Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Plataforma web gamificada para la enseñanza de Pensamiento Computacional

Gamified web platform for teaching Computational Thinking

Aranza Carolina Cabrera Castellano

La Laguna, 10 de septiembre de 2019

D. **Carina Soledad González González**, con N.I.F. 54.064.251-Z profesor Titular de Universidad adscrito al Departamento de Nombre del Departamento de la Universidad de La Laguna, como tutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Plataforma web gamificada para la enseñanza de Pensamiento Computacional"

ha sido realizada bajo su dirección por D. **Aranza Carolina Cabrera Castellano**,

con N.I.F. 78.857.287-Q.

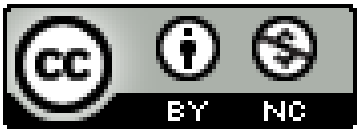
Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de septiembre de 2019

Agradecimientos

A mi tutora por guiarme durante la realización de este trabajo, por resolver cada una de mis dudas y preocupaciones y por el trato siempre cariñoso y positivo.

A mis padres y mi hermana por los consejos y todo el apoyo durante el grado y por no permitir que me rinda ante las dificultades que se han presentado.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial 4.0 Internacional.

Resumen

El objetivo principal de este proyecto es el de desarrollar una plataforma web abierta y de fácil acceso que esté orientada a niños, padres, madres y profesores. De esta manera son los niños los protagonistas y a los que se quiere fomentar el aprendizaje de pensamiento y habilidades computacionales a través de esta web y que su uso esté guiado por profesores, padres, madres o tutores. Se diseñará un sistema basado en los principios que siguen los juegos para motivar a los usuarios y captar su interés para aprender sobre el tema y a su vez que sirva de seguimiento y evaluación a los responsables de estos.

Un sistema gamificado consiste principalmente en recompensar a un usuario según los objetivos que haya alcanzado, de esta manera, se obtiene un reconocimiento positivo que influye directamente en su motivación para mejorar sus habilidades. Alguna de las técnicas para recompensar a los usuarios son la acumulación de puntos, el escalado de niveles, los regalos, las clasificaciones y rankings y la obtención de logros o insignias.

Durante este proyecto se realizarán dos tareas fundamentales, la investigación previa sobre los conceptos tratados para comprender mejor el contexto y la implementación de una aplicación gamificada.

El interés en introducir el pensamiento computacional en recursos educativos proviene de la necesidad de fomentar el pensamiento lógico, algorítmico y técnicas de resolución de problemas para que los jóvenes cuenten con las destrezas necesarias para adaptarse a las tecnologías del futuro. Y la finalidad última es que la aplicación que se va a crear sirva como material didáctico usual dentro de las aulas y cuyo sistema se comporte de la manera adecuada para que su uso sea prolongado.

Palabras clave: gamificación, pensamiento computacional, motivación, educación, sistema.

Abstract

The main objective of this project is to develop an open and easily accessible web platform aimed at children, parents, mothers and teachers. In this way, children are the protagonists and those who want to promote the learning of thinking and computational skills through this website and their use must be guided by teachers, parents, mothers or tutors. A system based on the principles of the games will be designed to motivate users and capture their interest to learn about the subject and at the same time to monitor and evaluate them.

A gamified system consists mainly of rewarding a user according to the objectives he has achieved, in this way, a positive recognition is obtained that directly influences his motivation to improve his abilities. Some of the techniques to reward users are the accumulation of points, the scaling of levels, gifts, rankings and the obtainment of achievements or badges.

During this project, two fundamental tasks will be carried out, prior research on the concepts discussed to better understand the context and implementation of a gamified application.

The interest in introducing computational thinking in educational resources comes from the need to encourage logical, algorithmic thinking and problem-solving techniques so that young people have the necessary skills to adapt to the technologies of the future. The ultimate purpose is that the application to be created will be used as the usual teaching material within the classrooms and whose system behaves in an appropriate manner so that its use will be prolonged.

Keywords: gamification, computational thinking, motivation, education, system.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

1.1 ¿Qué es la gamificación?

La gamificación o ludificación es una metodología de aprendizaje basada en las dinámicas propias de los juegos para facilitar la interiorización de información en los usuarios mediante la potenciación de la motivación. Para ello, se aplican técnicas de psicología con el objetivo de fomentar de forma positiva el aprendizaje, el cual puede consistir en absorber conocimientos, mejorar habilidades, o recompensar acciones concretas.

El concepto de gamificación fue construido según las definiciones de varios expertos, por ejemplo, según los autores del libro ***Gamificaction by Desing***, *Gabe Zichermann* y *Christopher Cunningham*, la gamificación es "un proceso relacionado con el pensamiento del jugador y las técnicas de juego para atraer a los usuarios y resolver problemas" [1]. Y según *Karl. M. Kapp* consiste "en utilizar las mecánicas y dinámicas de los juegos, la estética y el uso del pensamiento para atraer a las personas, incitar a la acción, promover el aprendizaje y resolver problemas"[2].

Finalmente, se debe destacar la definición de *Nick Pelling de 2002*, la cual defiende que "La gamificación es la aplicación de metáforas de juego para tareas de la vida real que influyen en el comportamiento y mejoran la motivación y el compromiso de las personas que se ven implicadas". Pero la definición más acertada hasta el momento es la de Deterding et al, que define la gamificación como "el uso de elementos de diseño de juego en contexto que no son de juego" [3].

Esta se convertiría en la definición oficial, pero todos los autores coinciden en que el uso de ciertos elementos presentes en los juegos generar un incremento en el tiempo de uso del usuario y su disposición psicología a continuar jugando.

1.2 ¿Qué es el Pensamiento Computacional?

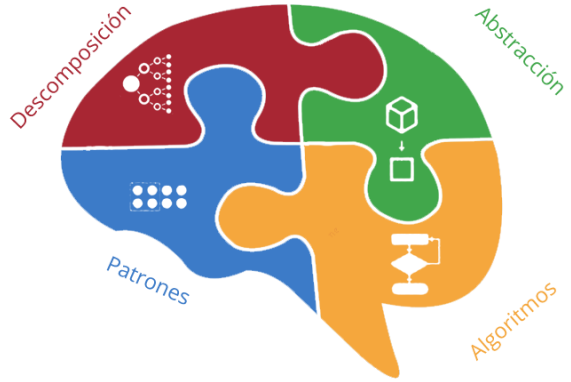
El pensamiento computacional definido por Jeannette M. Wing dicta que "el pensamiento computacional implica resolver problemas, diseñar sistemas y comprender el comportamiento humano, haciendo uso de los conceptos fundamentales de la informática" [4]. A pesar de tener definiciones variadas, se podría definir además como un conjunto de habilidades esenciales para reconocer los aspectos computables que existen en el entorno que nos rodea y para aplicar las herramientas y técnicas de la ciencia de la computación. No obstante, estas habilidades son útiles para todas las personas, no solo para los que se dedican al mundo de la computación. Proporciona a la sociedad la capacidad de enfocar

aspectos de la vida cotidiana como una serie de puzles que se pueden dividir en partes más pequeñas, permitiendo resolverlo poco a poco a través de la lógica y el pensamiento deductivo.

Un problema complejo es uno que, en principio, no se sabe cómo resolver, pero si se toma y se divide en una serie de problemas más pequeños y manejables, cada uno de ellos puede analizarse individualmente. Se pueden considerar resoluciones previas ante condiciones similares y centrarse solo en los detalles fundamentales, ignorando lo irrelevante para ese caso. Cuando un problema es resuelto, se puede establecer una colección de instrucciones simples que resolverán cada problema pequeño, que, tras culminar, proporcionan la resolución del problema mayor.

Esta explicación es una conceptualización de las distintas fases del pensamiento computacional: **Descomposición**, **Reconocimiento de patrones**, **Abstracción** y **Diseño de algoritmos**. Se profundizará en estas fases más adelante.

Figura 1. Fases del Pensamiento Computacional



1.3 Objetivos

Para este proyecto se propone desarrollar un sistema que cuente con las propiedades de la gamificación para proporcionar un medio que facilite a los protagonistas la interiorización de la teoría y la práctica del pensamiento computacional. Estos protagonistas principalmente serán niños y jóvenes en edades escolares (entre 5 años y 16 años), pero también puede participar en la plataforma cualquier persona.

Se propone a su vez investigar y desarrollar una aplicación web progresiva o PWA con el objetivo de caracterizar a la plataforma que se va a implementar con un comportamiento similar al que tiene una aplicación nativa de un dispositivo, con la ventaja de ser portable bajo cualquier sistema operativo, debido a que seguiría siendo una aplicación web.

Este sitio será accesible, responsivo y diseñado correctamente para que pueda ser utilizado por cualquier individuo, y, sobre todo, que pueda ser usado por niños de todas

las edades.

La aplicación deberá garantizan la privacidad de la información suministrada por el usuario mediante la encriptación, protegiendo de esta forma el proceso de transmisión de datos entre el usuario y el servidor. Por lo tanto, deberá contar con un certificado SSL.

A pesar de tratarse de un proyecto individual, se pretende aplicar prácticas propias de las metodologías ágiles con el objetivo de mejorar la planificación, estimación y gestión del proyecto.

1.4 Metodología de desarrollo

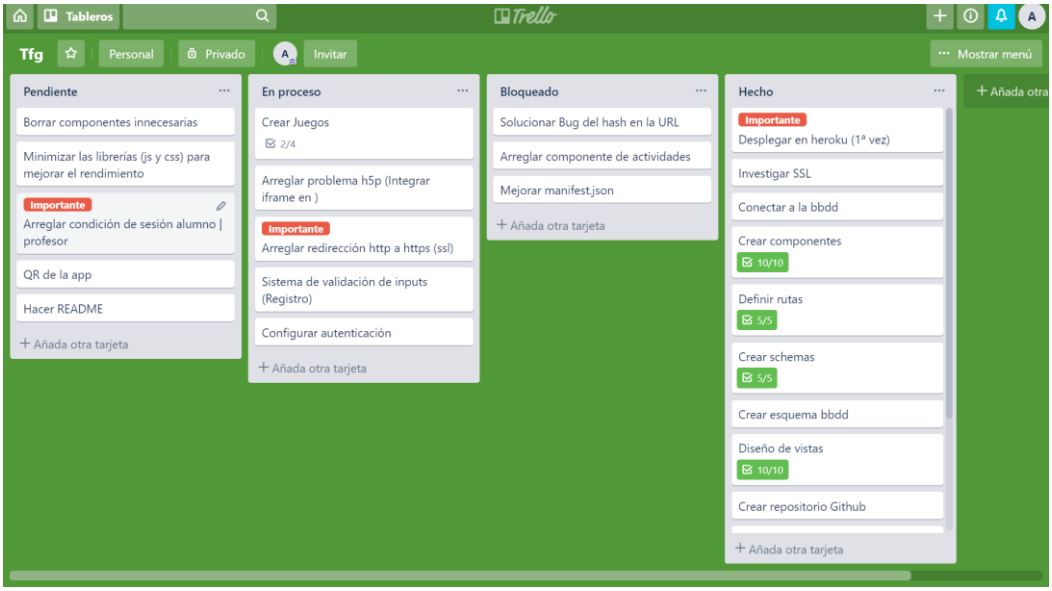
Algunas metodologías como la *Scrum* o la *Lean* son factores de éxito en el desarrollo de un proyecto web o software. Las metodologías agiles permiten trabajar con mayor velocidad y eficiencia, mejorar la calidad del producto y el trabajo del equipo desarrollador. Sin embargo, este proyecto será realizado de forma individual, por lo que una de las metodologías más adecuadas para este caso es la de *Kanban*.

La metodología *Kanban* consiste en la virtualización del trabajo en un esquema dividido en diferentes estados por los que van transcurriendo las tareas: Pendientes, en proceso, bloqueadas y listas. Las características que la definen son las siguientes:

- Visualización de todas las tareas
- En proceso
- Priorización según importancia y urgencia
- Seguimiento del tiempo

Se ha elegido a herramienta *Trello* porque resulta perfecta para aplicar *Kanban* debido a su aspecto visual y su configuración.

Figura 2: Tablero de Trello del proyecto



Capítulo 2

Antecedentes y estado actual del tema

2.1 Contexto

Los sistemas educativos cambian para adaptarse a las necesidades del momento y preparar a los jóvenes para el futuro que vivirán. Las necesidades actuales están estrechamente relacionadas con el mundo digital y por ello, los sistemas educativos se deben volver a plantear. Además de formar el alfabetismo lingüístico y numérico, también se deberá tratar el alfabetismo digital. Este cambio debe ser suficiente para dotar a los jóvenes de las herramientas cognitivas necesarias para saber desenvolverse con éxito. Pero el enfoque que se le ha estado dando dentro de la educación no es el adecuado para la mayor parte del sistema educativo no universitario. El cambio adaptativo que se suele hacer es introducir clases de informática donde se enseña a "programar" o donde se incluyen contenidos demasiados concretos. Y aunque no dejan de ser componentes importantes, no proporcionan en su totalidad las capacidades que se requieren para resolver problemas en el mundo digital.

Pero en los últimos años, se han implantado exitosamente metodologías más actuales y atractivas para los alumnos en algunos centros alrededor del mundo. Las expectativas y necesidades tecnológicas de la sociedad actual exigen a los profesores e instituciones a innovar en tecnologías emergentes que incorporen estrategias para aumentar la motivación y herramientas para favorecen el aprendizaje autónomo en las aulas. Y esto se ha logrado a través de la metodología de la gamificación. Incluir mecánicas de juegos en las clases genera en los alumnos importantes beneficios, ayuda a mantener su interés y evita que el proceso de enseñanza se convierta en algo aburrido.

Dentro de estas clases gamificadas, se pueden incluir complementariamente dispositivos electrónicos como móviles, tabletas e incluso ordenadores. Pero los más adecuados son aquellos con los que los jóvenes y los niños están más familiarizados y aquellos que son más fáciles de usar y transportar. Por tanto, el desarrollo de contenido preparado para este tipo de dispositivos es esencial para una experiencia digital en los procesos educativos.

En la actualidad, las aplicaciones, sitios web, programas y demás elementos digitales deben cumplir ciertos requisitos en su desarrollo para adecuar su uso a las personas. Y como los participantes del sistema educativo pueden abarcar todas las edades y una gran variedad de necesidades diferentes, es fundamental para un sistema gamificado, contar con pautas de usabilidad y accesibilidad.

Además de la educación, la gamificación representa una herramienta de motivación exitosa para otros entornos a los que se puede adaptar como pueden ser el empresarial, el personal, en recursos humanos, marketing, etc.

2.2 Plataformas actuales de gamificación

2.2.1 Para empresas

En el mundo empresarial se busca el compromiso de los clientes y los trabajadores. Por un lado, un sistema gamificado puede ayudar a obtener *engagement*, es decir la participación a través de estrategias que fomentan la fidelización, y, además, la experiencia positiva que genera en el usuario favorece a que este se convierta posteriormente en un recomendador de la marca. Y, por otro lado, es una tendencia actual aplacar el descontento de los empleados mediante mecánicas del juegos. Además de reforzar positivamente la imagen de la propia empresa sobre el empleado, favorece la relaciones con los compañeros, despierta su espíritu competitivo y mejora el ambiente laboral

Respecto al marketing basado en gamificación, podemos considerar como casos de éxitos la siguientes aplicaciones:

Nike+ [20]: consiste en una red social que permite "trackear" los pasos y las pulsaciones de un usuario que esté realizando alguna actividad deportiva. Este sistema cuenta las pulsaciones, los kilómetros, tiempos y calorías quemadas y además permite establecer una meta diaria. El usuario puede consultar esta información y esforzarse para superar sus puntuaciones.

Starbucks Rewards [21]: La compañía Starbucks diseño un programa de fidelización para retener a los nuevos clientes gracias a tarjetas prepago de regalo. Esta experiencia les permite ganar "estrellas" en función de las compras que haga mediante la tarjeta, y con ellas obtener *refills* de café, ofertas y productos gratuitos. Además, incluye una barra de progreso y una serie de logros.

2.2.2 Para Educación

Ya se ha comentado anteriormente la integración de la gamificación en entornos educativos, y a continuación se exponen algunos buenos ejemplos de herramientas gamificadas para el aula.

ClassDojo [22]: es un sistema dedicado a fomentar una conducta positiva por parte de los alumnos en el interior del aula. Cuenta con versiones para docentes, alumnos y padres y madres, donde pueden estar al tanto del progreso de sus hijo. La plataforma premia con puntos, insignias positivas y negativas según el desempeño de los alumnos en seis conductas (Ayuda a sus compañeros, tarea, participación, perseverancia, trabajo en equipo e intensidad del trabajo).

Figura 3: Insignias positivas de ClassDojo

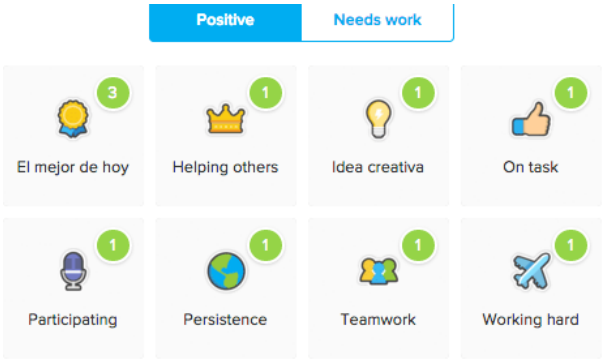
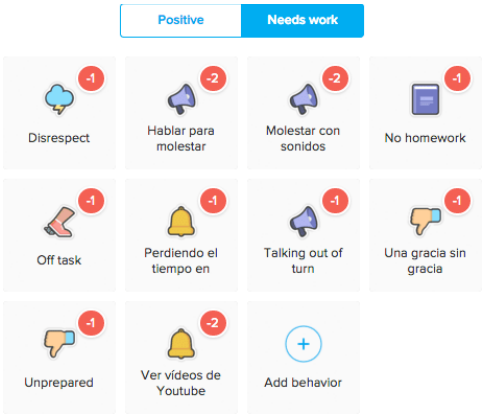


Figura 4: Insignias negativas de ClassDojo



***Kahoot [23]*:** es una plataforma orientada para su uso con dispositivos móviles en el aula donde los profesores pueden crear cuestionarios sobre cualquier tema y preparar concursos donde los alumnos son los concursantes. Cada alumno puede consultar su puntuación según las preguntas correctas que haya obtenido y el tiempo que haya usado. Y finalmente puede consultar su estado dentro de una clasificación que se organiza según las puntuaciones de sus compañeros de clase.

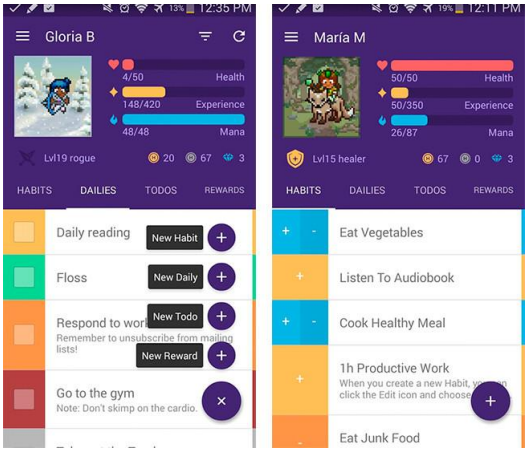
2.2.3 Para Hábitos

Se pueden aplicar también los principios de la gamificación para mejorar la vida del usuario, para conseguir ser más productivo o para logar los objetivos que quiera conseguir. Estos hábitos son costumbres que resultan difíciles de cambiar o que forman parte de la rutina de cada día, y que con la gamificación se obtiene un impulso motivacional para realizarlos. Algunos buenos ejemplos de este tipo de gamificación son las siguientes herramientas:

Fitbit [24]: es un caso a su vez de gamificación en marketing, debido a que el fabricante de pulseras Fitbit añadió desafíos como medida para aumentar el uso de las pulseras por parte de sus usuarios. Estas pulseras son seguidores de actividad que almacenan las marcas de cada usuario y los animan a lograr más pasos, competir, comparar las marcas y obtener logros.

Habitica [25]: es una aplicación que ayuda al jugador a mantenerse motivado para alcanzar sus objetivos. El jugador establecerá sus propias metas según los hábitos que quiera cambiar. Pueden ser hábitos positivos o negativos y tras realizarlo diariamente obtendrá puntos o los perderá respectivamente.

Figura 5: Perfil de usuario en Habitica.



2.3 Plataformas gamificadas sobre Pensamiento Computacional

Resulta conveniente potenciar la enseñanza de habilidades computacionales utilizando medios digitales. De esta manera se puede ofrecer una gran cantidad de contenido variado a los usuarios de manera sencilla a través de sus dispositivos móviles. Además, algunas plataformas pueden replicar mecánicas del pensamiento computacional de forma muy visual e intuitiva.

Scratch [26]: Es una de las herramientas más destacadas para desarrollar el pensamiento computacional. Consiste en un lenguaje de programación visual para niños y jóvenes que está orientado para usarse en los primeros pasos cuando se aprende a programar. Permite a los usuarios crear sus propios proyectos animados y les proporciona la capacidad de entender cómo pueden transformarse instrucciones en algo concreto que se quiera conseguir.

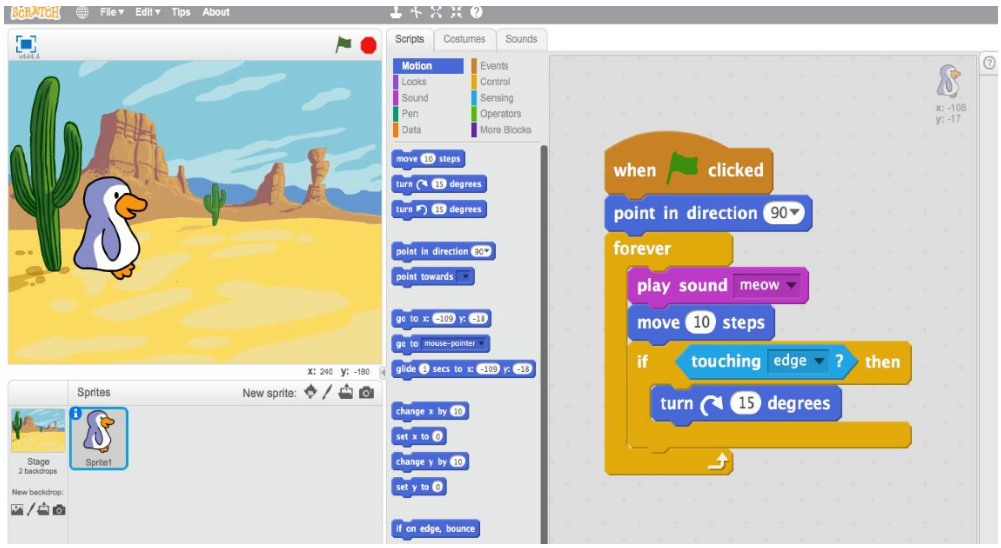


Figura 6: Creando una animación en Scratch

Capítulo 3
Definición de conceptos

3.1 Técnicas y componentes de la gamificación

Las personas estamos acostumbradas a adoptar ciertos comportamientos para conseguir bienes, por lo que un sistema gamificado diseñado correctamente puede lograr el objetivo que se ha propuesto para él. Para comprender mejor cómo están constituidos estos sistemas, se deben distinguir los conceptos de elementos, dinámicas y mecánicas.

3.1.1 Elementos

- **Idea del juego**: El objetivo que se quiere lograr a través del juego.
- **Mecánica**: La incorporación de recompensas que gana el usuario, con las que se fomenta a que continúe participando.
- **Conexión juego-jugador**: Es la relación que tiene el juego con el jugador y viceversa. El juego debe permitir al usuario encontrar lo que busca con relativa facilidad para evitar su frustración.
- **Jugadores**: Los sistemas gamificados pueden estar destinados a toda clase de perfiles de jugadores diferenciados por edad, sexo, ideas, intereses, ocupación, etc.
- **Motivación**: Es la predisposición psicológica del usuario a participar en el juego, y que se debe alimentar constantemente para evitar su abandono. Sin embargo, demasiada motivación puede saturar al usuario.

3.1.2 Dinámicas

Las dinámicas son las formas en la que se ponen en marcha las mecánicas del juego y determinan el comportamiento y la motivación de los usuarios.

- **Narrativa**: elemento estético con el que el usuario se apoya para comprender un camino de aprendizaje. Debe contar con ficción, fantasía y misterio con el objetivo de hacerla más entretenida y así captar la atención del usuario. Debe estar relacionada con el objetivo y la temática del juego.
- **Restricciones y contenido bloqueado**: limitando la libertad, se incentiva al usuario a seguir jugando para desbloquear el entorno.
- **Emociones**: Estas pueden ser el refuerzo de la curiosidad, la competitividad, aumentar el compromiso, etc.

- **Interacción social**: se fomenta la necesidad del usuario a interactuar con otros, por ejemplo, a través del reconocimiento social o logrando un status, un nivel o cierta puntuación.
- **Progreso o *Feedback*:** los usuarios necesitan algún tipo de medida de su progreso para reconocer lo que han hecho con anterioridad, y para darse cuenta de que no lo han hecho en vano.

3.1.3 Mecánicas

Las mecánicas consisten en los componentes básicos del juego, lo que dirige su funcionamiento y sus reglas.

- **Desafíos:** ayudan a mantener el interés del usuario porque le hace sentirse retado. Ponen a prueba sus conocimientos y le permite demostrarlos y lograr una recompensa por ello.
- **Competición**: Se utiliza a través de la clasificación o comparación entre los usuarios. Gestionada en una buena medida, resulta una técnica infalible para mantener el interés de los jugadores.
- **Recompensas:** son los beneficios obtenidos por lograr los objetivos.
- **Turnos:** consiste en una participación secuencial, equitativa y alternativa.

3.1.4 Componentes

Las componentes son los recursos y herramientas con las que cuenta un sistema gamificado. Se escogen para diseñar las actividades con las que contará.

- **Logros y colección de logros**: son distintivos o señales por lograr objetivos y son muy valorados por los usuarios.
- **Insignias o *badges*:** son representaciones visuales de los logros **.**
- **Avatares**: Representación visual del jugador en su perfil.
- **Combate**: Batalla definida entre usuarios.
- **Clasificaciones o rankings**: es la representación visual donde el usuario puede comparar su progreso con los demás usuarios.
- **Puntos**: Son las recompensas que representan la progresión del usuario y que le certifican de que su esfuerzo tiene valor.
- **Niveles**: Son diferentes estadios de progresión o dificultad. Se superan cuando se alcanza cierta cantidad de puntos.
- **Equipos**: Trabajo en grupo con un objetivo común.

- **Misiones o retos**: Desafíos predeterminados con objetivos y recompensas.

3.1.5 Fases del Pensamiento Computacional

El pensamiento computacional se puede proyectar sobre distintos ámbitos y en cada caso se manifiestan las 4 fases específicas que lo definen. En orden de ejecución, estas fases son:

i. **Descomposición de un problema**: consiste en dividir una tarea compleja en pasos más sencillos, de manera que se pueda indicar el proceso en forma de instrucciones.

ii. **Reconocimiento de patrones**: consiste en identificar patrones, similitudes, conexiones o diferencias comunes en los pasos del problema para hacer predicciones que conducen a atajos dentro del proceso. Esta fase es la base necesaria para construir un algoritmo.

iii. **Generalización de patrones y abstracción**: consiste en desechar la información irrelevante y generalizar la que es necesaria para resolver el problema con el objetivo de actuar de manera eficiente. Es una forma de resolver rápidamente los problemas sobre la base de la experiencia.

iv. **Diseño de un algoritmo**: en esta fase se desarrolla una estrategia siguiendo una secuencia de las instrucciones discretas obtenidas para resolver el problema en cuestión. En las ciencias de la computación, un algoritmo se diseña de manera abstracta utilizando variables.

La finalidad de seguir este proceso de cuatro pasos es la de definir un método eficiente y eficaz para reproducir una determinada tarea. Como se ha mencionado anteriormente, se puede aplicar el pensamiento computacional de distintas maneras, pero principalmente, se diseñan algoritmos para retransmitírselo a un sistema de cómputo con órdenes que comprenda y que transcriba para que resuelva problemas reiteradamente de forma autónoma.

Capítulo 4
Plataforma web *Gami*

4.1 ¿En qué consiste?

Gami es la una plataforma web gamificada que ha resultado del trabajo empleado en este proyecto. Su propósito es el de enseñar los conceptos del pensamiento computacional con mecánicas de juegos. Proporciona un entorno basado en sesiones donde un usuario puede crearse una cuenta que lo identifique y que le permita acumular puntos y logros que gane. Además, se puede contribuir al crecimiento de esta plataforma eligiendo otro tipo de perfil para crear desafíos relacionados con el pensamiento computacional, que servirán para que los demás usuarios comprendan conceptos computacionales. Cuenta con dos elementos interactivos principales: Juegos y desafíos, los cuales se describirán más adelante.

4.2 Tipos de actividades

La plataforma consta de dos actividades principales que son:

Desafíos: Consisten en pequeñas tareas para realizar en casa o en el aula. Están destinadas para que el usuario comprenda una lección del pensamiento computacional. Están clasificadas según sus cuatro fases: Reconocimiento de patrones, descomposición, diseño de algoritmos y abstracción. En primer lugar, un profesor deberá crear un desafío desde su perfil y una vez creado un alumno podrá verlo en la página Desafíos. Cuando la consulte, la leerá y si decide realizarla deberá seguir los pasos que se indican. Cuando termine rellenará un formulario para demostrar que la ha realizado y pueda ser evaluada por el mismo profesor que la creó. Cuando este la evalúe, directamente se le sumarán los puntos correspondientes al alumno.

Juegos: Son distintos tipos de juegos o actividades donde el usuario interacciona con distintos elementos para elegir la opción correcta. Cada juego abarca un tema y se clasifican como los desafíos.

4.3 Usuarios

Los usuarios que se registren pueden ser tanto "alumnos" como "profesores" dependiendo de sus intenciones dentro de la web.

Los "**alumnos**" serán los usuarios que quieran aprender pensamiento computacional, quienes completarán desafíos, jugaran a juegos, ganarán puntos y logros y estarán incluidos en el ranking.

Los "**profesores**" serán los usuarios que creen desafíos desde su perfil para que los alumnos las realicen.

En esta aplicación puede participar cualquier persona, no necesariamente deben ser alumnos y profesores. Está diseñado para que el usuario que pretende ampliar sus habilidades computacionales tome el rol de alumno y realice tareas que diseñan usuarios con el rol de profesor. Los perfiles de alumnos se encuentran divididos en periodos de edades para limitar a los usuarios de menor edad realizar tareas más complejas orientadas para edades superiores. El periodo que indica cada actividad denomina el periodo más bajo que puede realizarla.

Los periodos están organizados de la siguiente manera:

- Infantil – Edades de 5 a 8 años
- Junior – Edades de 5 a 8 años
- Senior – Edades de 5 a 8 años
- Pro – Edades de 5 a 8 años

4.4 Funcionamiento

En la página principal del sitio web se puede encontrar una guía para los pasos iniciales. Lo primero es registrarse rellenando un formulario con campos distintos para alumnos y profesores. Todos los campos del formulario deben estar introducidos correctamente para poder registrarse.

Si se entra como alumno, se deben indicar nombre y apellidos, la edad para poder identificarlo dentro de un periodo, correo electrónico para identificarlo de los demás usuarios y una contraseña que le permita entrar en el sistema. Por el contrario, si entra como profesor únicamente harán falta los campos del nombre, apellidos, correo electrónico y contraseña.

Una vez registrado puede iniciar sesión para acceder a su perfil personal de acceso único para ese usuario. El perfil de alumno es completamente diferente al perfil de un profesor debido a que desempeñarán funciones distintas dentro de la plataforma; el alumno estará centrado en conseguir recompensas y el profesor en crear contenido adecuado para los alumnos. Estos obtendrán un feedback positivo o negativo de cada alumno que realice alguno de sus desafíos que le ayudará a mejorar en el futuro.

4.5 Páginas

La web Gami está compuesta por la siguientes páginas:

Inicio: Es la página principal de la web. Da un mensaje de bienvenida y muestra una serie de pasos para crear una cuenta. Cualquier persona que entre a la plataforma tiene acceso a esta página sin necesidad de haber accedido a una cuenta.

Figura 7: Página de inicio

Registro: Página que permite crear una cuenta. Consta de un formulario con campos obligatorios que deben estar rellenados de forma correcta para poder continuar. Ofrece dos tipos de registro, uno para alumnos y otro para profesores con diferencias en los campos. Cualquier persona que entre a la plataforma tiene acceso a esta página sin necesidad de haber accedido a una cuenta.

Figura 8: Página de registro

Inicio de sesión: Es el paso siguiente al registro. Se deben introducir el correo electrónico y la contraseña del usuario correctamente. Al acceder, se redirecciona al perfil del propio

usuario. Cualquier persona que entre a la plataforma tiene acceso a esta página sin necesidad de haber accedido a una cuenta.

Figura 9: Página de inicio de sesión

Tutorial: Es una página de contenido estático que sirve para resolver algunas posibles dudas durante el uso de la plataforma. Contiene un conjunto de apartados a los cuales se puede acceder haciendo clic. Cualquier persona que entre a la plataforma tiene acceso a esta página sin necesidad de haber accedido a una cuenta.

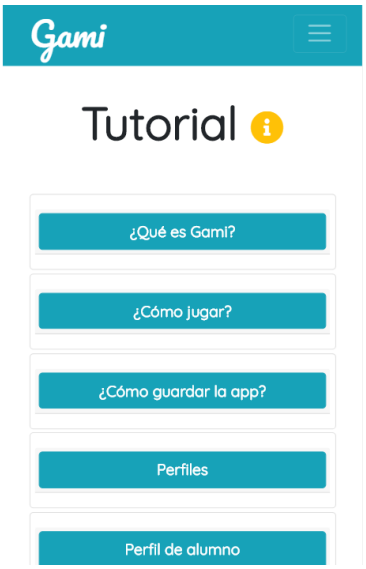


Figura 10: Página de tutorial

Ranking: En esta página se encuentra la clasificación de todos los usuarios de tipo "alumno" ordenados según los puntos en orden descendente. Cualquier persona que entre a la plataforma tiene acceso a esta página sin necesidad de haber accedido a una cuenta.

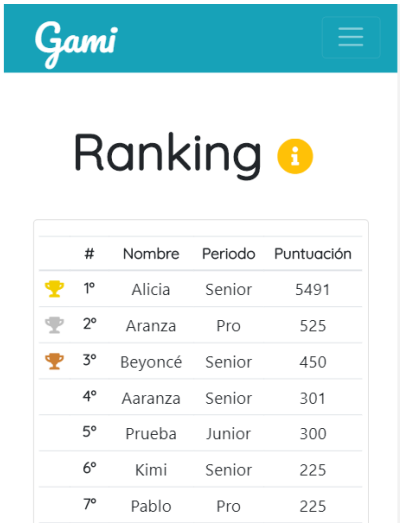


Figura 11: : Página de ranking

Juegos: Es una página donde se encuentran en una tabla todos los juegos disponibles. Permite filtrar según la categoría del juego. Haciendo clic sobre el botón "Jugar" se puede acceder a la página de cada juego.

Figura 12: : Página de juegos

Desafíos: Es una página donde se encuentran en una tabla todos los desafíos disponibles. Permite filtrar según la categoría del desafío. Haciendo clic sobre el botón "Ver" de cualquier desafío se abre una tarjeta donde aparecen los datos correspondientes a ese desafío: título, categoría, periodo y descripción de la actividad. Si el alumno decide que quiere completar ese desafío, tras haber hecho lo que ponía la descripción, deberá rellenar un cuestionario que se abre haciendo clic en el botón "Realizar". En este cuestionario deberá valorar en primer lugar si le ha parecido divertido o aburrido, describir cómo la ha resuelto, y opcionalmente añadir una imagen del proceso, para ayudar a demostrar que la ha realizado correctamente.

Cuando la envíe, se cerrará la ventana y se enviará directamente al profesor que la ha creado como solución.

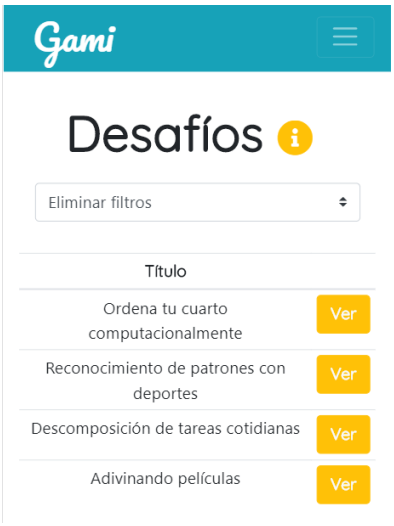


Figura 13: Página de desafíos

Perfil de usuario: consiste dos tarjetas a los laterales de la página. En la primera aparecen sus datos personales, una imagen de perfil acorde con su estatus y nivel actual, una barra de progreso que indica los puntos respecto al siguiente nivel, el número de juegos que ha completado con la mayor puntuación y la racha, que son los días seguidos que lleva conectándose a la plataforma. En la otra tarjeta aparecen los logros que ha conseguido hasta el momento. Estos usuarios pueden jugar a los juegos de la página Juegos o completar desafíos de la página Desafíos.

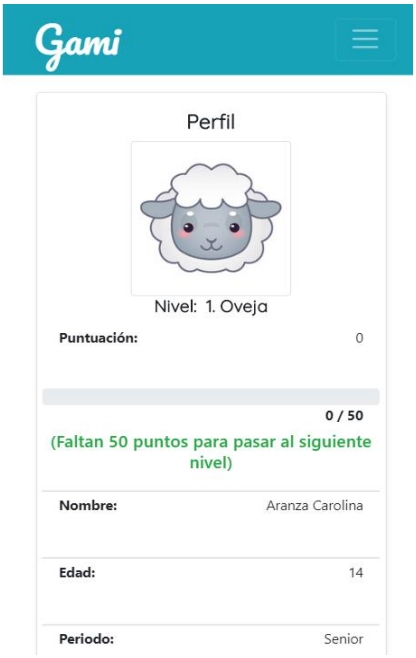


Figura 14: Perfil de alumno(I)

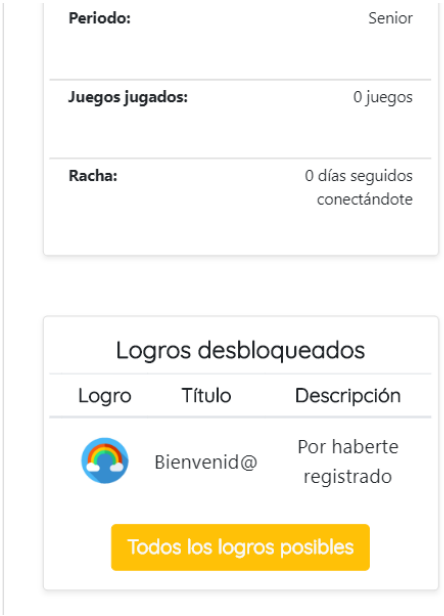


Figura 15: Perfil de alumno(II)

Perfil de profesor: consta de sus datos personales, el número de desafíos que ha creado, los desafíos que ha creado, y todas las soluciones a sus desafíos realizados por alumnos. Desde esta página pueden realizar dos funciones: crear desafíos y valorar las soluciones aportadas por los alumnos. Ambas se realizan mediante un formulario.

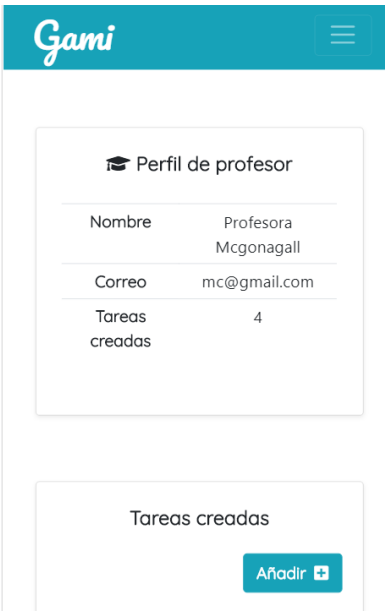


Figura 16: Perfil de profesor(I)

Figura 17: Perfil de profesor(II)

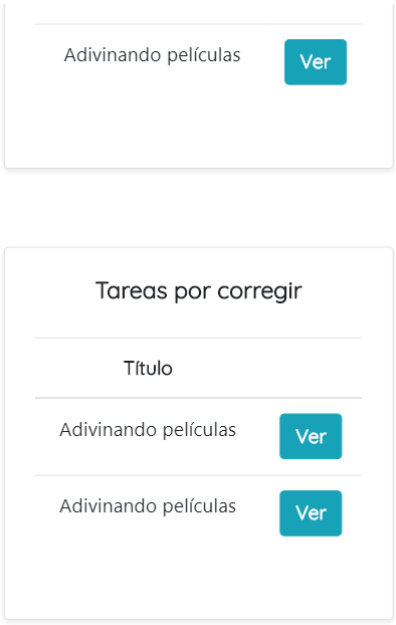


Figura 18: Perfil de profesor(III)

Logros: Es una página a la que se accede desde la página "Tutorial" y la de perfil del alumno, que consta únicamente de una tabla donde aparecen todos los logros disponibles.

4.6 Sistema gamificado

A continuación, se mencionan los elementos de gamificación presentes en la web.

4.6.1 Obtención de logros

Los logros son pequeñas insignias que aparecen en el perfil del alumno según las vaya adquiriendo.

Logros de nivel: Por cada nivel ganado, el usuario ganará un logro con el número y animal correspondiente.

Figura 19: Logros de nivel

Logro "Bienvenid@": Cuando el alumno inicia sesión por primera vez ganará un logro que le felicita por haberse registrado.

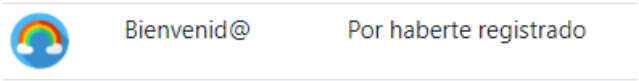


Figura 20: Logro "Bienvenid@"

Logro "En la cima": Cuando un alumno llega al primer puesto del ranking por primera vez gana este logro.

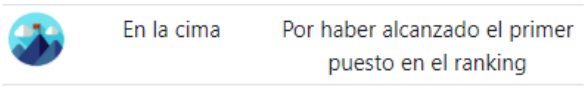


Figura 21: Logro "En la cima"

Logro "Invencible": Cuando el alumno supera todos los niveles y llega al último consigue este logro.

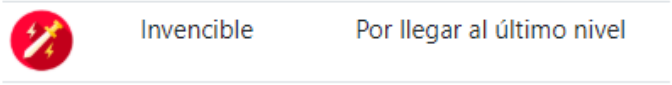


Figura 22: Logro "Invencible"

Logro "Enganchad@": Cuando la racha del alumno es igual a 7 días, o sea una semana, gana este logro.

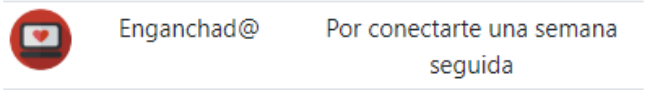


Figura 23: Logro "Enganchad@"

Logro "Premio a la perseverancia": cuando la racha del alumno es igual a 30 días, o sea un mes, gana este logro.

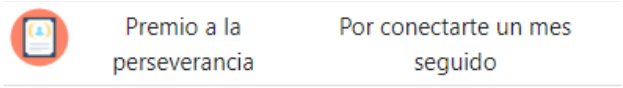


Figura 24: Logro "Premio a la perseverancia"

Logro "En la diana": cuando el alumno consigue 5 puntuaciones completas o perfectas en juegos gana este logro. Esto solo se aplica para 5 juegos distintos.

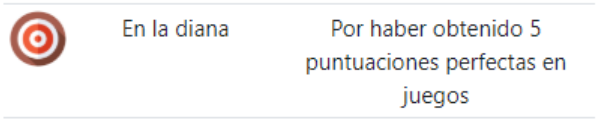


Figura 25: Logro "En la diana"

Logro "Pro del pensamiento computacional": Cuando el alumno consigue 15 puntuaciones completas o perfectas en juegos gana este logro. Esto solo se aplica para 15 juegos distintos.

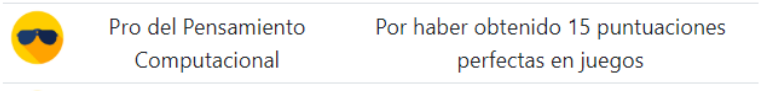


Figura 26: Logro "Pro del pensamiento computacional"

4.6.2 Rankings y competición

La web cuenta con una clasificación sencilla de alumnos que se ordena por puntuación total en orden descendente. Cuando un alumno tiene más puntuación que todos los demás consigue un logro por haber llegado al primer lugar. En la tabla del ranking aparecen los nombres, niveles y puntuaciones de cada alumno. De esta manera, los alumnos podrán comprobar sus progresos y competir con los demás jugadores para subir de puestos respecto a los demás.

4.6.3 Acumulación de puntos

La acumulación de puntos está destinada únicamente hacia los alumnos. Se pueden obtener mediante desafíos o juegos y se calculan en base a los resultados. En el caso de los desafíos, se aplica un factor de valor 5 a la calificación que da el profesor, la cual es de 0 a 10, por lo que los puntos disponibles son múltiplos de 5, siendo la puntuación más baja 0 y la más alta 50. En los juegos se puntúa de forma distinta, debido a que lo juegos tienen diversas puntuaciones totales. Al alumno se le asignan los puntos según los puntos obtenidos respecto del total, así se califica acorde con sus conocimientos en cada juego. De la misma forma, la puntuación mínima es 0 y la máxima de 50 puntos.

4.6.4 Escalado de niveles

El aumento de los puntos viene acompañado de unos límites que representan niveles. Un alumno logra subir de nivel cuando supera el límite superior del nivel en el que se encontraba. Entonces sus puntos relativos se resetearán para el nivel nuevo pero sus puntos totales no sufrirán cambios.

4.6.5 Incentivos mediante rachas

Cuando el participante inicia sesión dos días seguidos su racha aumentará en una unidad. Se comprueba si se ha conectado al menos 2 veces en las últimas 24 horas. Cuando este valor sea mayor que uno, se multiplica por la puntuación obtenida en cualquier juego o desafío, de manera que, si sigue conectándose cada día, cada vez le resultará más fácil subir de nivel.

4.6.6 Estatus

Cada nivel tiene asignado un estatus representado con imágenes de animales de menor a mayor tamaño o nivel de peligrosidad, como metáfora del aumento de los conocimientos con cada nivel.

Capítulo 5
Fases del proyecto

Como enfoque metodológico para este trabajo se ha elegido seguir el desarrollo en cascada, que consiste en una secuencia de etapas ordenadas enfocadas para desarrollar software de forma efectiva y eficiente. Sigue un principio simple de definir antes que diseñar y diseñar antes que codificar. Existen varias variaciones de este modelo, pero la que se ha utilizado para este proyecto es el que se divide en las siguientes etapas:

1. Planificación, análisis y especificación de los requisitos
2. Diseño
3. Implementación
4. Despliegue
5. Mantenimiento

5.1 Planificación, análisis y especificación de requisitos.

En esta fase se analizan las necesidades de los usuarios finales de la herramienta y las exigencias del sistema que se pretende desarrollar.

Como se trata de una herramienta que las personas no han exigido, sino que se les proporciona como complementación a su aprendizaje, se deben prever las necesidades que tendrán cuando la estén utilizando. Por lo que la aplicación deberá contar con los siguientes requisitos:

- Disponibilidad 24 horas.
- Totalmente responsiva.
- Ofrecer conexión segura mediante un certificado SSL (Secure Sockets Layer).
- Accesible y usable
- Portable mediante cualquier navegador
- Capaz de crecer gracias a una comunidad
- Escalable

5.1.1 Investigación y elección de tecnologías

Para proporcionar todas estas características a la plataforma web en cuestión, se han elegido las tecnologías web conscientemente. En el momento inicial se ha establecido que este proyecto consistía en una plataforma web donde los usuarios podrán registrarse para realizar actividades individuales y sencillas para aprender pensamiento computacional y que obtendrán recompensas por ello. Esto implica que existirá un cliente y un servidor en todo momento y que el cliente realizará peticiones constantemente al servidor para obtener y

enviar datos. En general, la carga de la web será relativamente ligera porque los datos que se transmitirán serán cadenas de textos, valores numéricos y arrays.

Se ha elegido seguir una de la técnicas más viables y atractivas para trabajar con un modelo cliente-servidor, el cual consiste en el desarrollo basado en API REST.

Una API REST es una interfaz de programación que está apoyada en la arquitectura REST para el desarrollo de aplicaciones web. REST proviene de "REpresentational State Transfer" lo que quiere decir que no tiene estado, y que, en ejecución, el servicio pierde todos los recursos que ha utilizado. Por lo tanto, un cliente no puede esperar que el sistema recuerde sus datos, sino que tiene que encargarse de hacer una llamada identificándose. El cliente inicia una solicitud, a través de JavaScript y el lenguaje del lado del cliente solicitará al servidor un recurso. El formato de intercambio de información comúnmente suele ser JSON o incluso XML. Posteriormente, no es necesario que el servidor escriba HTML, sino que genera los datos para enviárselos al cliente.

Para manipular estas solicitudes, los componentes de la red, que son clientes y servidores, se comunican a través de HTTP. Cada mensaje HTTP contiene toda la información necesaria para comprender la petición que puede hacer tanto el cliente como el servidor. HTTP define un conjunto de operaciones conocidas para transferir recursos que se suelen equiparar con las operaciones CRUD de las bases de datos. Estas operaciones son POST, GET, PUT y DELETE.

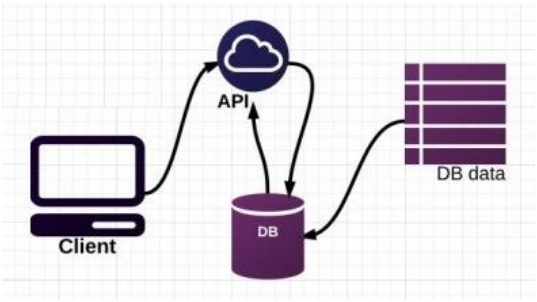


Figura 27: Esquema básico de las API REST

Para facilitar el trabajo, el mantenimiento, la escalabilidad y el trabajo con contenido dinámico en la web, se elegirá un framework para el lado del cliente y otro para el lado del servidor. Según las necesidades de este proyecto se ha considerado que como framework del lado del servidor es conveniente utilizar NodeJs. Este es un entorno de ejecución del lado del servidor y orientado a eventos. que se ejecuta sobre el intérprete de JavaScript V8 creado por Google. Fue diseñado para generar un sistema escalable y con la consistencia suficiente como para generar un elevado número de conexiones de forma simultánea con el servidor.

Figura 28: Logo de NodeJs

NodeJs cuenta con el *Node Package Manager* (*npm*) que es su administrador de paquetes preinstalado. Facilita la compilación, instalación y actualización de módulos, así como la gestión de las dependencias y da acceso a un conjunto de librerías muy extenso.

Figura 29: Logo de npm

El lenguaje del lado del cliente más significativo es *JavaScript* debido a que es un lenguaje de programación web universal en los navegadores más relevantes. Uno de los principales frameworks de *JavaScript* es *Vue.js*. Este es un framework de código abierto para construir interfaces de usuario y aplicaciones SPA(Single-Page Application). Es progresivo, lo que significa que se puede utilizar de forma básica o compleja. Esto es posible gracias a que está modularizado en diferentes librerías, de manera que se pueden ir añadiendo funcionalidades según se vayan necesitando. Una de esas librerías es express, que será imprescindible para el proyecto porque permite trabajar con el protocolo HTTP y tener un sistema de rutas.

Vue.js utiliza plantillas basadas en HTML lo que permite enlazar declarativamente el DOM con los datos de la instancia de *Vue.js*. Internamente compila las plantillas a funciones de renderizado del DOM virtual. En estas plantillas se pueden tener el código HTML, CSS y código JavaScript en un único archivo. A su vez también cuenta con los componentes *Vue.js*, que extienden elementos HTML para encapsular código reutilizable. Los componentes son elementos personalizados a los que el compilador de *Vue.js* asocia un comportamiento.

Figura 30: Logo de Vue.js

Con el fin de dar una experiencia fluida a los usuarios, la aplicación consistirá en una Single-Page Application (SPA) o aplicación de página única, la cual es una aplicación o sitio web que cabe en una sola página donde todos los códigos de HTML, JavaScript y CSS se cargan de una vez. Es decir, que todas las páginas con las que cuenta se pueden mostrar desde una sola, sin necesidad de recargar el navegador. Proporciona una experiencia de usuario agradable porque las transmisiones son muy rápidas y las comunicaciones entre cliente y servidor son fluidas.

Para asegurar la persistencia de los datos, se debe utilizar una base de datos que se comunique con el lado del servidor, de forma que los clientes puedan mantener sus datos y

acceder a ellos bajo demanda a través de las peticiones. La base de datos MongoDB almacena los datos como documentos, que suelen seguir un formato similar al JSON. Un documento se coloca dentro de una colección, que sería la equivalencia a una tabla en SQL. MongoDB cuenta con la capacidad de crear objetos dinámicos que se almacenan como documentos en la base de datos.

Figura 31: Logo de MongoDB

Mongoose es una biblioteca de JavaScript que le permite definir esquemas con datos fuertemente tipados. Cuando se crea un esquema, Mongoose le permite crear un modelo basado en un esquema especifico. Se puede hacer uso de esta biblioteca instalándolo en el proyecto mediante npm.

Para poder proporcionar conexión a esta base de datos a los clientes, por lo que se va a utilizar *mLab*. Se trata de una base de datos como servicio que permite utilizar la base de datos de MongoDB. Se ejecuta en los proveedores de la nube Amazon, Google y Microsoft Azure.

Figura 32: Logo de mLab

Se han escogido *Git* y *GitHub* tecnologías de control de versiones, ya que es la alternativa gratuita más popular, compitiendo con *Subversion* o *Bitbucket* . Utilizar un sistema de control de versiones aporta ciertas ventajas sobre el proyecto como la capacidad de ramificar según distintas *features* que se estén implementando, velocidad de acceso, seguridad y un flujo de trabajo adaptable.

Figura 33: Logos de Git y GitHub

Como se necesita que la aplicación se encuete ejecutándose en un dominio público, se ha elegido la plataforma de computación en la nube Heroku como medio que permite desplegar, ejecutar y administrar aplicaciones en node.js. Heroku lanza la aplicación cogiendo el código desde GitHub.

Figura 34: Logo de Heroku

5.2 Diseño

En este apartado se describe el proceso del diseño de los componentes de la aplicación (interfaz, actividades, etc.), incluyendo las técnicas utilizadas y los problemas encontrados.

5.2.1 De la interfaz

Para realizar el diseño de la interfaz de la aplicación se elaboró en primer lugar una lista de los elementos de interacción con los que constará la web y después se colocaron en un esquema por el orden por el que aparecerían.

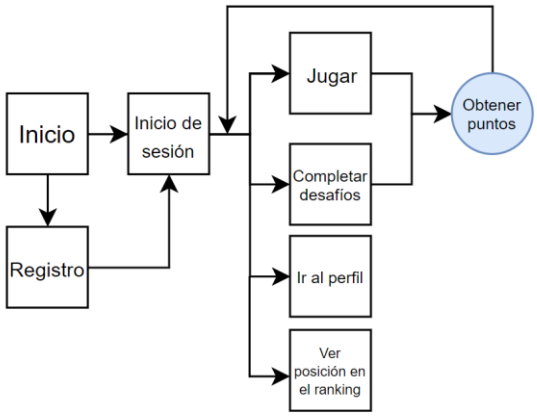


Figura 35: Esquema de las interacciones principales

Una vez pensadas las páginas, se hicieron bocetos a papel y mockups con el objetivo de facilitar su desarrollo. Se utilizaron unas plantillas simples con las proporciones aproximadas de una página web visualizada desde un escritorio para maquetar rápidamente las distintas ideas del diseño.

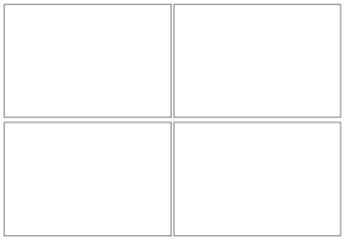


Figura 36: Ejemplo de plantilla de escritorio

Para mejora la experiencia del usuario, se han tenido en cuenta aspectos como la usabilidad, accesibilidad, contenido de calidad y facilidad de búsqueda. La usabilidad se mide en función de lo intuitiva y fácil que la web sea de manejar y además está relacionada con mejorar la satisfacción del usuario a través de su experiencia. Durante el proceso de diseño y el de desarrollo se han tenido en cuenta diversos factores para garantizar una buena

usabilidad, los cuales se exponen a continuación.

- **Diseño claro e intuitivo**: para ello se ha utilizado el framework de CSS Bootstrap. Esta tecnología permite crear interfaces web de forma simple con la gran ventaja de que se adaptan al tamaño del dispositivo en el que se visualiza. Los componentes de Bootstrap siguen unos estándares de diseño y de esa manera, el usuario encontrará una interfaz con la que puede estar familiarizado.

- **Evitar tiempos de carga lentos**: gracias a Vue.js se producen unos tiempos de carga casi instantáneos en la web gracias a que utiliza un renderizado en el navegador.

- **Congruencia:** para facilitar el uso de la web al usuario, se le debe proporcionar la información necesaria para que entienda lo que está sucediendo en cada momento. Para ello, cuando cambia el estado de su perfil aparecen notificaciones en la parte superior, completamente visibles para el usuario. Además, se le informa del estado de subida de la imagen cuando la adjunta en una solución.

Respecto a la accesibilidad, se ha desarrollado la web siguiendo las pautas WCAG que ya se conocían con anterioridad. Sin embargo, finalmente se ha utilizado la herramienta online WAVE para evaluar el estado del código. WAVE consiste en una web que determina la accesibilidad actual de otros sitios web, pero para este proyecto se utilizó la extensión de Google Chrome que realiza el test desde la misma aplicación.

5.2.2 Del contenido

Los elementos de interacción principal de la web son los desafíos y los juegos. Para crearlos y añadirlos a la web de la manera más sencilla se ha utilizado la plataforma de colaboración de contenido interactivo H5P (HTML5 Package). Esta herramienta permite crear contenido como videos interactivos, presentaciones, tests y demás contenidos mediante un editor versátil, por lo que permitió ahorrar mucho tiempo en la implementación. Pero en primer lugar se tuvieron que diseñar cómo serían estas actividades. Se diseñaron actividades y juegos de todas las categorías que permitiría la aplicación (Abstracción, descomposición, reconocimiento de patrones y algoritmos). Las ideas para estas actividades fueron extraídas de la investigación previa y de algunos recursos en páginas con propósitos similares.

5.3 Desarrollo

Como se ha mencionado anteriormente, la aplicación consistirá en una web SPA (Single Page Application), desarrollada sobre Node.js.

5.3.1 Configuración del espacio de trabajo

Para facilitar el desarrollo del código, se ha utilizado el editor de código fuente Visual Studio Code, el cual tiene un buen soporte para aplicaciones de Node.js. En primer lugar, se ha descargado el programa para Windows 10 desde su página oficial [12]. Y para configurarla correctamente se han realizado los siguientes pasos:

- Instalar Node.js en el sistema operativo de la página oficial [13]. Node Package Manager (npm) se incluye ya dentro de la distribución.
- Instalar Git en el sistema operativo para poder utilizar la bash de Linux. Tras instalarla, se debe seleccionar en la configuración de VSC la bash de Git como Shell por defecto. A partir de ahora, cada terminal que se añada será una bash. Así se podrá trabajar fácilmente con los comandos del sistema de ficheros, Git, npm, etc.

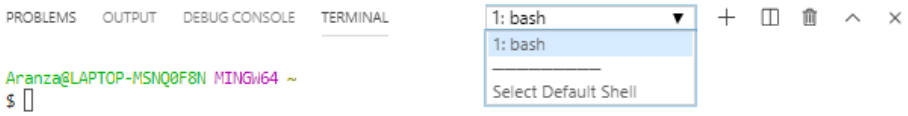


Figura 37: Bash en VSC

- Instalar Heroku CLI [14] en el sistema operativo, y necesita que Git esté instalado en el sistema. De esta forma se podrá gestionar la aplicación subida a Heroku desde la terminal.
- Se creó el repositorio correspondiente al proyecto en GitHub.
- Se habilitaron dentro de VSC una serie de extensiones para facilitar la escritura de código. Algunas de ellas son: *GitLens*, *Auto Rename tag*, *Auto close tag*, *Beautify*, *Color Highlight* y *heroku-cli*.

Se creó la base de datos para almacenar la información del proyecto en mLab [15]. Para ello era necesario crearse una cuenta en la página y elegir un cluster en la nube, que en este caso sería el gratuito que proporciona Amazon Web Services.

Además, se han instalado en el navegador Google Chrome las extensiones de *WAVE* para la evaluación de accesibilidad y *Lighthouse* para los reportes de la web. Y para visualizar la base de datos, se ha instalado la aplicación de escritorio MongoDB Compass Community.

5.3.2 Desarrollo Back-end

Vue.js cuenta con una herramienta de terminal para construir la estructura de las aplicaciones de manera casi inmediata llamada Vue-cli. Esta se instala mediante npm:

```
npm install -g @vue/cli
```

Para crear un proyecto:

```
vue create my-project
```

Despúes se podrán elegir las caracteristicas que tendrá el proyecto, de las cuales se han elegido las que están señaladas en la figura 38.

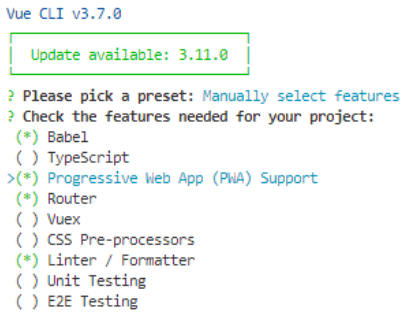


Figura 38: Generación de aplicación con vue-cli

Esta parte del desarrollo pertenece al Front-end, que se realizó posteriormente, pero de esta manera ya se tenía la base de los ficheros preparada para comenzar el Back-end. En primer lugar, se creó el servidor web desde un fichero llamado server.js, el cual contiene una serie de instrucciones que inicializan el servidor. El módulo express es un framework para node que permite generar aplicaciones rápidamente y aporta un sistema de rutas estable. La configuración básica para crear un servidor web es la siguiente:

```
const express = require('express')
const app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})

app.listen(3000)
```

Figura 39: Configuración básica de express.

Pero en adelante se añadirán middlewares al stack del objeto “app” para modificar la configuración del servidor según se necesiten.

Para continuar con el desarrollo del Back-end, se preparó el sistema de rutas para las peticiones del cliente al servidor. Se crearon distintos ficheros que serían modelos y otros que serían rutas. Primero se definieron los modelos los cuales consisten en esquemas de bases de datos o schemes que definen los atributos que tendrá cada colección que se cree. Estos son una de las muchas funcionalidades de Mongoose. Se ha decidido que la información que se almacene en la base de datos sea los datos de los usuarios, los desafíos creados, las soluciones a esos desafíos, los juegos y los logros. En cada schema se debe indicar el nombre del atributo y el tipo de dato que debe ser.

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

let Juego = new Schema({
    titulo: {
        type: String
    },
    categoria: {
        type: String
    },
    carpeta: {
        type: String
    }
}, {
    collection: 'juegos'
});

module.exports = mongoose.model('Juego', Juego);
```

Figura 40: Schema de la colección juegos

Express cuenta con una clase Router para crear manejadores de rutas. Por cada esquema se ha creado una instancia de Router para dividir las rutas y evitar errores. Por lo que cada colección tendrá su colección de rutas según las operaciones que se le quieran realizar. En este caso, se utilizan los métodos de direccionamiento Get y Post para la comunicación del cliente con el servidor y de este con la base de datos.

```
const express = require('express');
const juegosRoutes = express.Router();
let Juego = require('../models/juegos.model');

// Añadir juego nuevo (solo administrador)
juegosRoutes.route('/add').post(function (req, res) {···
});

// Obtener todos los juegos
juegosRoutes.route('/').get(function (req, res) {···
});

// Obtener un juego por id
juegosRoutes.route('/:id').get(function (req, res) {···
});

module.exports = juegosRoutes;
```

Figura 41: Rutas para la colección juegos

5.3.3 Desarrollo Front-end

Dentro del proyecto generado con Vue CLI, se instala un binario llamado vue-cli-service. El comando vue-cli-service serve inicia un servidor de desarrollo donde se ven de forma instantánea los cambios que se hagan en las componentes de vue. El comando vue-cli-service build produce un paquete listo para producción en el directorio /dist donde se minimizan los archivos JS, CSS y HTML.

Existe una componente por cada página de la web, y cada una incluye su código HTML

correspondiente, los estilos únicos que se le aplican y las funcionalidades que necesita. El código de cada componente se inyecta dentro de la componente App, por lo que en el fichero App.js es donde se ha añadido la barra de navegación para que esté presente en todas las páginas.

Respecto a las funcionalidades y las peticiones a la base de datos, existe un método propio de las componentes llamado created() que se ejecuta cada vez que se accede a la componente. El módulo axios de Node.js permite hacer peticiones HTTP desde un objeto en el que se especifican la dirección, el método (get o post) y los datos que se quieran enviar. En este caso, estas peticiones se comunicarán con las rutas que ya se han definido para cada colección de la base de datos.

```
created() {
  this.axios.get("/juegos").then(res => {
    this.todos = res.data;
  });
},
```

Figura 42: Método created para la componente Juegos

En la figura anterior, se está haciendo una petición a la ruta "/" de /juegos para mostrar todos los elementos de la colección "juegos".

5.3.4 Bases de datos

Para este proyecto se escogió MongoDB como sistema de base de datos y como base para la persistencia de los datos. La aplicación está diseñada para realizar las operaciones CRUD desde el código del Back-end, por lo que no es necesario instalar MongoDB en la consola. Si fue necesario buscar una solución en la nube para evitar que los datos estuvieran únicamente en la maquina local, por lo que se usó mLab. Cuando se crea una base de datos, mLab ofrece una URI que permite a mongoose conectarse con la base de datos en la nube.

```
48  mongoose.connect(config.DB, {
49      useNewUrlParser: true
50  }).then(
51      () => {
52          console.log('Database connected')
53      },
54      err => {
55          console.log('Can not connect to the database' + err)
56      }
57  );
```

Figura 43: Conectando con mlab mediante la URI

Y para poder visualizar los datos se usó la plataforma de escritorio MongoDB Compass Community conectándose con la misma URI, y un usuario y contraseña con los permisos suficientes.

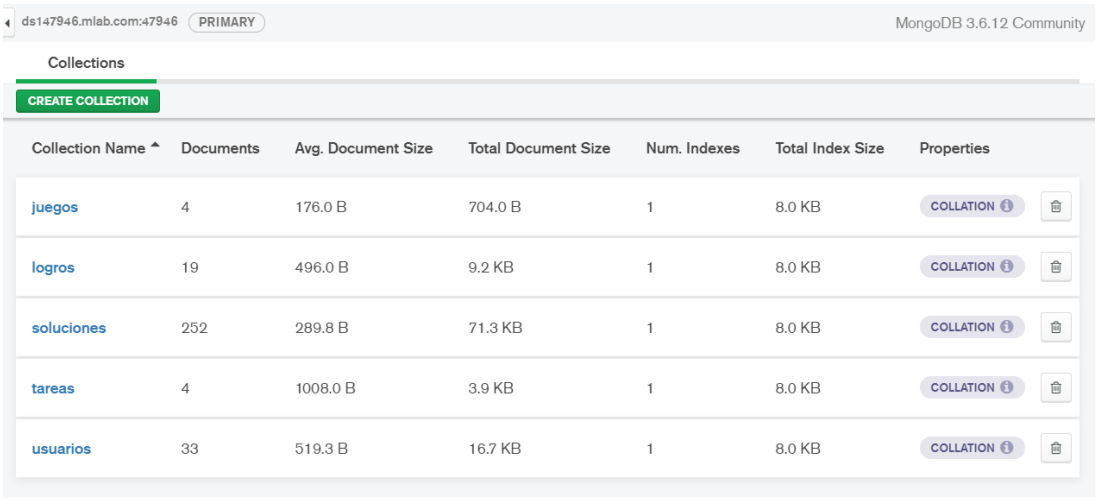


Figura 44: MongoDB Compass Community

5.3.5 Control de versiones

El trabajo se encuentra bajo el control de versiones, de manera que se puede acceder a todas las versiones del proyecto desde su inicio hasta su fin. Se ha utilizado git y github para ello, y a pesar de ser un proyecto individual, se crearon ramas para abordar el desarrollo de diversas características de la aplicación. La aplicación se encuentra disponible en el repositorio de GitHub: https://github.com/alu0100889680/gamiapp

5.3.6 Seguridad

Un certificado SSL garantiza la seguridad al encriptar la comunicación entre cliente y servidor. El servicio de despliegue Heroku ya proporciona un certificado SSL gratuito siempre que la dirección URL de la página tenga el sufijo "herokuapp.com". Pero una de las mayores dificultades fue la confusión de que en algunas ocasiones en el navegador la aplicación tenía certificado y en otras no, por lo que se consideraba un "sitio no seguro". El problema era que la página no redireccionaba de HTTP a HTTPS, por lo que si se accedía con HTTP el sitio era no seguro y con HTTPS si reconocía el certificado SSL. Tras una pequeña investigación, se encontró un módulo llamado *express-sslify* que lo solucionaba, redireccionando siempre de HTTP a HTTPS. Este módulo se instaló desde npm y se añadió a la configuración de app.

```
var enforce = require('express-sslify');

app.use(enforce.HTTPS({
    trustProtoHeader: true
}));
```

Figura 45: Configuración express-sslify

5.4 PWA

Las Progressive Web Apps son aplicaciones que pueden ejecutarse en una pestaña del navegador, pero también son instalables. Una PWA instalada se ve y se comporta como la demás aplicaciones nativas. Se inicia desde el mismo lugar donde se lanzan estas y se ejecuta en una ventana de aplicación, sin una barra de direcciones o ninguna apariencia. Son rápidas, confiables y funcionan en cualquier ordenador.

Vue-cli generó por defecto en la creación del proyecto, dos ficheros cruciales para configurar PWA. Estos ficheros son manifest.json y registerServiceWorker.js. El fichero manifest.json permite hacer la aplicación instalable y es donde se especifica el nombre de la aplicación cuando se instale, los iconos que se mostrarán, el punto de comienzo desde donde arrancará, la opción para mostrar u ocultar la barra del navegador, el color de fondo cuando la app se inicia y el color de tema. Por otro lado, el fichero registerServiceWorker.js consiste en una secuencia de comandos, que tu navegador ejecuta en segundo plano, separado de la web. Ya incorporan funciones como notificaciones push, funcionamiento offiline a través de contenido cacheable y sincronización en segundo plano. Como idea futura para la aplicación, se pretenden añadir las notificaciones y el modo offiline.

```
{
  "name": "Gami",
  "short_name": "Gami",
  "icons": [{
      "src": "/img/icons/icon4.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "/img/icons/icon3.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ],
  "start_url": "/",
  "display": "standalone",
  "background_color": "#000000",
  "theme_color": "#4DBA87"
}
```

Figura 46: manifest.json para Gami

5.5 Despliegue

El despliegue de la aplicación no ha sido final sino continuo. De esta manera se ha podido ir comprobando si las funcionalidades que se han añadido no tienen errores en la PWA ni en el navegador. Heroku ofrece despliegue para aplicaciones Node.js y una especie de tutorial para lograrlo. Existen dos métodos de despliegues disponibles, mediante Heroku CLI y mediante GitHub. Para este proyecto se ha usado la opción de GitHub por lo sencillo que resulta subir los cambios a la aplicación. Se debe conectar el repositorio con Heroku y para subirla se deberá hacer un push a la rama "heroku":

```
git push heroku master
```

Capítulo 6
Conclusiones y líneas futuras

Este capítulo recoge las conclusiones a las que se ha llegado con este trabajo y las líneas futuras de desarrollo, que son todos aquellos aspectos no desarrollados que podrían completarse.

6.1 Conclusiones

El objetivo de realizar este trabajo era ahondar en las aplicaciones de la gamificación en ámbitos tecnológicos, y mediante la investigación realizada y la implementación de un sistema de este estilo, se ha podido comprender el potencial que tienen. Las herramientas en educación nunca son insuficientes, y cuanto mejor adaptadas, el aprendizaje será más eficaz. Sin embargo, aunque la gamificación está diseñada para captar la atención de los alumnos, se debe medir su uso para no saturarlos.

Mediante la gamificación se pueden educar diversas materias, y puede ser aprovechada para introducir temas no muy presentes en la educación tradicional, como es el pensamiento computacional. Gracias a la investigación previa al trabajo, se ha podido comprender que el pensamiento computacional no consiste en técnicas de programación, sino representa un ingrediente vital del aprendizaje de la ciencia, la tecnología, la ingeniería y las matemáticas.

En la plataforma Gami, se han dividido el pensamiento computacional en sus cuatro fases con el propósito de presentar este concepto a los usuarios de una manera sencilla de entender. Y gracias a la dinámica de los juegos y desafíos presentes, adquieran un pensamiento más lógico.

Hoy en día, la web se ha convertido en algo indispensable en la vida de las personas que buscan concretamente lo que necesitan. La versatilidad de las tecnologías actuales de desarrollo web permiten suplir todo tipo de necesidades y en este caso, ha facilitado la creación de este sistema gamificado. Y se ha comprendido, que nunca se puede dejar de aprender en el mundo de la informática, y que hay que formarse según avanza la tecnología. Es por ello por lo que se ha querido profundizar en las aplicaciones web progresivas (PWA). A pesar de no ser una tecnología reciente, está de moda desde que Windows, Google Chrome, Firefox y Safari las soportan de forma nativa. Gracias a que ha configurado Gami como una PWA podemos obtener beneficios de aplicaciones nativas sin que sea desarrollada en los lenguajes de los sistemas operativos.

A pesar de tener distintos objetivos en este proyecto, no se han abandonado buenas prácticas como la de garantizar que la web sea responsiva, accesible y usable. Y se ha comprendido la importancia de proporcionar un certificado SSL para asegurar una transmisión segura entre servidor y usuario.

Finalmente, se van a comentar las líneas futura. Este proyecto se podría continuar desarrollando con facilidad, integrando más funcionalidades y mejorando muchos aspectos para adecuar su uso en entornos reales, ya sea en aulas, en casa, o por cuenta propia.

6.2 Líneas futuras

El desarrollo de este proyecto se ha seguido conforme a lo planeado, resolviendo la mayor parte de los objetivos propuestos. Sin embargo, durante el desarrollo han surgido nuevas ideas que resultarían beneficiosas para el proyecto. Algunas de ellas se han logrado incluir a tiempo y otras se han convertido en nuevos objetivos a cumplir para una etapa posterior. Las principales líneas futuras son:

- Aumentar la potencia de la PWA aprovechando sus características. Se podría ofrecer un modo offline cacheando las librerías necesarias y guardando sesiones de usuarios. Además, es posible añadir una función para manda notificaciones *Push* a los dispositivos.
- Proporcionar un método de subida de contenidos HTML de h5p para que los profesores puedan crear sus propios juegos y añadirlos a la plataforma de forma sencilla.
- Optimizar al máximo el rendimiento y adecuar el uso de las librerías incluidas dado que pueden impactar negativamente en la experiencia del usuario.
- Indexar la web en los buscadores.
- Se pretende agregar a la aplicación funciones como:
    - Permitir a un usuario editar o borrar su cuenta.
    - Creación dinámica de logros
    - Facilitar mecanismos que permitan crear una comunidad más conectada (chats, mensajes, comentarios, valoraciones)
    - Permitir que los usuarios cambien de periodo a uno superior según crecen.
    - Añadir algún tipo de recompensa para los profesores, para que se sientan motivados a participar en la plataforma.
- Integrar la aplicación en aulas reales para observar su utilización en un entorno real con usuarios reales.
- Atender en profundidad la accesibilidad y usabilidad para garantizar principios de un diseño adecuado, agradable y usable por personas de diferentes capacidades.

Capítulo 7
Conclusions

This chapters gathers the conclusions reached with this work and the future lines of development, which are all those aspects that have not been completed.

7.1 Conclusions

The objective of carrying out this work was to delve into the applications of gamification in technological fields, and through the research carried out and the implementation of a system of this kind, has been possible to understand the potential they have. The tools in education are never insufficient, and the better adapter, the learning will become more effective. However, although gamification is designed to capture the attention of students, its use should be measured to not saturate them.

Various subjects can be educated through gamification and can be used to introduce topics not very present in traditional education, such as computational thinking. Thanks to pre-work research, it has been understood that computational thinking does not consist of programming techniques, but represents a vital ingredient of learning science, technology, engineering and mathematics.

On this Gami platform, computational thinking has been divided into its four phases with the purpose of presenting this concept to users in a simple way of understanding. And thanks to the dynamics of the games and challenges present, acquire a more logical thinking.

Today, the web has become indispensable in the lives of people who are specifically looking for what they need. The versatility of current web development technologies allows to meet all kinds of needs and in this case, it has facilitated the creation of this gamified system. And it has been understood that you can never stop learning in the computer world, and that you must train as technology advances. That is why we wanted to deepen the progressive web applications (PWA). Despite not being a recent technology, it is fashionable since Windows, Google Chrome, Firefox and Safari support them natively. Thanks to the fact that *Gami* has been configured as a PWA, we can obtain benefits from native applications without being developed in the languages of every operating system.

Despite having different objectives in this project, good practices such as ensuring that the website is responsive, accessible and usable have not been abandoned. And the importance of providing an SSL certificate to ensure a secure transmission between server and user has been understood.

Finally, the future lines will be commented. This project could be easily developed, integrating more features and improving many aspects to adapt its use in real environments, whether in classrooms, at home, or on their own.

Capítulo 8
Presupuesto

Este capítulo recoge una propuesta de presupuesto que estima el coste del trabajo según el tiempo empleado en este proyecto.

Tareas	**Horas**	**Presupuesto**
Investigación de conceptos previa	15 horas	7€/hora
Análisis de requisitos	5 horas	7€/hora
Investigación de las tecnologías adecuadas	20 horas	7€/hora
Diseño de la aplicación	20 horas	20€/hora
Desarrollo del lado del servidor	60 horas	25€/hora
Desarrollo del lado del cliente	70 horas	25€/hora
Preparación para el despliegue a Heroku	20 horas	25€/hora
Configuración de la PWA	25 horas	25€/hora
Preparación test de usuarios	5 horas	7€/hora
TOTAL	240 horas	5090€
Mantenimiento	4 horas/mes	40€/hora
TOTAL	+4 horas mensuales	+160€ mensuales

Tabla 1: Modelo de presupuesto

Servicio	**Descripción del plan**	**Horas**	**Presupuesto**
Hosting *mLab*	El plan gratuito Sandbox de *mLab* proporciona una base de datos con un almacenamiento de 0.5GB en un servidor compartido que se ejecuta en una máquina virtual compartida	Indefinido	Sin costo
Dominio *Heroku*	*Heroku* ofrece 550 horas de alojamiento gratuito al mes. Si la web no se usa en 30 minutos se queda inactiva y si un usuario accede a ella gasta a 1 hora.	550 horas al mes	Sin costo
Certificado SSL	*Heroku* ofrece certificados SSL a todas las aplicaciones gratuitas bajo el dominio *.herokuapp.com	Indefinido	Sin costo

Tabla 2: Modelo de presupuesto de servicios

Bibliografía

[1] G. Zichermann and C. Cunningham, *Gamification by design*, 2011. [Online]. Disponible en: https://pdfs.semanticscholar.org/dda6/737d6ea16d1ee4ee8de87ffc4fe1fd16df8b.pdf?_ga=2.126205453.1490286080.1568090041-555388941.1568090041

[2] K. M. Kapp, "The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education", Pfeiffer, may. 2012. [Online]. Disponible en: http://scholar.google.com/scholar_url?url=https://www.igi-global.com/viewtitle.aspx%3Ftitleid%3D74836%26isxn%3D9781466612266&hl=es&sa=X&scisig=AAGBfm2fnCFULelq8VIZCQPmHpvCEk80Ow&nossl=1&oi=scholarr

[3] S. Deterding, M. Sicart, L. Nacke, K. O'Hara and D. Dixon, "Gamification: Toward a Definition", may. 2011. [Online]. Disponible en: http://gamification-research.org/wp-content/uploads/2011/04/02-Deterding-Khaled-Nacke-Dixon.pdf

[4] J.M. Wing, "Computational thinking", *Communications of the ACM*, vol. 49, no. 3, mar. 2006. [Online]. Disponible en: https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdfkl

[5] F. J. Sánchez i Peris, "Gamificación", ", *Education in The Knowledge Society* (EKS) vol. 16, no. 2, Jun. 2015. [Online]. Disponible en: http://gamisolution.es/wp-content/uploads/2016/06/gamificaci%C3%B3n.pdf

[6] J. D. Cruzado y Y. Rodríguez, "El potencial de la gamificación aplicado al ámbito educativo", Dep. de Psicología Social, Univ. de Sevilla. [Online]. Disponible en: https://fcce.us.es/sites/default/files/docencia/EL%20POTENCIAL%20DE%20LA%20GAMIFICACI%C3%93N%20APLICADO%20AL%20%C3%81MBITO%20EDUCATIVO_0.pdf

[7] V. Gaitán, "Gamificación: el aprendizaje divertido", *Educativa. [Online]. Disponible en* https://www.educativa.com/blog-articulos/gamificacion-el-aprendizaje-divertido/

[8] M. R. González, "Codigoalfabetización y pensamiento computacional en educación primaria y secundaria: validación de un instrumento y evaluación de programas", tesis doctoral, Univ. Nacional de Educación a Distancia, 2016. [Online]. Disponible en: http://e-

spacio.uned.es/fez/eserv/tesisuned:Educacion-Mroman/ROMAN_GONZALEZ_Marcos_Tesis.pdf

[9] W. Bender, C. Urrea y M. Zapata-Ros, "Presentación", *Revista de Educación a Distancia* vol. 46, pp. 3, Sep. 2015. [Online]. Disponible en: https://www.um.es/ead/red/46/presentacion.pdf

[10] A-M. Ortiz-Colón, J. Jordán y M. Agredal, "Gamificación en educación: una panorámica sobre el estado de la cuestión", *Educação e Pesquisa* vol. 44, 2018. [Online]. Disponible en http://www.scielo.br/pdf/ep/v44/1517-9702-ep-44-e173773.pdf

[11] F. L. Largo, F. J. García-Peñalvo, X. M. Prieto y E. V. Vidal, "La enseñanza de la informática, la programación y el pensamiento computacional en los estudios preuniversitarios", *Education in The Knowledge Society* (EKS) vol. 18, no. 2, 2017. [Online]. Disponible en https://rua.ua.es/dspace/bitstream/10045/68598/1/2017_Faraon-Llorens_etal_EKS.pdf

[12] Microsoft, "Visual Studio Code", 2019. [Online]. Disponible en: https://code.visualstudio.com/

[13] Node.js Developers Joyent, "Node.js. [Online]. Disponible en: https://nodejs.org/es/.

[14] Heroku Dev Center, "Node.js". [Online]. Disponible en: https://devcenter.heroku.com/categories/nodejs-support.

[15] MongoDB Inc, "MongoDB Hosting: Database-as-a-Service by mLab" [Online]. Disponible en https://mlab.com/.

[16] P. LePage, "Tu primera Progressive Web App", *2019.* [Online]. Disponible en: https://developers.google.com/web/fundamentals/codelabs/your-first-pwapp/?hl=es.

[17] "H5P", 2019. [Online]. Disponible en https://h5p.org/.

[18] MongoDB Inc, "MongoDB", 2019. [Online]. Disponible en: https://www.mongodb.com/es

[19] MongoDB Inc, "MongoDB Docs". [Online]. Disponible en: https://docs.mongodb.com/

[20] Nike, Inc., "Nike Run Club for Iphone & Android, 2019. [Online]. Disponible en "https://www.nike.com/us/en_us/c/nike-plus/running-app-gps

[21] Starbucks, “Starbucks Rewards program: Starbucks Coffee Company”, 2019. [Online]. Disponible en: https://www.starbucks.com/rewards/

[22] ClassDojo, Inc., “ClassDojo”, 2019. [Online]. Disponible en: https://www.classdojo.com/

[23] Kahoot, “Kahoot!”, 2019. [Online]. Disponible en: https://kahoot.com/

[24] Fitbit Inc., “Fitbit Official Site for Activity Trackers & More”, 2019. [Online]. Disponible en: https://www.fitbit.com/home?utm_source=&utm_medium=paidsearch&gclid=EAIaIQobChMI9qyJisvF5AIVCRgMCh27cw78EAAYASAAEgKsk_D_BwE&gclsrc=aw.ds

[25] Habitica, “Habitica, Gamify your life”, 2019. [Online]. Disponible en: https://habitica.com/static/home

[26] Scratch, “Scatch”, 2019. [Online]. Disponible en: https://scratch.mit.edu/