvas: Canvas 2D context, score: Integer, elapsed_time: Float)					
+ draw(canvas: Canvas 2D context)					

Figura 83: Clase Obstacle

Funciones:

- **Obstacle**: constructor. Encargado de inicializar las variables del obstáculo.
- **update**: actualizar su posición para cada frame. Si estamos en el nivel de oscilaciones, la posición del obstáculo en el eje de las x variará siguiendo una sinusoidal. En la figura 84 podemos ver en más detalle el funcionamiento de esta función.

```

update(gamespeed, canvas, score, elapsed_time) {
    this.y += this.speed * gamespeed;

    //Si estamos en el nivel de oscilaciones
    if ( this.level == 1 ) {
        //Modifica la posición en el eje de las x de manera sinusoidal
        this.t += 0.5 * gamespeed * elapsed_time;
        //sinusoidal centrada en spx
        this.x = this.spx + ( Math.sin( this.t + this.sx ) * 150 );
    }

    //Si ha llegado al final del canvas
    if ( this.y > canvas.height + this.height) {
        //En caso de no haber sido golpeado por el jugador incrementa
        //la puntuación
        if (!this.isHit) {
            score++;
        }
        //Indicamos que ha llegado al final para que pueda ser eliminado
        this.isEnd = true;
    }

    //Devolvemos la puntuación
    return score;
}

```

Figura 84: Función update de los Obstáculos

- **draw:** representarlo en el canvas.

d) Recursos

La **clase Resource** (encargada de representar a los recursos) presentará una estructura bastante similar a la de los obstáculos, con la principal diferencia de que la función encargada de actualizar su posición no tendrá que preocuparse del nivel en el que estamos puesto que siempre avanzarán en línea recta.

e) Bucle de juego

Lo primero en ejecutarse será la fase de carga, donde se precargarán todas las imágenes que van a salir y se crearán todos los objetos que aparecerán en el juego, es decir, se instanciará al jugador, al mapa, un array de obstáculos y otro de recursos. Una vez completado este paso, empezaremos con el bucle de juego, el cual se repetirá infinitamente hasta que indiquemos que queremos salir (cuando se termina la partida o entramos en modo pausa). Se compone de las siguientes fases:

1. Limpiar todo lo pintado en la escena

2. Actualizar datos del juego
3. Pintar todos los elementos

En la fase 1, simplemente borraremos todo lo pintado en el canvas y volveremos a pintar el mapa encima. Con esta técnica logramos que lo pintado en el frame anterior no interfiera en el actual (en caso contrario veríamos los dos a la vez).

Una vez tenemos el fondo pintado, entramos en la fase de actualizar todos los datos. Empezaremos comprobando si el usuario ha clicado sobre la pantalla, en caso afirmativo, llamaremos a la función del jugador “moveOnMouseDown” (ya hemos visto cómo funciona en el apartado 7.2.b) para actualizar su posición. Acto seguido, actualizaremos el frame de la animación que se está mostrando del jugador para dar la sensación de que se está moviendo.

A continuación, actualizaremos los obstáculos. Para ello, nos recorreremos toda la lista de obstáculos y desplazaremos cada uno de estos obstáculos hacia abajo. Una vez actualizada su posición, comprobaremos si ha llegado al final del mapa, en caso afirmativo, aumentaremos la puntuación en una unidad. Una vez hemos terminado de recorrernos todos los obstáculos, llegará el momento de crear uno nuevo.

Creación de los obstáculos

Al inicio del juego, los obstáculos irán apareciendo poco a poco, pero a medida que progrese el juego, estos irán apareciendo con más frecuencia. Para ello, utilizaremos un intervalo de tiempo que se encargará de controlar cada cuanto hemos de crear un obstáculo. En la figura 85 podemos visualizar la función encargada de gestionar la generación de obstáculos:

```

if ( this.total_frames % this.obstacles_interval == 0 ) {
    // Creamos un obstáculo
    var obs = createObstacle( ... );
    if ( obs != null ) {
        // Lo añadimos al array con el resto
        this.obstacles.push( obs );

        // 50% probabilidades de crear un segundo obstáculo
        var rnd = Math.floor( Math.random() * 2);

        // Crear un segundo obstáculo si:
        // - Estamos en el primer nivel y se ha conseguido mas de 10 puntos en el juego
        // - No crearlo en el nivel con ID = 1 (nivel de oscilaciones)
        if ( (this.levelID == 2 || this.score >= 10) && rnd == 0 &&
            this.levelID != 1 ){
            // Guardamos donde estaba el que acabamos de crear para evitar que se
            // solapen
            var x = obs.x;
            var y = obs.y;
            // Segundo obstaculo
            var obs2 = createObstacle( ... );
            if ( obs2 != null ) {
                this.obstacles.push( obs2 );
                // Repetir el mismo proceso para crear un tercer obstáculo solo si
                // estamos en nivel 3 (nivel de más obstáculos)

                // . .
            }
        }
    }

    // Decrementar el intervalo que regula la aparición de los obstáculos para que
    // a medida que pasa el tiempo aparezcan más seguidos
    if ( this.obstacles_interval > 50 ) {
        this.obstacles_interval -= 50;
    }
}
}

```

Figura 85: Función para generar nuevos obstáculos

Hasta ahora tenemos definido cuando se generan los obstáculos, pero nos queda por ver cómo se crean. El objetivo es que el obstáculo aparezca de manera aleatoria en uno de los puntos de aparición, para ello, necesitamos saber esos puntos, pero, además, tal y como hemos visto en la función de arriba, es posible que se generan 1, 2 o 3 obstáculos (dependiendo del nivel en el que

estemos) así que hemos de saber si hay otro obstáculo justo en esa fila para evitar su solapamiento. El tipo de obstáculo se escogerá de manera aleatoria.

Creación de los recursos

El proceso para generar los recursos será muy similar al de los obstáculos

Tal y como se especificó en el apartado de “Sistema central del juego”, habrá 2 tipos distintos de recursos: quiz y vida. Teniendo esto en cuenta, a la hora de crear un nuevo recurso, este proceso se llevará de manera aleatoria con un 2/3 de posibilidades a favor de que aparezca el quiz y un 1/3 para la vida. De esta manera priorizamos a que salga más veces una pregunta quiz.

En la figura 86 podemos ver el resultado visual de mostrar el mapa, el jugador, los obstáculos y los recursos:

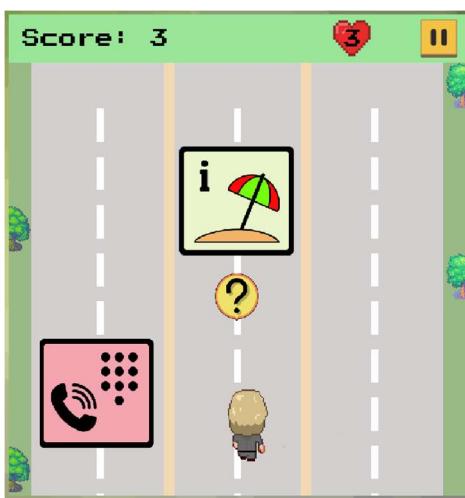


Figura 86: Visualización del juego

Colisiones

Una vez hemos terminado de actualizar la posición de los obstáculos y los recursos, ha llegado el momento de comprobar si se ha producido una colisión con respecto al jugador.

Para testear las colisiones se ha utilizado la caja de colisión que cubre a cada uno de estos objetos. Es cierto que la colisión no será tan perfecta como si testeáramos con cada uno de los pixeles que forman los objetos, pero al hacerlo

con las cajas de colisiones obtenemos un mayor rendimiento debido a que solo hemos de testear con 4 puntos.

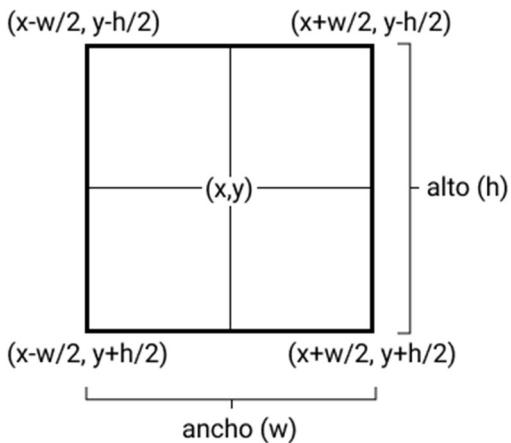


Figura 87: Definición caja de colisión

La posición de los objetos en mundo está declarada respecto al centro del objeto, a la hora de calcular la caja de colisión lo deberemos tener en presente. En la figura 87 podemos ver como se define la caja.

Teniendo esto en cuenta, el algoritmo para detectar si hay o no colisión entre dos rectángulos sería el que nos muestra la figura 88:

```
function collisionCenter(rect1_x, rect1_y, rect1_width, rect1_height,
    rect2_x, rect2_y, rect2_width, rect2_height) {

    //obtener el vértice superior izquierdo del primer rectángulo
    var new_x1 = Math.floor( rect1_x - rect1_width/2 );
    var new_y1 = Math.floor( rect1_y - rect1_height/2 );
    //y el del segundo rectángulo
    var new_x2 = Math.floor( rect2_x - rect2_width/2 );
    var new_y2 = Math.floor( rect2_y - rect2_height/2 );

    //mirar si se solapan
    if (new_x1 < new_x2 + rect2_width &&
        new_x1 + rect1_width > new_x2 &&
        new_y1 < new_y2 + rect2_height &&
        rect1_height + new_y1 > new_y2) {
        return true;
    }
    return false;
}
```

Figura 88: Función para detectar una colisión

Este algoritmo se aplicará para cada uno de los obstáculos y recursos que encontraremos en escena. En caso de que el jugador colisione con un obstáculo, se le decrementará en una unidad su vida y el obstáculo desaparecerá.

Para evitar que el obstáculo desaparezca instantáneamente, se mostrará la animación que aparece en la figura 89 simulando una explosión:

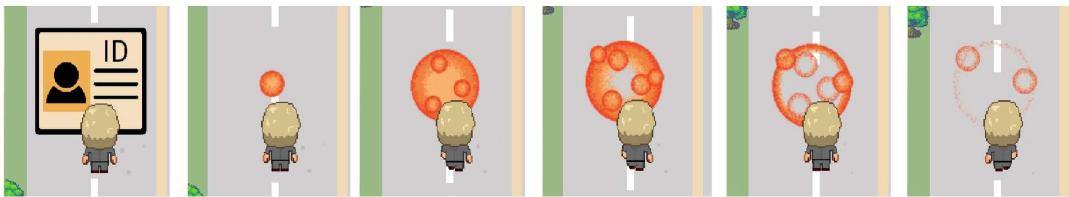


Figura 89: Colisión con obstáculo

Por otro lado, si ha habido una colisión con un recurso, se tendrá que comprobar si este era del tipo vida (augmentará la vida del jugador en una unidad) o del tipo quiz. En caso de ser un quiz, el juego quedará pausado y se pasará a mostrar la pantalla con una pregunta de quiz.

Pregunta quiz

La pregunta quiz será escogida de entre un conjunto posible de preguntas. Cada una de estas preguntas vendrá definida como el objeto descrito en la figura 90:

```
{  
    id: 0,           // Identificador  
    question: [  
        "Question", // Pregunta en inglés  
        "Pregunta"  // Pregunta en español  
    ],  
    answers: [       // Respuestas  
        { name: [ "Answer True", "Respuesta Verdadera" ], status: true },  
        { name: [ "Answer False", "Respuesta Falsa" ], status: false },  
        { name: [ "Answer False", "Respuesta Falsa" ], status: false },  
        { name: [ "Answer False", "Respuesta Falsa" ], status: false }  
    ]  
}
```

Figura 90: Definición pregunta quiz

Para cada una, guardamos su identificador, un array con la pregunta en ambos idiomas y otro con las respuestas. En el array de respuestas, aparte de guardar el texto en cada idioma tendremos la variable “status” que nos indica si esa respuesta es verdadera o falsa.

Una vez escogida, se reordenarán las respuestas de manera aleatoria. Así, en caso de que el usuario falle la pregunta y le vuelva a aparecer más adelante,

tendrá que prestar atención a las respuestas puesto que estas habrán cambiado de lugar. En la figura 91 podemos ver el ejemplo de una pregunta (a la izquierda) y a la derecha vemos otra, pero esta vez una de las respuestas aparece de color gris, esto se debe a que el personaje con el que se está jugando posee la habilidad de descartar una respuesta errónea, ayudando así al usuario.



Figura 91: Pregunta quiz estándar (izquierda) y pregunta con respuesta descartada (derecha)

Para indicar al usuario si ha acertado o no, se utilizarán sonidos indicativos.

Menú de pausa

En cualquier momento podremos acceder al menú de pausa al clicar al botón que aparece en la esquina derecha superior, en ese instante, el juego quedará pausado, por lo que ningún elemento actualizará su posición. Como se definió en el apartado de diseño, dependiendo del botón que se pulse en este menú, se realizará una acción u otra. A continuación, en la figura 92, podemos ver el resultado visual de esta pantalla junto al menú de obstáculos que se mostraría al pulsar el botón de obstáculos.

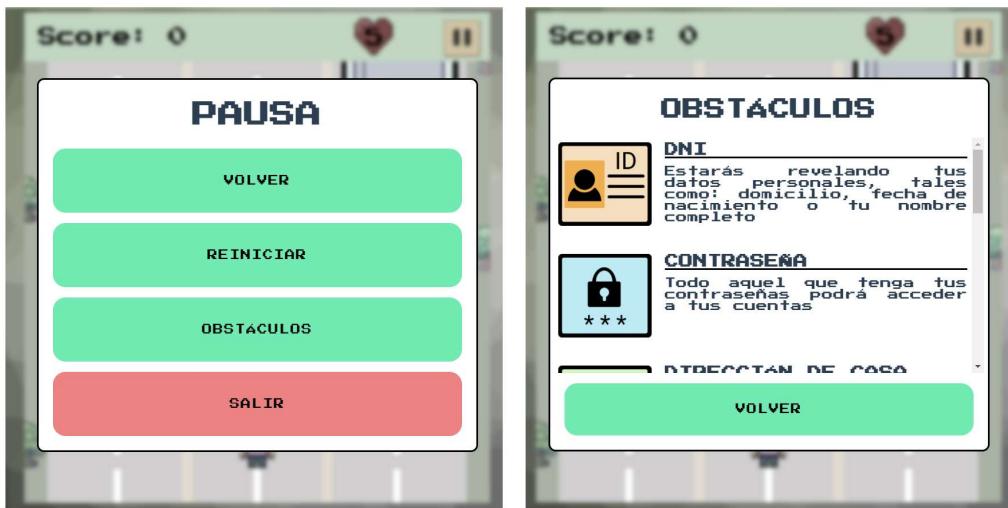


Figura 92: Menú de pausa (izquierda) y lista de obstáculos (derecha)

Fin de partida

En el instante en que el jugador se quede sin vidas, el juego llegará a su fin y será el momento de cambiar a la sección “Final Stage” para que puedan mostrarse por pantalla los resultados obtenidos.

7.6 Final Stage

Última sección del juego. Corresponde tanto a la pantalla de resultados como a la del ranking dado que ambas comparten la misma estructura.

Este componente recibirá por parte del juego principal el JSON que se muestra en la figura 93, que contendrá información sobre la partida:

```
//this.$props['score_obtained'] = { score: integer, bonus: integer, character: integer, level: integer, time: integer }
var data = JSON.parse( this.$props['score_obtained'] );
//guardamos los datos en local
this.game_score = data.score; //puntos juego
this.bonus_score = data.bonus; //puntos parte quiz
this.time = data.time; //tiempo total de juego
this.character_id = data.character; //id del personaje
this.level_id = data.level; //id del nivel
this.total_score = this.game_score + this.bonus_score;
```

Figura 93: JSON puntuación

Una vez lo hemos leído, procedemos a actualizar el ranking local (Figura 94):

```
// Extraemos datos de los resultados anteriores (previamente guardados en la store)
this.standings = this.local_standings[ this.level_id ].standings;

// Actualizamos el ranking mirando si la nueva puntuación ha superado a alguna
var max_index = -1;
for( var i = this.standings.length-1; i >= 0; i-- ) {
    if( this.total_score > this.standings[i].points ) {
        max_index = i;
    }
}

// Si ha superado alguna puntuación de las que había previamente
if( max_index != -1 ) {
    // Añade la puntuación al ranking
    this.standings.splice( max_index, 0, { pos: max_index,
                                              points: this.total_score } );

    for ( var i = 0; i < this.standings.length; i++ ) {
        this.standings[i].pos = i + 1;
    }

    // Nos quedamos con los 5 primeros resultados
    this.standings = this.standings.slice( 0, 5 );
}

this.standings_player_pos = max_index;
this.local_standings[ this.level_id ].standings = this.standings;

// Guardamos el ranking en el local storage del navegador
localStorage.setItem( 'local-standings', JSON.stringify( this.local_standings ) );
```

Figura 94: Función para actualizar el ranking

Una vez hemos extraído la puntuación obtenida en el juego y actualizado el ranking, procedemos a mostrar los resultados por pantalla. Para ello, haciendo uso de la directiva “v-if” y “v-else” de VUE, mostraremos la puntuación obtenida en la partida o bien el ranking local con la puntuación de partidas anteriores. En la figura 95 podemos ver una versión simplificada del HTML encargado de mostrar estas dos pantallas.

```

<div class="board-menu">
  <div class="board-header">
    <span class="header-title">
      <h1 v-if="showResults">{{ titles.TITLE_GAMEOVER[language_id] }}</h1>
      <h1 v-else>{{ titles.TITLE_GAMEOVER_STANDINGS[language_id] }}</h1>
    </span>
    <button class="header-btn btn-pointer" :style="{ backgroundImage:
      'url(' + require(`@/assets/podium.png`) + ')'}"
      @click="renderStandings">
    </button>
  </div>

  <div class="board-content" v-if="showResults">
    <!--Mostrar tiempo total de juego-->
    <!--Mostrar puntuación-->
    <!--Mostrar puntuación bonus-->
    <!--Mostrar puntuación total-->
  </div>

  <div class="board-standings" v-else>
    <div v-for="(stand, index) in standings" :key="index"
      v-bind:class="[standings_player_pos == index ? 'standing player' : 'standing']">
      <!--Mostrar top 5 mejores resultados en el ranking-->
    </div>
  </div>

  <!-- Botones parte inferior -->
  <div class="board-buttons">
    <button class="btn btn-pointer" @click="reset">
      {{ buttons.BUTTON_GAMEOVER_RESET[language_id] }}
    </button>
    <button class="btn btn-pointer" @click="goHome">
      {{ buttons.BUTTON_GAMEOVER_HOME[language_id] }}
    </button>
  </div>
</div>

```

Figura 95: FinalStage simplificado HTML

En la figura 96 podemos ver el resultado visual final de esta sección de juego:



Figura 96: Pantalla de resultados (izquierda) y ranking (derecha)

8. Fase de validación

Una vez finalizada la implementación del juego, ha llegado el momento de validarla para determinar si el resultado final cumple con los requisitos y propósitos iniciales. Para ello, se ha procedido a realizar una evaluación cuantitativa con 12 adolescentes de entre 15 y 19 años, quienes tras probar el juego durante unos minutos deberán responder a un cuestionario.

Esta evaluación consiste en los siguientes pasos:

1. **Introducción a la evaluación:** se le hará una pequeña introducción al usuario explicándole en qué consiste el proyecto y qué es lo que tendrá que realizar durante esta fase de evaluación. Se le dejará claro que todos sus datos serán tratados con anonimato.
2. **Perfil del encuestado:** a continuación, se le pedirá que responda a una breve encuesta para conocer su perfil de usuario. Estas preguntas también nos servirán para conocer qué medios son los más populares, así como la cantidad de horas dedicadas a los mismos y su nivel de preocupación acerca de la privacidad.
3. **Jugar al juego:** durante los próximos 15 minutos, el usuario deberá probar el juego en profundidad con el fin de que pueda ejecutar todos los escenarios posibles.
4. **Encuesta:** transcurrido este tiempo, el usuario deberá responder a una serie de preguntas relacionadas sobre el juego que nos permitirán conocer sus opiniones sobre él, pero también para validar si los requerimientos no funcionales han sido aplicados con éxito.

La encuesta (tanto la parte destinada a conocer el perfil de usuario como la valoración al finalizar el juego) se encuentra en el Anexo.

8.1 Resultados encuesta

Todos los encuestados se encuentran en el rango de edad de entre 15 y 19 años, donde el 66'7% de ellos son hombres y el 33'3% restante son mujeres.

Para conocer su perfil de usuario, se les preguntó qué tan frecuente utilizaban los medios sociales, y en todos los casos respondieron que los utilizaban casi siempre o de manera muy frecuente (el 66'7% estimó que dedicaba más de 2 horas al día, seguido por un 25% que lo usaba entre 1 y 2 horas, tal y como muestra el gráfico de la figura 97).

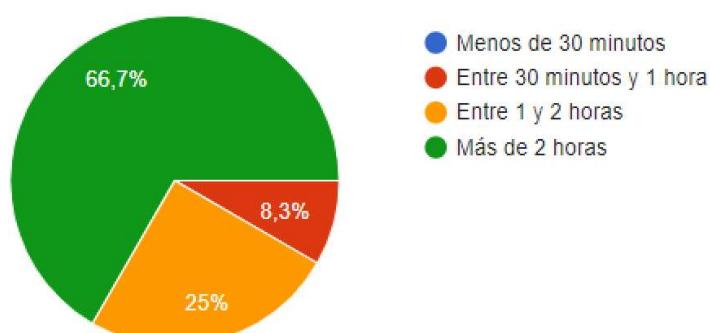


Figura 97: Cuánto tiempo les dedican a los medios sociales de media al día

Los medios sociales más utilizados fueron los medios multimedia (en un 75% de los casos) seguido por las redes sociales. Finalmente, el 50% dijo que le era indiferente como se tratase su privacidad en este tipo de medios, mientras que al 33'4% sí que le preocupaba más y al 16'6% restante casi nada. Con estos datos podemos ver la necesidad de educar a los adolescentes sobre los peligros de exponer su privacidad en estos medios ya que es muy probable que no estén al corriente de los riesgos que conlleva exponer su privacidad al mundo exterior.

Una vez conocido como es su perfil, pasamos al análisis del juego, donde, prácticamente todos ellos fueron capaces de avanzar en el juego sin ningún problema, por lo que demuestra que el diseño de la interfaz es intuitivo.

Un 83'3% de ellos aseguró haber podido visualizar correctamente todos los elementos gráficos (imágenes, botones...) sin que ninguno presentase ninguna distorsión. Mientras que el 16'7% notó que algún elemento se distorsionó un poco, pero esto se debe a las pequeñas variaciones que hay entre las

dimensiones de los dispositivos en los que se jugó ya que al hacerlo adaptivo para tanto versión de escritorio como móvil, es posible que, en algún dispositivo, algún escalado de los elementos no haya dado el resultado que se deseaba. De todas formas, a todos ellos les pareció que existe una coherencia visual con todos los elementos que han aparecido en el juego, por lo que podemos garantizar que la coherencia estética ha sido aplicada con éxito.

A nivel de rendimiento, podemos garantizar que el tiempo de respuesta al realizar una acción ha sido rápido, aunque en algunos casos puntuales, algunas acciones han tardado un poco más de lo que desearían.

Cómo muestra la figura 98, el 75% de ellos logró identificar lo que significan los obstáculos que se han de evitar durante el juego, mientras que el 25% restante presentó alguna dificultad a la hora de averiguar el significado de algún obstáculo. Sin embargo, gracias al menú que muestra la definición de cada uno de ellos, lograron entenderlo.

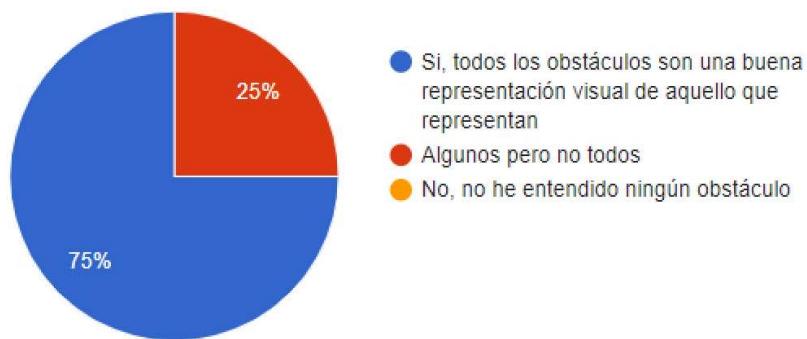


Figura 98: Identificación de los obstáculos sin la necesidad de consultar su definición

Por lo que respecta a las preguntas quiz, les pareció una buena idea y encajaron correctamente en el juego, aunque se podría subir el nivel de dificultad de algunas ellas ya que cómo muestra el gráfico de la figura 99 algunos usuarios las encontraron más sencillas de lo que se esperaban (siendo 1 muy fácil y 5 muy difícil).

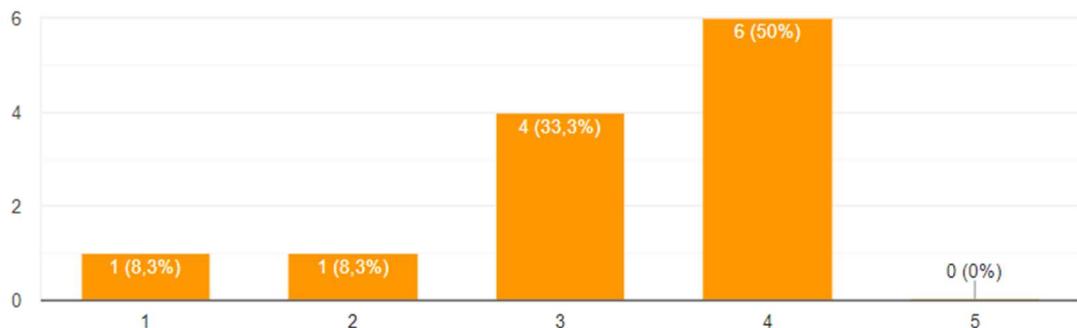


Figura 99: Nivel de dificultad de las preguntas quiz

Para comprobar la fiabilidad del juego, se les preguntó si en algún momento dejó de funcionar, y en el 83'3% de los casos no hubo ningún fallo. Sin embargo, el 16'7% presentó algún fallo leve en algún instante, pero se solventó rápidamente por lo que podemos garantizar que el sistema es a prueba de fallos.

Finalmente, como se muestra en el gráfico de la figura 100, el 50% de ellos aseguro haber aprendido varios conceptos nuevos (barras con los valores 4 y 5), mientras que un 25% aprendió algo y el 25% restante un poco menos. Por lo que podemos concluir que en todos los casos se logró aprender algo por más mínimo que fuese.

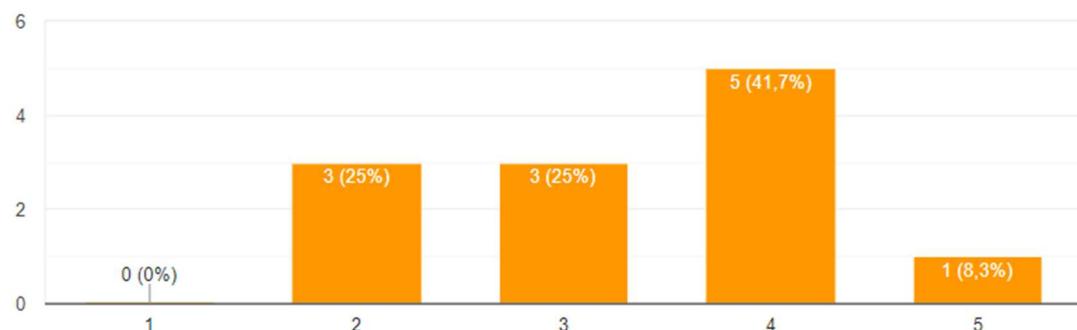


Figura 100: Aprendizaje en el juego

9. CONCLUSIONES Y TRABAJO FUTURO

9.1 Conclusiones

Tras analizar los resultados extraídos de la fase de validación, me ha sorprendido bastante la cantidad de horas que los adolescentes dedican de media a los medios sociales cada día, pero sobre todo la poca importancia que le dan a sus datos personales, lo que me da a entender que no están demasiado al corriente de los riesgos que supone la publicación de estos. No obstante, gracias a un juego de este estilo, no solo pudieron entretenerte durante un rato, sino que también lograron aprender conceptos nuevos sobre la privacidad que pueden resultarles de gran ayuda en su día a día. Por ende, gracias a este proyecto he podido comprender mejor la importancia y el potencial de estos juegos educativos, herramientas que esconden un objetivo que va más allá de la simple diversión y entretenimiento.

Uno de los grandes retos con los que me encontré durante la creación del juego fue la necesidad de adaptarlo para que fuera funcional tanto para la versión de escritorio como la versión móvil ya que hacer una aplicación web que se adapte a cualquier dispositivo sin deformar su contenido no es una tarea sencilla. También, en un inicio, el juego empezaba sin una fase de carga lo que generaba que las imágenes mostradas y los sonidos aparecieran de manera repentina en mitad del juego, por lo que se rompía un poco el flujo del juego. Para solventar este problema, me vi con la necesidad de precargar todos los recursos audiovisuales antes de hacer uso de ellos. Gracias a esta mejora, el tiempo de respuesta al realizar una acción por parte del usuario también se vio reducido. Todo ello logró incrementar el rendimiento del juego.

Por último, por lo que hace al proceso de aprendizaje, este trabajo me ha servido para mejorar mis conocimientos sobre HTML, CSS y JS, así como para aprender a utilizar el framework de Vue. Por otro lado, también he podido poner en práctica las distintas fases que hay que seguir para la creación de un proyecto orientado a la web siguiendo el proceso de ingeniería de software.

9.2 Trabajo futuro

A continuación, se muestran una serie de mejoras y variantes que se podrían introducir en el minijuego:

- **Cambio de “assets”:** el juego ha estado pensado para tratar el tema de la privacidad en los medios sociales, pero si cambiamos los assets del juego, es decir, los obstáculos, los recursos, los personajes o el mapa, podríamos adaptar este juego a otras temáticas a la vez que mantenemos las mecánicas de juego. Simplemente nos bastaría con hacer un rediseño de los assets.
- **Más idiomas:** por ahora, el juego se puede jugar en dos idiomas: inglés y español, pero podríamos añadir algunos más sin mucho esfuerzo.
- **Ranking online:** si disponemos de un servidor, podríamos crearnos una base de datos que permita el registro de los jugadores. De esta manera, pasaríamos de tener un juego local a uno online donde los jugadores podrían compartir sus resultados entre ellos.

REFERENCIAS

- [1] Simon Kempt, 2021. “Digital 2021: The latest insights into the ‘state of digital’”. We are social. [Online]. Disponible:
<https://wearesocial.com/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital> [Accedido: 2 febrero 2021]
- [2] Kaplan Andreas M., Haenlein Michael, 2010, “Users of the world, unite! The challenges and opportunities of social media”. Business Horizons, Vol. 53, p. 59-68. Disponible:
<https://www.webatlas.it/pdf/kaplan-haenlein-users-of-the-world-unite.pdf>
[Accedido: 10 enero 2021]
- [3] “Blog”. Real Academia Española. [Online]. Disponible:
<https://dle.rae.es/blog> [Accedido: 10 enero 2021]
- [4] WordPress. [Online]. Disponible:
<https://wordpress.com/> [Accedido: 10 enero 2021]
- [5] Curtis Foreman, 2017. “10 Types of social media and how each can benefit your business”. Hootsuite. [Online]. Disponible:
<https://blog.hootsuite.com/types-of-social-media/> [Accedido: 10 enero 2021]
- [6] O'Keeffe GS, Clarke-Pearson K; Council on Communications and Media, 2011. “The impact of social media on children, adolescents, and families”. Pediatrics. Disponible:
<https://pediatrics.aappublications.org/content/pediatrics/127/4/800.full.pdf>
[Accedido: 12 marzo 2021]
- [7] Carol Espona, 2020. Rtve. “BackUp: Una serie de investigación sobre delitos digitales”. [Video]. Disponible:
<https://lab.rtve.es/lab/backup/> [Accedido: 13 marzo 2021]
- [8] Oesía, 2021. “En red sin riesgos, RSC de Ciberseguridad”. [Online]. Disponible: <https://enredsinriesgos.oesia.com/> [Accedido: 29 junio 2021]
- [9] John Palfrey, Urs Gasser, Danah Boyd, 2010. “Response to FCC Notice of Inquiry 09-94: Empowering Parents and Protecting Children in an Evolving Media Landscape”. Berkman Center for Internet and Society, Harvard University. Disponible:

- https://cyber.harvard.edu/sites/cyber.law.harvard.edu/files/Palfrey_Gasser_boyd_response_to_FCC NOI 09-94 Feb2010.pdf [Accedido: 12 marzo 2021]
- [10] “Digital Footprint”, 2021. Wikipedia. [Online]. Disponible: https://en.wikipedia.org/wiki/Digital_footprint [Accedido: 12 marzo 2021]
- [11] Nadia Kovacs, 2021. “Tips for protecting your social media privacy”. Norton. [Online]. Disponible: <https://us.norton.com/internetsecurity-privacy-protecting-privacy-social-media.html> [Accedido: 25 febrero 2021]
- [12] Total Eclipse, 2012. “A Clockwork Brain”. [Online]. Disponible: <http://www.aclockworkbrain.com/> [Accedido: 11 enero 2021]
- [13] Gusmanson, 2018. “Bad News”. [Online]. Disponible: <https://www.getbadnews.com/> [Accedido: 12 enero 2021]
- [14] The Carnegie Cyber Academy, 2021. “Training tomorrow’s heroes”. [Online]. Disponible: <https://www.carnegiecyberacademy.com/> [Accedido: 29 junio 2021]
- [15] ABCya, 2021. “Cyber Five: Internet Safety”. [Online]. Disponible: https://www.abcyah.com/games/cyber_five_internet_safety [Accedido: 29 junio 2021]
- [16] “The Companion – Courage”. UPF. [Online]. Disponible: <https://www.upf.edu/web/courage/the-companion> [Accedido: 20 noviembre 2020]
- [17] PixelFed. [Online]. Disponible: <https://pixelfed.org/> [Accedido: 20 noviembre 2020]
- [18] Open-Source Initiative. [Online]. Disponible: <https://opensource.org/> [Accedido: 21 diciembre 2020]
- [19] “HTML Introduction”. w3schools. [Online]. Disponible: https://www.w3schools.com/html/html_intro.asp [Accedido: 2 enero 2021]
- [20] Javier Eguiluz, 20XX. “Introducción a CSS”. Capítulo 1.2. [Online]. Disponible: <https://uniwebsidad.com/libros/css/capitulo-1/breve-historia-de-css> [Accedido: 2 enero 2021]
- [21] “JavaScript”. Tutorialspoint. [Online]. Disponible:

- <https://www.tutorialspoint.com/javascript/index.htm> [Accedido: 2 enero 2021]
- [22] “JQuery”, 2021. Wikipedia. [Online]. Disponible: <https://en.wikipedia.org/wiki/JQuery> [Accedido: 3 enero 2021]
- [23] “Usage statistics of JavaScript libraries for websites”. W3Techs. [Online]. Disponible: https://w3techs.com/technologies/overview/javascript_library [Accedido: 2 enero 2021]
- [24] Angular documentation. [Online]. Disponible: <https://angular.io/docs> [Accedido: 8 enero 2021]
- [25] React documentation. [Online]. Disponible: <https://reactjs.org/docs/getting-started.html> [Accedido: 8 enero 2021]
- [26] Vue documentation. [Online]. Disponible: <https://vuejs.org/v2/guide/> [Accedido: 8 enero 2021]
- [27] Andrés Abraham, 2019. “Angular vs React vs Vue: ¿Cuál es la mejor opción?”. [Online]. Disponible: <https://medium.com/somoswigou/angular-vs-react-vs-vue-cu%C3%A1l-es-la-mejor-opci%C3%B3n-941a207951c7> [Accedido: 8 enero 2021]
- [28] Asset. Geekno. [Online]. Disponible: <https://www.geekno.com/glosario/asset> [Accedido: 12 enero 2021]
- [29] PiskelApp. [Online]. Disponible: <https://www.piskelapp.com/> [Accedido: 12 enero 2021]
- [30] Map Editor. [Online]. Disponible: <https://www.mapeditor.org/> [Accedido: 13 enero 2021]
- [31] Figma. [Online]. Disponible: <https://www.figma.com/> [Accedido: 10 enero 2021]
- [32] Brandon Jones, 2017. “5 Personal details you should never post on social media”. PSafe Blog. [Online]. Disponible: <https://www.psafelogin.com/en/blog/5-personal-details-you-should-never-post-on-social-media/> [Accedido: 26 febrero 2021]
- [33] Itay Keren, 2015. Independent Games Summit, GDC. “The Theory and Practice of Cameras in Side-Scrollers”. [Video]. Disponible: <https://www.youtube.com/watch?v=pdvCO97jOQk> [Accedido: 8 marzo 2021]

- [34] Brown, Dan M. (2010). “Communicating Design: Developing Web Site Documentation for Design and Planning”.
- [35] Vuex library documentation. Vue. [Online]. Disponible: <https://vuex.vuejs.org/guide/> [Accedido: 9 enero 2021]

ANEXO

Encuesta

Defend your privacy

Defend your privacy es un minijuego que nace con el objetivo de ayudar a los adolescentes a entender lo que no deben publicar en los medios sociales relacionados con temas sobre la privacidad.

Enlace al juego: <https://defendyourprivacy.github.io/>

Agradezco su colaboración en este estudio, y me gustaría recalcar que todos los datos extraídos de este cuestionario van a ser tratados de manera anónima y no serán utilizados fuera de este estudio.

* Obligatoria

¿En qué rango de edad te sitúas? *

- Menos de 13 años
- Entre 13 y 14 años
- Entre 15 y 16 años
- Entre 17 y 18 años
- 19 años o más

Te identificas como *

- Hombre
- Mujer
- No binario
- Prefiero no contestar

¿Qué tan frecuente utilizas los medios sociales? *

- Nunca
- Pocas veces
- A veces
- Casi siempre
- Muy frecuente

¿Cuál de los siguientes medios sociales sueles utilizar más en tu día a día? *

- Redes sociales (ex: Facebook, Twitter)
- Blogs (ex: Blogger, WordPress)
- Multimedia (ex: YouTube, Instagram, Snapchat)
- Mundos virtuales (ex: Sims online, Habbo)
- Juegos en línea (ex: World of Warcraft, League of Legends)
- Otros:

¿Cuánto tiempo le dedicas a los medios sociales de media al día? *

- Menos de 30 minutos
- Entre 30 minutos y 1 hora
- Entre 1 y 2 horas
- Más de 2 horas

¿Te preocupa tu privacidad a la hora de utilizar los medios sociales o internet en general? *

Nada Mucho

Una vez hayas terminado de probar el juego, contesta a las siguientes preguntas

¿En algún momento te has sentido incapaz de avanzar? *

- No, en todo momento me ha quedado claro lo que debía hacer
- Si, ha habido momentos en los que no sabía cómo continuar

¿Se han podido visualizar correctamente todos los elementos gráficos (imágenes, botones...) de manera que ninguno de ellos presente una distorsión?

*

- Si, todos los elementos se ven bien
- Algunos presentan distorsiones
- No, todos se ven muy mal

¿Qué te parece el tamaño de letra? *

- Grande
- Tamaño correcto
- Pequeño

¿Crees que hay una coherencia visual entre todos los elementos que han aparecido en el juego? Es decir, presenta un formato constante, un estilo visual similar entre todos los elementos, la música encaja con el juego... *

- Totalmente de acuerdo
- En gran parte sí
- No mucho
- No

¿Cómo valorarías el tiempo de respuesta al realizaran una acción? Por ejemplo, si al clicar a un botón el juego reacciona rápido o lento, si al clicar sobre la pantalla el personaje empieza a moverse al instante o tarda un tiempo en reaccionar *

- Todo reacciona rápido a mis acciones
- A veces ha tardado un poco en reaccionar
- Va con mucho retraso

¿Has podido jugar a todos los niveles? *

- Si, todos funcionan
- No, alguno no se ha iniciado

¿Has podido jugar con todos los personajes que has probado? *

- Si, todos los que he probado funcionan
- No, alguno ha dado error

¿Has logrado identificar lo que significan los obstáculos que has de evitar en el juego sin necesidad de entrar en la definición que aparece en el menú de pausa?*

- Si, todos los obstáculos son una buena representación visual de aquello que representan

- Algunos, pero no todos
 No, no he entendido ningún obstáculo

¿Qué te ha parecido la idea de combinar preguntas quiz dentro del juego? *

- Es una propuesta interesante y encaja bien
Indiferente
 No me acaba de convencer

Valora el nivel de dificultad de las preguntas quiz *

Muy fácil Muy difícil

¿En algún momento el juego ha dejado de funcionar? *

- Si y no he podido seguir utilizándolo
 Si, pero se ha solucionado rápidamente por lo que he podido continuar utilizándolo
 No, todo ha funcionado sin errores

En caso afirmativo, ¿en qué parte del juego ha fallado y cómo ha sucedido este fallo?

Al finalizar la partida, en la pestaña de ranking, ¿aparecen los resultados de las partidas anteriores? Recuerda que los resultados se guardan para cada nivel por separado *

- Si
 No
 No he clicado a ese botón por lo que no lo he podido visualizar

¿Aprendiste algo nuevo? *

Nada Mucho

¿Te pareció un juego interesante? *

Nada Mucho

Para acabar, selecciona aquellos datos que NO deberías publicar en los medios sociales con el fin de proteger tu privacidad *

- DNI
- Número de teléfono personal
- Aficiones / Hobbies
- Dirección de tu casa
- Nombre / Imagen de tu mascota
- Contraseñas
- Días exactos en los que te vas a ir de vacaciones
- Tarjeta de crédito / Información bancaria
- Idiomas que entiendes
- Carné de conducir
- Frase inspiracional

Valoración final / Sugerencias para mejorar el juego
