CIENCIA DE DATOS Y ANALÍTICAS DE APRENDIZAJE APLICADAS AL VIDEOJUEGO ARTICODING PARA LA EVALUACIÓN DE CONCEPTOS BÁSICOS EN ALUMNOS DE SECUNDARIA

DATA SCIENCE AND LEARNING ANALYTICS APPLIED TO THE VIDEO GAME ARTICODING FOR THE EVALUATION OF BASIC CONCEPTS IN SECONDARY SCHOOL STUDENTS

TRABAJO FIN DE GRADO

CURSO 2022-2023

AUTORES

DAVID GAVILANES DE DIOS

ÁNGEL HORTELANO PÉREZ

PABLO RABADÁN ARZA

DIRECTORES

BALTASAR FERNÁNDEZ MANJÓN

ANTONIO CALVO MORATA

GRADO EN INGENIERÍA INFORMÁTICA

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

CIENCIA DE DATOS Y ANALÍTICAS DE APRENDIZAJE APLICADAS AL VIDEOJUEGO ARTICODING PARA LA EVALUACIÓN DE CONCEPTOS BÁSICOS EN ALUMNOS DE SECUNDARIA

DATA SCIENCE AND LEARNING ANALYTICS APPLIED TO THE VIDEO GAME ARTICODING FOR THE EVALUATION OF BASIC CONCEPTS IN SECONDARY SCHOOL STUDENTS

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTORES
DAVID GAVILANES DE DIOS
ÁNGEL HORTELANO PÉREZ
PABLO RABADÁN ARZA

DIRECTORES
BALTASAR FERNÁNDEZ MANJÓN
ANTONIO CALVO MORATA

CONVOCATORIA: JUNIO 2023

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

29 DE MAYO DE 2023

DEDICATORIA

A mis compañeros de proyecto,

por realizar juntos este arduo trabajo conmigo.

A mi familia y pareja, por aguantar las

innumerables quejas mientras trabajaba en el proyecto.

Y a mí mismo, por conseguir algo impensable

por el David de hace cuatro años.

David Gavilanes de Dios

A todas aquellas personas que enseñan con pasión,

porque no estaría en este punto del camino sin ellos.

A mis amigos y a mi madre,

por haberme ayudado en los momentos más difíciles.

Y a mis compañeros, por unir fuerzas en este proyecto,

que, además, ayudará a gente a descubrir su pasión por la informática.

Pablo Rabadán Arza

A mis compañeros de TFG por hacer posible la realización del proyecto.

A mi familia y amigos por todos los buenos ratos que me hacen pasar.

A mis abuelos por cuidarme desde que soy pequeño y en especial a mi abuelo José Antonio Pérez Bustos por enseñarme a utilizar un ordenador cuando apenas podía subirme a la silla.

A la UCM por aportarme buenos momentos y formarme.

Ángel Hortelano Pérez

AGRADECIMIENTOS

En esta sección queremos mostrar nuestro más profundo agradecimiento a aquellas personas que han permitido y ayudado a que este trabajo haya salido adelante.

Con este motivo, damos gracias a los dos profesores que evaluaron nuestra plataforma web y que nos proporcionaron una valiosa retroalimentación para poder mejorarla.

Muchas gracias a los autores de los anteriores proyectos de Articoding y a los alumnos que lo probaron en diversas sesiones, ya que sin ellos no sería posible disponer del juego y de trazas de este libremente.

Por último, pero no menos importante, deseamos expresar nuestro sincero agradecimiento a nuestros tutores, Baltasar Fernández Manjón y Antonio Calvo Morata, por su invaluable apoyo y guía en este proyecto, permitiéndonos así el logro de nuestros objetivos.

RESUMEN

Ciencia de datos y analíticas de aprendizaje aplicadas al videojuego Articoding para la evaluación de conceptos básicos en alumnos de secundaria

Este Trabajo de Fin de Grado se centra en el desarrollo de una herramienta para evaluar el conocimiento adquirido por los jugadores en juegos serios de programación. El proyecto aborda específicamente los desafíos que enfrentan los profesores al monitorear el juego de los estudiantes y evaluar el conocimiento adquirido durante el juego.

Con este fin se utiliza Articoding, un juego gratuito y de código abierto que cumple con los estándares de recopilación de datos. Mediante el uso de xAPI para la recopilación de datos, el proyecto tiene como objetivo proporcionar a los profesores una plataforma con análisis y visualización de datos detalladas para facilitar una evaluación efectiva. El trabajo incluyó jugar a Articoding, identificar métricas de análisis útiles, desarrollar una plataforma web para la visualización de datos y realizar evaluaciones de usuarios.

Gracias a este enfoque, se pudo garantizar una mejora continua y la alineación con los requisitos de los usuarios. Los resultados contribuyen a mejorar la educación y las prácticas de evaluación de la programación en las escuelas primarias y secundarias.

Palabras clave

Articoding, xAPI, juegos serios, análisis de datos, plataforma web, evaluación.

ABSTRACT

Data science and learning analytics applied to the video game Articoding for the evaluation of basic concepts in secondary school students

This Bachelor's thesis focuses on the development of a tool to assess the knowledge acquired by players in serious programming games. The project specifically addresses the challenges faced by teachers in monitoring student gameplay and evaluating the knowledge acquired during the game.

To this end, Articoding is used, a free and open-source game that complies with data collection standards. By using xAPI for data collection, the project aims to provide teachers a platform with detailed analysis and visualization of data to facilitate effective evaluation. The work involved playing Articoding, identifying useful analysis metrics, developing a web platform for data visualization, and conducting user assessments.

Thanks to this approach, continuous improvement and alignment with user requirements could be ensured. The results contribute to improving programming education and assessment practices in primary and secondary schools.

Keywords

Articoding, xAPI, serious games, data analysis, web platform, evaluation.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1 - Introducción

1.1 Motivación

La programación se ha convertido en una habilidad cada vez más relevante en la sociedad actual. Con el avance de la tecnología, la capacidad de programar se ha vuelto esencial en diversos ámbitos, desde la automatización y simplificación de tareas hasta el desarrollo de software y soluciones innovadoras. Por esta razón, la enseñanza de la programación en etapas tempranas de la educación, como primaria y secundaria, ha cobrado una importancia creciente.

Además, no hay que olvidar que permite desarrollar habilidades cognitivas y de pensamiento crítico en los estudiantes. A través de la programación, se fomenta el pensamiento lógico y algorítmico de los alumnos, ya que pueden aprender a descomponer problemas complejos en pasos más pequeños. Asimismo, la programación promueve la creatividad, la competencia intelectual, la resolución de problemas, la colaboración y la toma de decisiones, habilidades clave en la sociedad digital actual [1].

En este contexto, los videojuegos han surgido como una herramienta educativa efectiva para enseñar programación. Los llamados "juegos serios" utilizan la dinámica de los videojuegos para motivar a los estudiantes y proporcionarles un entorno interactivo y lúdico en el cual aprender. Estos juegos permiten a los alumnos adquirir conocimientos de programación mientras se divierten y participan activamente en el proceso de aprendizaje.

Sin embargo, los profesores que desean utilizar videojuegos como herramientas educativas se enfrentan a desafíos que pueden desincentivar su uso. Uno de ellos es la dificultad de saber qué ocurre mientras los alumnos juegan. Los profesores pueden tener dificultades para monitorear el progreso individual de cada estudiante que el juego transfiere. La falta de herramientas de monitoreo adecuadas para evaluar el conocimiento obtenido durante el juego dificulta la aplicación de la herramienta, así como la retroalimentación y la evaluación precisa del aprendizaje de los estudiantes por parte del profesor.

1.2 Objetivos

El presente trabajo tiene como objetivo principal proporcionar plataformas efectivas para evaluar el conocimiento adquirido por los jugadores al participar en juegos serios centrados en la programación. Para lograr este objetivo, se abordará el estudio y desarrollo de plataformas de análisis y visualización de datos utilizando como caso de estudio el juego Articoding, un juego de software libre que usa estándares para la recogida de datos.

Los subobjetivos específicos que se persiguen son los siguientes:

1. Conocer Articoding y su contexto de uso: El primer subobjetivo se centra en adquirir un conocimiento profundo del juego serio Articoding, un trabajo realizado por estudiantes de otro año [2] y [3] con el objetivo de enseñar conceptos básicos de programación. Esta tarea se centra en comprender su contexto de uso en la enseñanza de la programación. Se investigará la mecánica y funcionamiento del juego, los conceptos de programación que se abordan y los objetivos de aprendizaje que se persiguen con su uso. Este conocimiento permitirá comprender las características del juego y el proceso que sigue para abordar su objetivo educativo.
2. Aprender xAPI: El segundo subobjetivo se enfoca en el aprendizaje y dominio del estándar xAPI (Experience API). xAPI es un protocolo que permite recopilar datos sobre las interacciones de los jugadores en los juegos y almacenarlos en un formato estructurado. Mediante el estudio de este, se podrá comprender cómo se recogen y estructuran los datos de Articoding, ya que utiliza este estándar. Esto permitirá su posterior análisis y evaluación de los jugadores.
3. Desarrollar herramientas de análisis y visualización de datos: El tercer subobjetivo implica el desarrollo de herramientas de análisis y visualización de datos basadas en los datos de interacción proporcionados por Articoding y su implementación de xAPI. Estas herramientas permitirán examinar de manera detallada las acciones y decisiones tomadas por los jugadores durante el juego. Se explorarán técnicas de análisis de datos,

como las usadas en Dr. Scratch [4], y se utilizarán visualizaciones efectivas para extraer información relevante para los docentes. Estas herramientas servirán para evaluar el conocimiento adquirido por los jugadores de forma más precisa y eficiente.

1.3 Plan de trabajo

El plan de trabajo para el desarrollo de este proyecto se dividió en varias etapas, abordando cada uno de los subobjetivos planteados anteriormente. Primero, comenzamos por jugar a Articoding para familiarizarnos con sus mecánicas. Durante esta etapa identificamos la información relevante para un profesor al evaluar el conocimiento de los estudiantes.

En la segunda etapa del plan de trabajo nos centramos en adquirir un conocimiento más profundo sobre el estándar xAPI, familiarizándonos con los principios y conceptos clave de xAPI, como las declaraciones, los actores, los verbos y los objetos. Además, analizamos la información generada por el juego en este formato y su representación en formato JSON. Asimismo, llevamos a cabo un estudio exhaustivo de plataformas que se utilizan para evaluar el aprendizaje en juegos serios de programación. Esta investigación nos permitió comprender mejor las características y funcionalidades que dichas plataformas ofrecen, así como identificar las buenas prácticas y los aspectos y desafíos a tener en cuenta en el desarrollo de nuestra propia plataforma.

Como tercera etapa, recopilamos datos de interacción de usuarios con el juego y nos centramos en crear una plataforma web donde los profesores pudieran acceder y visualizar los análisis generados. Esta plataforma permitiría a los profesores visualizar las distintas sesiones de juego realizadas con sus alumnos y acceder a los análisis, gráficas y tablas específicas con la información relevante de cada sesión. El objetivo era proporcionar una interfaz intuitiva y fácil de usar que brindase a los profesores una visión clara del progreso y desempeño de sus estudiantes.

Después de completar la implementación de la plataforma web y como última etapa, nos centramos en realizar evaluaciones con usuarios. Para ello, invitamos a dos profesores de universidad, que habían utilizado ya con anterioridad el juego Articoding,

a probar la plataforma desarrollada. Durante estas evaluaciones, los profesores utilizaron la plataforma y nos brindaron valiosos comentarios y sugerencias para mejorarla. Analizamos detenidamente sus comentarios y decidimos implementar los cambios y detalles que consideramos más necesarios y beneficiosos para el funcionamiento y la experiencia del usuario.

El plan de trabajo y cada una de sus etapas se abordaron de manera iterativa. A medida que avanzábamos en cada etapa, revisábamos y mejorábamos el trabajo realizado hasta el momento para asegurarnos de que la plataforma cumpliera con los requisitos establecidos. Este enfoque iterativo nos permitió realizar ajustes continuos y mejorar la funcionalidad y usabilidad de la plataforma.

Capítulo 2 - Estado de la cuestión

Hoy en día, uno de los principales métodos de enseñanza de programación a alumnos de corta edad son los juegos de programación, dado que de forma lúdica se pueden crear y fortalecer conocimientos que a priori parecen complicados.

Se ha realizado una revisión exhaustiva de diversos juegos de programación [5] [6], muchos de los cuales no son plataformas de uso libre y, en su mayoría, ya no están disponibles para jugar. Algunos ejemplos de estos juegos mencionados en las revisiones son Unnamed RPG, Gidget o RoboBuilder. Además, la ausencia de este tipo de juegos de uso libre se debe, en la mayoría de los casos, a que no son lanzados públicamente o que son resultado de trabajos académicos que a veces están incompletos o tienen errores. Debido a esto, es muy importante hacer plataformas libres que permitan a la gente que quiere aprender programación poder jugarlos teniendo libre acceso a los mismos. Como consecuencia de ello, se parte del juego Articoding debido a que cumple con estas características y porque nos permitían tener acceso a datos de uso reales recogidos en años anteriores por otros jugadores para poder ver qué análisis se podrían ir creando.

Asimismo, dentro de este ámbito destacan Scratch y Dr. Scratch [4], siendo de las herramientas más usadas hoy en día. Scratch es un lenguaje de programación visual que fue especialmente diseñado para que tanto niños como programadores principiantes aprendiesen a programar gracias a una metodología divertida y entretenida. Además, Dr. Scratch es una herramienta en línea que analiza proyectos realizados con Scratch y proporciona retroalimentación sobre la calidad y eficiencia del código tras cargar los proyectos en su página web.

2.1 Articoding

Articoding [2] [3] es un juego serio que tiene como finalidad acercar conceptos de programación básicos a personas que no han programado anteriormente. Este proceso se lleva a cabo mediante la programación visual, lo que les permite programar mediante una interfaz visual en la que se deben seleccionar y configurar bloques de código gráficos, los cuales representan los diferentes conceptos de programación,

como variables, funciones o bucles. Esto hace que la programación sea más atractiva y accesible para los usuarios. El jugador debe resolver los distintos niveles encontrando la solución a un problema que se le presenta en un escenario en forma de tablero. Este sistema de juego es muy similar al utilizado por Scratch.

El principal objetivo del juego consiste en conseguir que un láser apunte a una esfera y, para ello, el jugador debe mover los distintos objetos distribuidos por el tablero usando bloques de código gráficos denominados "tarjetas" que simulan elementos de programación básicos. Estos mismos son introducidos de forma incremental y progresiva en los diferentes niveles que componen el juego, resultando en un aprendizaje absorbente que invita a seguir formándose en cada nuevo problema a resolver. Con la finalidad de motivar aún más a los jugadores, el argumento de Articoding trata sobre un pingüino científico llamado Albert que ha sido encerrado en un laboratorio y que debe escapar del mismo usando sus conocimientos de programación.

Este juego está dirigido a adolescentes y jóvenes con pocos o ningún conocimiento sobre programación que quieran aprender de forma fácil e interactiva los elementos básicos de la misma. Como consecuencia, un buen contexto de uso sería en una clase de programación en la que el profesor, además de explicar los conceptos necesarios, quiera que sus alumnos experimenten y aprendan de una forma diferente bajo su supervisión.

Los niveles están agrupados por temas, de manera que cada conjunto introduce diferentes conceptos básicos de programación progresivamente. A medida que el jugador avanza en el juego, los temas introducidos se vuelven más complejos que los anteriores, incluyendo conceptos cómo variables, bucles o funciones. En total, se han organizado siete grupos de niveles que se explicarán en los siguientes apartados. En general, cada grupo consta de siete niveles, excepto la categoría de variables que tiene diez niveles y el grupo de funciones que tiene tres. En total, el juego cuenta con cuarenta y ocho niveles. Además, para avanzar es necesario completar cada nivel antes de comenzar el siguiente. De esta manera, se deben superar los 48 niveles para llegar al final del juego. Los niveles están diseñados con una dificultad progresiva que permite analizar el progreso de los alumnos y en qué conceptos se quedan atascados para poder reforzar su aprendizaje.

En todo momento, tanto durante la ejecución de los niveles como en el menú inicial, los jugadores pueden acceder a la guía, la cual muestra la información detallada sobre cada agrupación de niveles una vez hayan sido completados. Esto les permite repasar rápidamente cualquier concepto anterior que no recuerden con claridad o no hayan comprendido bien. Además, si se hace clic y se arrastra el cursor en el tablero de un nivel, es posible cambiar la perspectiva de la vista de una vista cenital en 2D a una vista en 3D. Esto ayuda a visualizar mejor los componentes, conocer su disposición en el tablero y comprender qué se debe hacer para solucionar el problema.

Tras completar cada nivel, se muestra una pantalla que muestra las estrellas obtenidas, siendo este el elemento que define la habilidad con la que un nivel se ha completado. El máximo de estrellas que se pueden obtener por nivel es tres y cada una de ellas se basa en distintos parámetros, como en función de si se han cumplido los pasos mínimos necesarios, si se ha utilizado un bloque especial y la limpieza general del código. Si se cumplen estos tres criterios se conceden las tres estrellas, las cuales se pueden intentar conseguir en su totalidad, habiendo 144 en todo el juego.

Por último, la aplicación también ofrece un modo de juego que permite a los jugadores aplicar sus conocimientos adquiridos de una forma más creativa, usando el modo de creación de niveles. En él, los jugadores deben diseñar sus propios tableros, arrastrando componentes a la posición del tablero que deseen y seleccionando las celdas de este para modificar sus dimensiones y forma. Asimismo, también tienen que aportar una posible solución del nivel creado, lo cual provoca que tengan que usar los conocimientos de programación adquiridos, tanto para crear un problema, como para poder resolverlo, pudiendo profundizar así en sus nuevas competencias.

Además, se les pasa un cuestionario a los jugadores antes y después de jugar a Articoding. Gracias a los cuestionarios podemos ver qué conceptos han aprendido, si les ha gustado y cómo se podría mejorar la experiencia de usuario.

2.1.1 Niveles de introducción

Los primeros niveles son de introducción (figura 2-1) y en ellos se explica cómo funciona el juego y las tarjetas más básicas: *Gira* y *Mueve Láser* (figura 2-2).

Figura 2-1: Tablero del nivel 1 de introducción

Figura 2-2: Ejemplos de tarjetas Gira y Mueve Láser

En la mitad izquierda de la pantalla se sitúan las tarjetas y en la derecha se sitúa el tablero. Una vez que el jugador termina de formar su código con estas tarjetas, lo ejecuta y los elementos del tablero se mueven conforme a lo descrito en las tarjetas. El nivel se considera superado si tras ejecutar todos los pasos el láser termina enfocando a la esfera. Si no se consigue, debe repetirse para poder pasar al siguiente nivel.

2.1.2 Niveles de variables

En el siguiente bloque de niveles aparecen las variables (figura 2-3). Éstas nos permiten almacenar valores enteros que se repiten a lo largo del nivel.

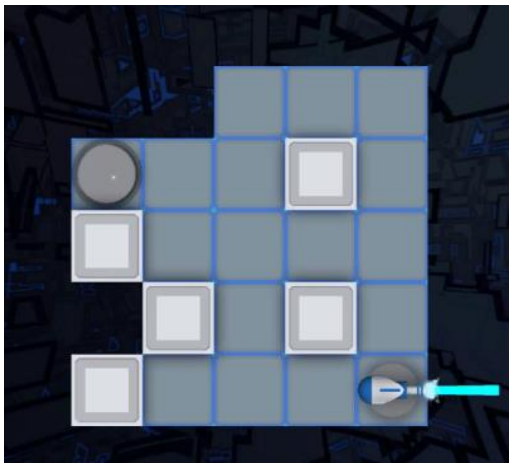

Figura 2-3: Tablero del nivel 8 de variables

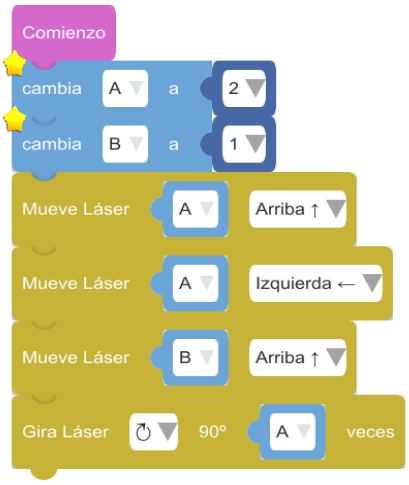

Figura 2-4: Ejemplos de tarjetas Cambia

Para almacenar las variables se utiliza la tarjeta *Cambia* (figura 2-4), que asigna un valor a las variables que podemos crear. Esto permite al jugador utilizar los mismos valores en diferentes secciones del programa y le ayuda a familiarizarse con el uso de variables. Se puede cambiar el valor de una variable el número de veces que necesite el usuario, sustituyendo siempre el nuevo valor al asignado con anterioridad.

2.1.3 Niveles de tipos de datos

Después de completar los niveles de variables llegan los que explican los diversos tipos de datos (figura 2-5), es decir, qué tipo de información se almacena una variable. En este bloque nos exponen los valores booleanos y cómo asignar objetos a variables.

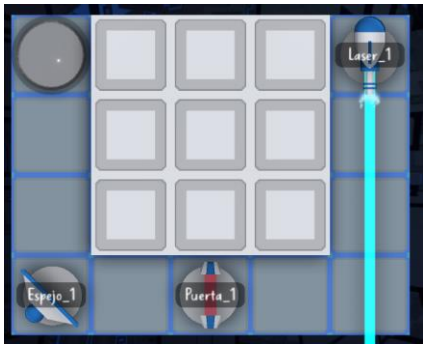

Figura 2-5: Tablero del nivel 6 de tipos de datos

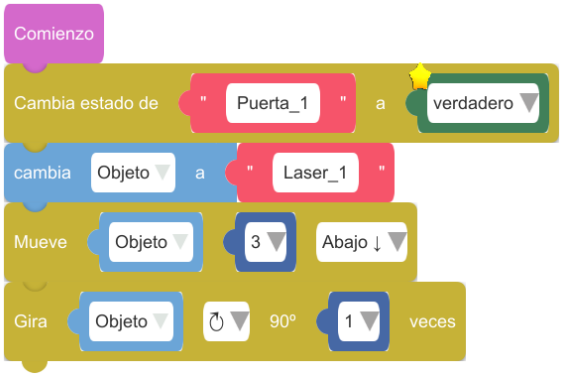

Figura 2-6: Ejemplos de tarjetas con valores booleanos

Las tarjetas de color verde representan condicionales y en este primer bloque simplemente aparecen los valores booleanos *true* y *false*, los cuales son utilizados para abrir y cerrar puertas que bloquean el paso de los láseres. Las tarjetas rojas sirven para introducir los nombres de los objetos del tablero y poder asignarlos a variables, lo cual nos permite no sólo mover y girar el láser, si no que podemos aplicar esas acciones a otros elementos, como espejos. Otro uso que pueden tener es el de literales en las tarjetas de acción gracias a la tarjeta *Cambia* (figura 2-6).

2.1.4 Niveles de operadores básicos

En estos niveles (figura 2-7), se enseña a los usuarios que en la programación podemos realizar operaciones entre variables, como sumas entre dos variables para

obtener un nuevo valor y almacenar el resultado en una tercera variable. Asimismo, hasta ahora las únicas opciones que tenían para los booleanos eran verdadero o falso, pero a partir de estos niveles aparecen las primeras condiciones, las cuales permiten ejecutar diferentes secciones de código si una evaluación lógica es verdadera o falsa.

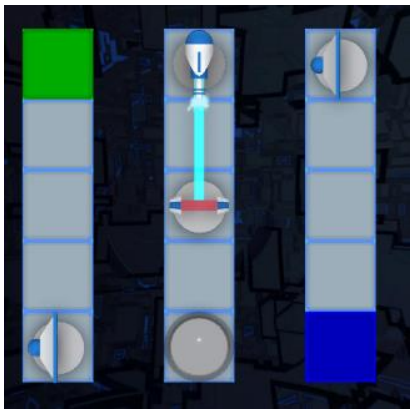

Figura 2-7: Tablero del nivel 5 de condicionales

Figura 2-8: Ejemplos de tarjetas usando operadores básicos

Figura 2-9: Ejemplo de tarjeta que permite hacer operaciones aritméticas

Figura 2-10: Ejemplo de tarjeta 'no'

En este ejemplo (figura 2-8) podemos comprobar cómo gracias a la tarjeta *¿Todas las celdas color ocupadas?*, la *puerta_1* se abre cuando un objeto (*espejo_1*) se coloca en la casilla verde. La tarjeta azul oscura (figura 2-9) nos permite realizar operaciones aritméticas básicas sobre variables o valores. Además, el resultado se puede meter en acciones o asignarlo a variables. Por último, aparece la tarjeta *no* (figura 2-10), la cual se pone delante de las condiciones e invierte lo devuelto por ellas, es decir, si la condición es verdadera se cambia a falsa y viceversa.

2.1.5 Niveles de bucles

En este bloque de niveles (figura 2-11) se introducen los bucles, lo cual les permite a los jugadores poder repetir un conjunto de instrucciones un número concreto de veces. En Articoding existen dos tipos de bucles: uno similar a un *for* y otro a un *while*.

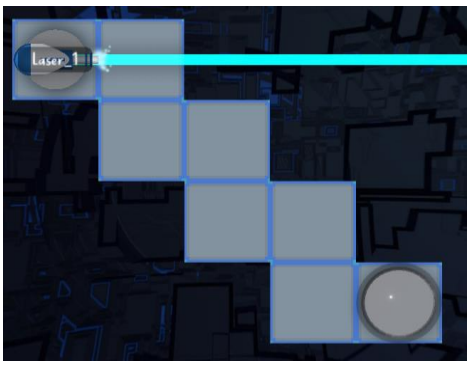

Figura 2-11: Tablero del nivel 2 de bucles

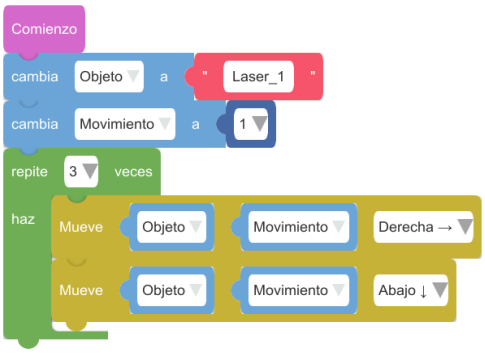

Figura 2-12: Ejemplo de tarjeta de bucle tipo for

Figura 2-13: Ejemplo de tarjeta de bucle tipo while

La tarjeta *Repite x veces* (figura 2-12) realiza lo contenido en el bloque haz "x" veces, como si se tratase de un bucle *for*, es decir que ejecuta de forma secuencial las tarjetas contenidas en el bucle un número concreto de veces. En la tarjeta *Mientras que* (figura 2-13) hay que especificar una condición de las aprendidas en el bloque anterior y se realizan las acciones especificadas en el bloque *haz* mientras se cumpla la condición, igual que un bucle *while*.

2.1.6 Niveles de condicionales

Las sentencias *if* e *if-else* aparecen en estos niveles (figura 2-14). También se introduce una nueva tarjeta que simula el comportamiento de los operadores *and* (&&) y *or* (||) lógicos.

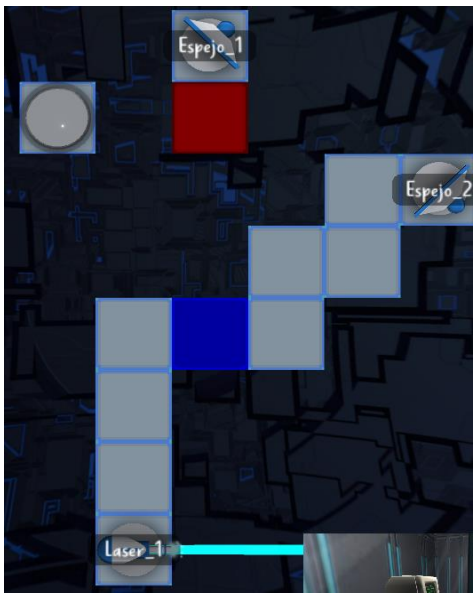

Figura 2-14: Tablero del nivel 9 de condicionales

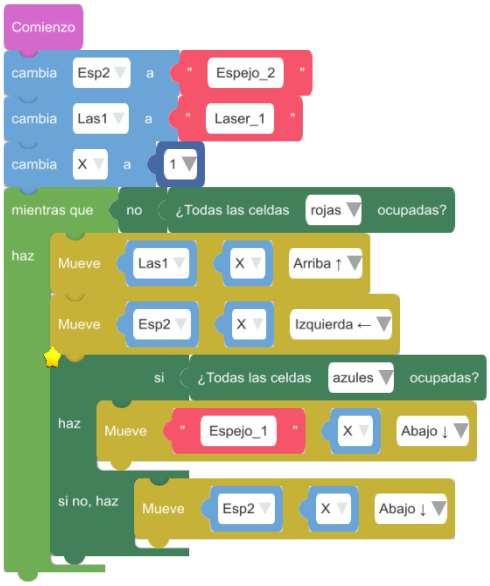

Figura 2-15: Ejemplo de tarjetas usando condicionales

Figura 2-16: Ejemplo de uso del operador &&

La tarjeta *Si…haz…si no, haz…* (figura 2-15) representa un bloque *if-else* donde si se cumple la condición, realiza el primer bloque y si no, el segundo. La tarjeta *y/ó* (figura 2-16) funciona como los operadores *&&* y *||*, permitiendo evaluar dos condiciones y se puede encajar en las tarjetas que las evalúan (*while, if, if-else, no*).

2.1.7 Niveles de funciones

En este bloque de tres niveles (figura 2-17) se hace una pequeña introducción a las funciones, sin parámetros ni valores de retorno.

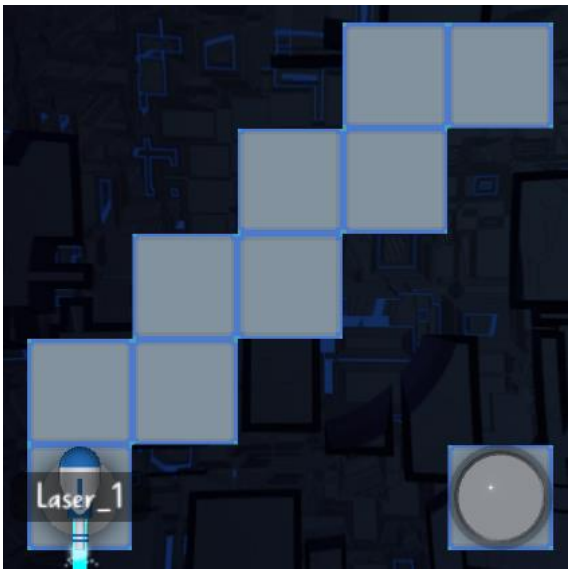

Figura 2-17: Tablero del nivel 1 de funciones

Figura 2-18: Ejemplo de tarjetas de funciones

Las funciones se representan mediante una tarjeta de color morado (figura 2-18). Estas tarjetas se colocan aparte de la línea que sigue la ejecución, pero pueden ser llamadas desde cualquier punto.

2.2 xAPI

En la actualidad, el acceso a los datos se ha vuelto cada vez más sencillo, lo que nos brinda la oportunidad de obtener información valiosa sobre los usuarios, comprendiendo así sus comportamientos, hábitos y necesidades. Una herramienta que facilita este proceso es xAPI (Experience API), un estándar educativo ampliamente utilizado en la estructura de datos para rastrear y compartir el aprendizaje entre diferentes sistemas de aprendizaje [7].

En este contexto, es relevante mencionar los MOOCs (Cursos Masivos Abiertos en Línea) y los LMS (Sistemas de Gestión del Aprendizaje), los cuales proporcionan herramientas de análisis e información sobre los estudiantes [8]. Dentro de estos entornos, xAPI se utiliza para rastrear las actividades que realizan los usuarios en tiempo real. Para ello, xAPI utiliza declaraciones que constan de un actor, un verbo y un objeto. Estas

declaraciones siguen la sintaxis JSON (JavaScript Object Notation) y permiten el intercambio de información entre los diferentes sistemas educativos. Un ejemplo sería:

```
{
  "actor": {
    "name": "Sally Glider",
    "mbox": "mailto:sally@example.com"
  },
  "verb": {
    "id": "http://adlnet.gov/expapi/verbs/experienced",
    "display": { "en-US": "experienced" }
  },
  "object": {
    "id": "http://example.com/activities/solo-hang-gliding",
    "definition": {
      "name": { "en-US": "Solo Hang Gliding" }
    }
  }
}
```

Figura 2-19: Ejemplo de estructura JSON

Como se puede observar en la figura 2-19, se sigue la sintaxis JSON para la definición de la declaración. Se usa principalmente debido a que facilita el paso de información de un LRS a otro. Un LRS (Learning Record Store) es un contenedor de datos donde se almacenan las distintas acciones que recoge xAPI sobre las decisiones del usuario. Necesitamos este tipo de repositorio de datos para que xAPI funcione de forma más fluida [9]. Un ejemplo representativo de todo este proceso es el que se muestra a continuación en la figura 2-20:

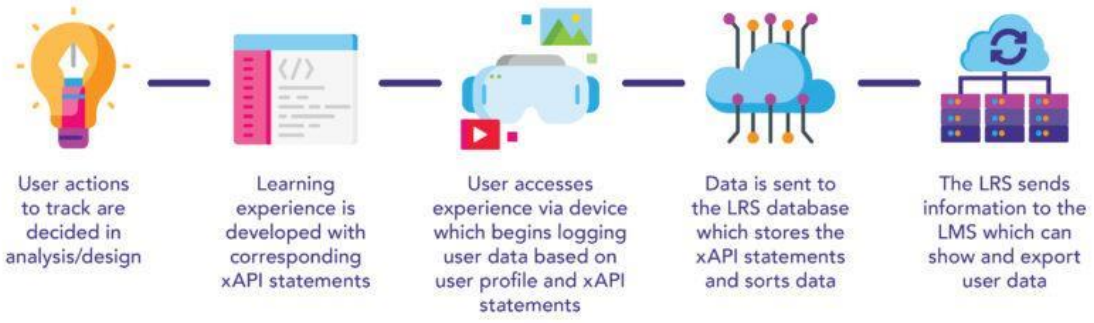

Figura 2-20: Flujo del seguimiento de información en xAPI [9]

A continuación, se explicarán las declaraciones xAPI y sus diferentes tipos siguiendo la información proporcionada en [10], el cual es un libro de referencia ampliamente utilizado en el campo de xAPI.

2.2.1 Declaración

Para la definición de una declaración son necesarios tres requisitos:

- Cada propiedad (actor, verbo, objeto) no puede ser usada más de una vez.
- Es necesario que haya, al menos, un actor, un verbo y un objeto.
- La declaración puede incluir las propiedades en cualquier orden.

Además, una propiedad que es importante utilizar es un ID para la declaración. Esta propiedad debería ser establecida por el LRP (Learning Record Provider, un cliente de xAPI que le manda los datos al LSR) y, en su defecto, el LSR debe encargarse de ello. Esto se debe a que, si no se hace, no habrá manera en un futuro de volver a encontrar la declaración. Asimismo, es aconsejable que sea el programa de origen o LRP quien lo fije, debido a que si otros programas necesitan usar la declaración y el ID ha sido implantado por el LSR, solamente el propio LSR lo conoce, por lo que se convierte en un proceso complicado el usar la determinada declaración.

2.2.2 Actor

El actor es una persona, grupo de personas (identificadas individualmente o no) o un sistema que ha hecho algo. Adicionalmente, un actor solo puede ser uno de los siguientes objectTypes: un Agente (un individuo o un sistema) o un Grupo (más de un individuo). Un Agente debe usar uno de los cuatro tipos de identificadores funcionales inversos, que son maneras únicas de identificar a una persona, grupo o sistema:

- mbox: representa una dirección email que es única para esa persona, tanto hasta este momento desde el pasado como en un futuro. Debe seguir el formato "mailto:email address".
- mbox_sha1sum: simboliza también una dirección email que está cifrada gracias a un algoritmo hash.

- openID: usado por diversos sitios conocidos, como Google o Yahoo, es una especificación para una única y anónima identificación a través de numerosos sitios web.
- account: representa una cuenta en una intranet, LMS (Learning Management System) u otro sistema que permita identificar al agente.

En algunos casos, como cuando se usa un LMS o un "launcher", la cuenta usada para entrar será el identificador. Además, un mbox será parte de la información de la cuenta y, si se utiliza solo, el atributo "name" no aparecerá en el JSON.

Un agente tiene tres propiedades disponibles: objectType, name e identificador funcional inverso, el cual es el único y obligatorio. Para obtener el nombre se usará el que fue usado en la cuenta del usuario. Otro modo de obtenerlo es crear una ventana donde el usuario escribirá su nombre junto a su email en tiempo de ejecución. El objectType solo será necesario ser definido cuando el agente sea usado como un objeto en la declaración.

Asimismo, cada agente sólo debe contener un identificador único, por lo que, si ya cuenta con una dirección email, no debemos usar una cuenta y si usamos una cuenta, el agente no debe incluir un email para definir el mbox. Por último, los individuos no deben usar identificadores que también son usados por grupos.

Cuando el objectType se define como Grupo, el actor es un grupo de agentes y puede ser usado en prácticamente las mismas situaciones que un individuo. Hay dos tipos principales de grupos: grupos anónimos y grupos identificados.

Un grupo anónimo es definido en la especificación como un clúster de personas donde no hay un identificador preparado para el mismo. Para solucionar esto, se usa el objectType Grupo y se añade una lista de los identificadores únicos de cada miembro del grupo en la propiedad "member", por lo que son dos propiedades obligatorias. Aparte de estas dos, hay otra que es opcional, que es el nombre del grupo ("name").

No obstante, un grupo identificado tiene un identificador único que es usado para representar el propio grupo. Tiene cuatro propiedades asociadas: objectType, que es Grupo; name (opcional); member (opcional) y un identificador único. Además, como

tenemos un identificador único del grupo, no es necesario que tengamos los identificadores únicos de los individuos que lo conforman.

2.2.3 Verbo

Tal y como muestra la especificación, los verbos exponen la acción que tiene lugar entre el actor y la actividad. No hay una especificación sobre los verbos, pero un registro ADL identifica verbos y sus definiciones para que se pueda ser consistente en su uso. Asimismo, disponen de dos propiedades: ID y display. La primera es obligatoria y es única para cada verbo. La segunda se trata de una breve descripción del verbo, lo que facilita entender las acciones a personas que no se conocen los IDs de memoria.

Además, el ID de un verbo suele ser llamado el IRI (Internationalized Resource Identifier) del verbo, el cual simplemente es una URL que muestra la definición del verbo, la propiedad display y esta última en otras lenguas.

Algunos verbos en la especificación de xAPI cmi5 son "launched", que indica que la LMS ha lanzado la AU (Assignable Unit); "initialized", el cual muestra que la AU está preparada para la interacción (inicio del juego, de un nivel, reintento o inicio del modo creación); "completed", que significa que el usuario ha realizado todo el material necesario en la AU (el jugador ha completado todos los niveles o alguno de ellos); "passed", que señala cuando el usuario ha intentado y completado con éxito una determinada actividad de la AU y "failed", el cual indica cuando el usuario ha intentado y completado sin éxito una determinada actividad de la AU. Otros verbos de la especificación son "abandoned", "waived", "terminated" y "satisfied", todos ellos referidos a la finalización de una actividad de distintas maneras posibles (actividad abandonada, actividad omitida, sesión terminada y criterios satisfechos, respectivamente).

2.2.4 Objecto

Los objetos pueden ser prácticamente cualquier cosa, incluyendo, como se ha dicho anteriormente, un agente, un grupo, una actividad... Asimismo, cuando el objectType es una actividad, hay una serie de propiedades que son necesarias. Entre ellas se encuentra el identificador único (IRI), que se tiene que referir siempre a la misma

actividad, ya que, tal y como ya se ha explicado, un LRS no es capaz de tratar con dos actividades que tengan el mismo ID. Además, existe la propiedad "moreInfo" que proporciona más información a través de un enlace a otro documento. Por último, también se puede añadir opcionalmente una definición a cada actividad.

Algunos ejemplos de objetos usados en declaraciones son:

- Agente o grupo: Laura entrevistó a Víctor.
- Subdeclaración: Laura ha respondido a que Víctor ha acabado la aplicación.
- Declaración: Víctor ha acabado la aplicación.

2.2.5 Resultado

El objeto Resultado nos permite mostrar un resultado medido referido a la declaración de la que proviene. Una manera de realizar esto es a través de una puntuación, la cual sería un porcentaje obtenido a través de un decimal entre -1 y 1. También se puede seguir una puntuación ascendente con el campo "raw". Otra propiedad de este objeto es "success", que nos permite conocer si un determinado intento de la actividad ha tenido éxito o no. Aparte de éstas, está "completion", que indica si la actividad se completó o no; "response", que almacena la respuesta a una pregunta y "duration", que muestra el periodo de tiempo durante el cual tuvo lugar la declaración. Por último, se puede añadir un campo "extensions" que permite que las aplicaciones definan sus propios resultados personalizados.

2.2.6 Contexto

El objeto Contexto posibilita que información contextual sea añadida a la declaración. Dicho contexto añade más información referida a cómo la declaración se relaciona con otras actividades. Una propiedad de este objeto es "instructor", la cual enlaza una declaración con un instructor o grupo de instructores. También existe la propiedad "team", que enlaza la declaración con un equipo que no sea el actor. Como añadido a estas, hay varias propiedades más: "contexactivities", que relaciona la declaración a otras actividades gracias al contexto; "grouping", la cual es una relación indirecta con la actividad; "category", que permite categorizar la declaración según la

actividad y "other", donde se añade la demás información que no encaja en las otras categorías.

2.2.7 Otros campos

Otros campos que se usan en xAPI y que también son importantes son:

- "Timestamp" y "stored", que permiten conocer cuando se produjo la experiencia y cuando se agregó la misma al LRS de xAPI.
- "Version", que indica la versión de xAPI que siguen las declaraciones.
- "Voided", el cual muestra si la declaración ya no es información válida.
- "Signed", que posibilita incluir una firma digital para proveer de autenticidad e integridad a la declaración.

Por lo tanto, para dar por finalizadas las explicaciones de los campos que se usan en xAPI, podemos ver un pequeño resumen en la figura 2-21, donde se muestran varios de los más usados, estando sombreados en color gris aquellos que son imprescindibles, tal y como se expuso anteriormente en las propiedades de las declaraciones.

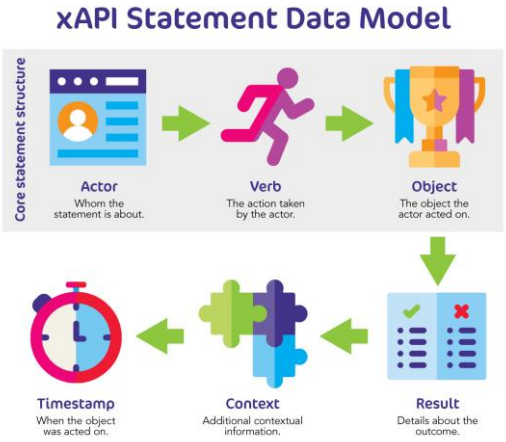

Figura 2-21: Estructura de las propiedades xAPI [9]

2.2.8 xAPI vs SCORM

Antes de la aparición de xAPI, el modelo estándar que se usaba en el sector del e-learning era SCORM, puesto que ofrecía lo necesario a las plataformas de aprendizaje online para que realizasen sus proyectos. No obstante, tiene dos problemas principales, que son la baja compatibilidad con otras aplicaciones o softwares y su desempeño

respecto a la capacidad de rastreo de la actividad de los usuarios. Todo esto provoca que para que funcione y sea posible realizar un control de la experiencia del usuario sea imprescindible el uso de otras plataformas concretas, como los LMS [8].

Estos problemas quedan solucionados gracias a xAPI, el cual permite realizar un seguimiento muy avanzado de las actividades de los usuarios en distintos tipos de aplicaciones y softwares, tanto en videojuegos como en LMS. Asimismo, para su rastreo hace uso de las eficientes tecnologías Cloud, alojándolo en la nube, desde donde se conecta con el programa en concreto.

Aunque xAPI está marcando una tendencia en el mundo del e-learning debido a que abre las puertas a cursos más flexibles, personalizados y enfocados en el usuario, sigue siendo bastante distinto a SCORM, ya que este último está pensado para utilizarse en cursos empaquetados donde los cuestionarios o formularios son la metodología principal de evaluación. Además, SCORM describe la estructura y el paquete del curso y xAPI no. Por último, cabe mencionar que cada estándar del e-learning se construye sobre las funcionalidades del anterior [11].

2.3 xAPI en Articoding

El juego Articoding también usa xAPI como estándar para representar la información que recoge de las interacciones que los usuarios realizan al jugar al videojuego. Un ejemplo de traza donde podemos ver los campos que se usan es el siguiente (figura 2-22):

```
{
    "actor": {
        "name": "fybu",
        "account": {
            "homePage": "https://simva-api.simva.e-ucm.es",
            "username": "fybu"
        }
    },
    "result": {
        "score": {
            "raw": 3
        },
        "extensions": {
            "minimum_steps": true,
            "no_hints": true,
            "first_execution": true,
            "steps": 5
        },
        "success": true
    },
    "verb": {
        "id": "http://adlnet.gov/expapi/verbs/completed"
    },
    "_id": "60bf4b3e1f7061006ecb50b9",
    "object": {
        "definition": {
            "type": "https://w3id.org/xapi/seriousgames/activity-types/level"
        },
        "id": "https://simva-api.simva.e-ucm.es/activities/60bf4b3e1f7061006ecb50b9/types_6"
    },
    "timestamp": "2021-06-09T19:32:57.369Z"
}
```


Figura 2-22: Ejemplo 1 de xAPI en Articoding

Para el actor, vemos que se usan los campos "name" y "account" (su identificador único), y dentro de este último encontramos la "homePage" y el "username".

Dentro del campo Resultado encontramos el "score", que tiene el valor "raw" 3, lo que nos quiere expresar que se ha completado el nivel con tres estrellas. Además, vemos que se usa "extensions", que con sus cuatro propiedades adicionales nos da un resultado más personalizado mostrando los pasos que ha requerido la acción ("steps"), si son los pasos mínimos ("mínimum_steps"), si ha sido la primera ejecución ("first_execution") y si no ha usado pistas ("no_hints"). Por último, tiene la propiedad "success", que indica si la acción ha sido realizada con éxito (en este caso si ha completado el nivel).

Respecto al verbo, tenemos la propiedad ID (IRI), que incluye la URL a una página web donde se muestra información sobre el verbo usado ("completed"). Cómo últimos campos tenemos "_id", que representa el identificador de la declaración; objeto, que tiene como propiedades "definition", que muestra el tipo del objeto, e ID y "timestamp", que guarda el instante en el que se llevó a cabo la actividad.

Podemos deducir que este ejemplo se trataba del jugador "fybu" completando un nivel, el sexto de la categoría de Types, gracias a toda esta información, como el ID del objeto o del verbo.

Otro ejemplo de traza que incluye información distinta de Articoding es la siguiente (figura 2-23):


```
{
    "actor": {
        "name": "fybu",
        "account": {
            "homePage": "https://simva-api.simva.e-ucm.es",
            "username": "fybu"
        }
    },
    "result": {
        "extensions": {
            "block_type": "variables_get",
            "code": "\r\n<block type=\"variables_get\" id=\"variables_get_brub\" x=\"30\" \
                    y=\"-280\">\r\n <field name=\"VAR\" id=\"tT8T=N7x38oJ/Ep)@%fP\" \
                    variableType=\"\">pasos</field>\r\n</block>",
            "level": "types_6",
            "action": "create"
        }
    },
    "verb": {
        "id": "http://adlnet.gov/expapi/verbs/interacted"
    },
    "_id": "60bf4b3e1f7061006ecb50b9",
    "object": {
        "definition": {
            "type": "https://w3id.org/xapi/seriousgames/activity-types/game-object"
        },
        "id": "https://simva-api.simva.e-ucm.es/activities/60bf4b3e1f7061006ecb50b9/variables_get_brub"
    },
    "timestamp": "2021-06-09T19:32:45.707Z"
}
```


Figura 2-23: Ejemplo 2 de xAPI en Articoding

En ella, se puede ver información parecida a la de antes, exceptuando las nuevas propiedades del campo Resultado, ya que ahora dentro de la propiedad "extensions" usa "block_type" para mostrar el tipo de bloque, "code", que muestra el estado del código, "level", el cual representa el nivel del juego y "action" para designar la acción realizada. Dada esta nueva información en la traza se puede saber que el mismo jugador ha creado una tarjeta de variables en el sexto nivel de Types.

Las diferencias entre ambas imágenes se deben a que ambas trazas son de tipos y sobre acciones diferentes. Esto da lugar a numerosos tipos de trazas que existen en Articoding, las cuales están recogidas en la tabla del Apéndice A.

2.4 xAPI en otros videojuegos y plataformas

Como consecuencia de que xAPI fuese desarrollado teniendo en cuenta que la mayoría de las personas usan dispositivos móviles que tienen acceso a Internet, numerosos videojuegos y plataformas hacen uso de esta tecnología para brindar a sus actividades de aprendizaje y a su sistema de informes más interoperabilidad y permitir que esos mismos usuarios puedan disponer de ellos. Por ejemplo, en un simulador de conducción puede generar eventos según se pulse el freno, se alcance determinada velocidad o se infrinjan normas viales.

Las LXP (Learning Experience Platforms) son plataformas que han surgido como resultado de la creación de xAPI. Estas plataformas tienen como objetivo mejorar el compromiso y la productividad de los usuarios al ofrecerles una experiencia de aprendizaje más personalizada, inmersiva e interactiva, adaptada a sus necesidades, gustos y objetivos. A diferencia de las LMS (Learning Management Systems), las cuales son gestionadas por la empresa y determinan los contenidos y las rutas de aprendizaje, las LXP permiten a los usuarios tener mayor autonomía y control sobre su propio proceso de aprendizaje [12]. Además, las LXP brindan recursos a las empresas para la formación y educación de sus colaboradores, puesto que administran sus recursos, capital y desempeño humano, mejorando sustancialmente los resultados de las mismas. Un ejemplo concreto de LXP es DL [13].

Respecto a los videojuegos, aquellos que se usan para rastrear los usuarios son denominados también juegos serios, debido a que su principal función no es divertir, sino enseñar. Estos mismos están dirigidos a una gran variedad de público con el objetivo de mejorar el aprendizaje y la motivación respecto al ámbito del propio juego, como pueden ser operaciones militares, empresariales y/o sanitarias o mejorar habilidades tanto blandas como técnicas [14].

Debido a que el ámbito de aplicación de los juegos serios es muy extenso, la clasificación también lo es, incluyendo tipos como los advergaming, aquellos que publicitan una marca o producto; los edutainment, los cuales enseñan mediante el uso de recursos lúdicos; los newsgames, que informan sobre eventos recientes; o los quizás más conocidos simuladores, cuyo aprendizaje se centra en imitar situaciones reales o

hipotéticas para enseñar habilidades o conocimientos. Algunos ejemplos de juegos serios concretos de los tipos mencionados son El Jardín del Vino, Real Lives, Democracy II y FlightGear, respectivamente [14].

Otros ejemplos de videojuegos que usan xAPI mencionados en [14] son:

- Darfur is Dying, un juego de simulación política que expone las condiciones en la propia región del Sudán, país del norte de África, gracias a la experiencia de los personajes en los campos de refugiados.
- Electroplankton, un juego edutainment con el que se puede aprender a crear melodías a través de la combinación de imágenes en movimiento mediante una pantalla táctil.

2.5 XML en Articoding

Los registros de trazas en Articoding incluyen la etiqueta "code" en diferentes eventos, y el significado de esta etiqueta varía según el verbo y el tipo de objeto. Cuando el verbo es "interacted" y el objeto es "level", la etiqueta "code" contiene información individual de bloques creados o modificados, mientras que cuando el verbo es "progressed" y el objeto es "level" la etiqueta de código contendrá la información completa de la ejecución de un intento/nivel por parte del usuario. Estos casos se pueden comprobar en la tabla del Apéndice A.

En Articoding, el código se almacena en un campo de texto en forma de string utilizando un formato XML. Esto se debe a que Articoding utiliza el intérprete de Blockly llamado Ublockly, que utiliza XML como formato de representación de los bloques de programación.

Articoding ha adoptado el entorno Blockly debido a que es un lenguaje de programación visual basado en bloques, desarrollado por Google. Blockly está diseñado para ofrecer un entorno intuitivo y accesible para principiantes de la programación. Además, permite conectar bloques entre sí para construir programas de manera modular, los cuales pueden ser traducidos a diferentes lenguajes de programación.

En el contexto de Articoding, Ublockly [15] es un intérprete específico de Blockly, el cual utiliza XML como formato de representación de los datos debido a varias razones. En primer lugar, XML es un formato reconocido para estructurar y procesar datos. Proporciona una sintaxis clara y legible mediante el uso de etiquetas. Además, XML es compatible con diversos sistemas, lo que permite el uso de una amplia gama de herramientas para analizar y comprender tanto los bloques individuales como el flujo de ejecución del código a través de su estructura anidada.

La utilización de XML facilita la transferencia de información entre diferentes programas y lenguajes, Sus etiquetas y estructuras son fácilmente procesadas por cualquier sistema, brindando flexibilidad en el intercambio de información.

Articoding, al tratarse de un juego serio dedicado a la enseñanza de la programación, el propio movimiento y las acciones que realiza el jugador se pueden traducir como conceptos básicos de programación. Un ejemplo básico sería asociar el movimiento de los objetos del juego al uso de variables ya que, al mover 3 pasos el láser equivale a usar el comando "*mover_laser*" con la variable "*cantidad_pasos*" la cual tiene un valor 3.

Con lo cual podemos ver que, a la hora de crear un sistema que analice las acciones del jugador, es vital saber analizar código básico y la calidad de este. Como las acciones que realiza el jugador se guardan en un string en formato XML, hemos tenido que comprender el funcionamiento del mismo para sacar el máximo provecho a los análisis realizados.

2.5.1 Estructura básica

```xml
<xml>
  <variables>
    <variable type="" id="4Y1jFgEnfT+#?Z*jc9N_">Cantidad</variable>
  </variables>
  <block type="start_start" id="start_start_ly3x" x="189" y="-235">
    <next>
      <block type="variables_set" id="variables_set_5355">
        <field name="VAR" id="4Y1jFgEnfT+#?Z*jc9N_" variableType="">Cantidad</field>
        <value name="VALUE">
          <block type="math_number" id="math_number_mhxk">
            <field name="NUM">2</field>
          </block>
        </value>
        <next>
          <block type="movement_move_laser" id="movement_move_laser_ln1q">
            <field name="DIRECTION">UP</field>
            <value name="AMOUNT">
              <block type="variables_get" id="variables_get_4wqx">
                <field name="VAR" id="4Y1jFgEnfT+#?Z*jc9N_" variableType="">Cantidad</field>
              </block>
            </value>
            <next>
              <block type="movement_move_laser" id="movement_move_laser_zlvk">
                <field name="DIRECTION">RIGHT</field>
                <value name="AMOUNT">
                  <block type="variables_get" id="variables_get_4286">
                    <field name="VAR" id="4Y1jFgEnfT+#?Z*jc9N_" variableType="">Cantidad</field>
                  </block>
                </value>
                <next>
                  <block type="movement_move_laser" id="movement_move_laser_ckbr">
                    <field name="DIRECTION">DOWN</field>
                    <value name="AMOUNT">
                      <block type="variables_get" id="variables_get_yt5m">
                        <field name="VAR" id="4Y1jFgEnfT+#?Z*jc9N_" variableType="">Cantidad</field>
                      </block>
                    </value>
                  </block>
                </next>
              </block>
            </next>
          </block>
        </next>
      </block>
    </next>
  </block>
</xml>
```

Figura 2-24: Ejemplo de código XML de Articoding

Como se puede observar en la figura 2-24, el documento inicia con la etiqueta <xml>, la cual va a contener toda la información de las acciones realizadas por un jugador durante un intento. Cada elemento tiene una etiqueta de inicio y de cierre.

Dentro del elemento raíz XML se encuentran contenidos dos elementos:

- <variables>: Elemento que guarda la información básica de todas las variables definidas, como su nombre e identificador.
- <block>: Elemento que representa una tarjeta de código y, que contiene así mismo la etiqueta <next> que denota el final del bloque y el inicio de

la siguiente tarjeta utilizada. Este formato representa la lectura secuencial de las tarjetas.

2.5.2 Variables

Dentro del objeto <variables>, encontramos la etiqueta <variable> que contiene información acerca del tipo, nombre e identificador de una variable determinada. Un ejemplo es la figura 2-25:

```
<variable type="" id="4Y1jFgEnfT+#?Z*jc9N_">Cantidad</variable>
```

Figura 2-25: Ejemplo de etiqueta <variable>

En el objeto <variables> habrá un número de elementos de tipo <variable> equivalente al número de variables definidas en el código.

2.5.3 Bloques de código

Las etiquetas de tipo <block> varían mucho dependiendo de su tipo, ya que cada uno representa una tarjeta de código del juego.

Aunque en las variables todas son de un mismo nivel, si se entiende como un árbol, siendo <variables> el padre y cada etiqueta <variable> los hijos en la estructura del XML.

En XML los bloques representan las posiciones ordenadas de las tarjetas y el primer bloque en el contenedor que incluye el resto de los bloques, mientras que el segundo contiene todos los bloques menos el primero. Este proceso se repetirá de forma recursiva hasta que el bloque final no contenga a ninguno, dando a entender que el código se ha terminado.

Figura 2-26: Ejemplo de estructura XML

Como podemos comprobar en la figura 2-26, para saber cuándo un bloque termina y otro empieza se usa la etiqueta <next>, la cual no tiene atributos. Con esto podemos decir que el último bloque ejecutado no tendrá esta etiqueta.

2.5.4 Tipos de bloques

Para poder analizar el XML hay que conocer los diferentes tipos de bloques que hay en el juego, ya que nos permiten entender su estructura y analizarla sintácticamente. A continuación, se explicará el funcionamiento de los bloques encontrados en los XMLs del juego:

- El bloque ***start_start*** marca el inicio del código, el primer bloque tendrá este tipo.
- El bloque ***variables_set*** indica que se ha asignado el valor a una variable existente. Este bloque tiene la siguiente estructura:

```
<block type="variables_set" id="variables_set_5355">
        <field name="VAR" id="" variableType="">
            Cantidad
        </field>
        <value name="VALUE">
                <block type="math_number" id="">
                    <field name="NUM">2</field>
                </block>
        </value>
//Siguientes bloques
</block>
```

Figura 2-27: Ejemplo de bloque tipo "variables_set"

Como podemos comprobar en la figura 2-27, el bloque contiene dos etiquetas:

- <field> que contiene el nombre, el id y el tipo de la variable modificada.
- <value> el cual tiene el atributo name que designa el tipo de valor que se va a asignar. Esta etiqueta además contiene un bloque con el valor que se ha cambiado.

math_number contiene un valor número. Este bloque se introduce en otros bloques, como en los de *variables_set* o *movement_move_laser*.

movement_move_laser indica que se va a mover un láser, el cual tiene la siguiente estructura (figura 2-28):

```
<block type="movement_move_laser" id="">
                                <field name="DIRECTION">RIGHT</field>
                                <value name="AMOUNT">
                                    <block type="variables_get">
                                        <field …>Cantidad</field>
                                    </block>
                                </value>
```

Figura 2-28: Ejemplo de bloque tipo "movement_move_laser"

- <field> contiene la dirección a la que se moverá el láser, en este caso, se desplazará a la derecha.
- <value> contiene un bloque con el número de pasos que se moverá el láser, como podemos observar en este ejemplo se ha usado un variable en lugar de un bloque *math_number*, los dos casos son posibles.

variables_get bloque que designa cuando se ha obtenido el valor de una variable ya declarada para ser usada por otro bloque. En el caso anterior vemos que se usa para mover el láser CANTIDAD veces, la cual se declaró con un valor de 2.

movement_rotate_laser bloque que rota el láser un número determinado de grados. Su estructura es la siguiente (figura 2-29):

```
<block type="movement_rotate_laser" id="...">
   <field name="ROTATION">ANTI_CLOCKWISE</field>
   <value name="AMOUNT">
       <block type="variables_get" id="...">
           <field name="VAR" id="" variableType="">x</field>
       </block>
   </value>
```

Figura 2-29: Ejemplo de bloque tipo "movement_rotate_laser"

- <field> contiene el tipo de rotación que se realizará, en sentido horario (CLOCKWISE) o antihorario (ANTI_CLOCKWISE).
- <value> contiene el número de veces que rotará el láser. En este caso se incluye la variable X, la cual tiene un valor de 3.

2.6 Revisión de plataformas de análisis sobre el aprendizaje de estudiantes

En el ámbito de la educación y el uso de juegos serios para el aprendizaje de la programación, existen diversas plataformas que han sido desarrolladas con el objetivo de facilitar la evaluación y análisis del desempeño de los estudiantes. Estas plataformas permiten a los profesores obtener información muy importante sobre el progreso de los alumnos, identificar áreas de mejora y proporcionar retroalimentación efectiva.

En este apartado, se presentarán y analizarán algunas de estas plataformas similares a la que se pretende desarrollar en el presente trabajo, destacando sus características, funcionalidades y contribuciones al campo de la educación en programación. Cada subapartado se centrará en una plataforma en particular, examinando su enfoque, su arquitectura y las soluciones que ofrecen para el análisis de datos de juegos serios. A través de este análisis comparativo, se busca comprender mejor las fortalezas y debilidades de cada una y extraer lecciones útiles para la implementación de la plataforma propuesta.

2.6.1 Bealink

Bealink [16] es una plataforma de aprendizaje acreditada por ADL que ofrece una experiencia de usuario única y personalizada, debido a que integra diferentes

modalidades y formatos pedagógicos. Gracias a esto, facilita la colaboración y el intercambio de conocimientos entre los alumnos. Con el fin de llevar a cabo sus análisis de datos, utilizan el estándar xAPI para recopilar y almacenar información sobre las actividades de aprendizaje de los alumnos y ofrecer una visualización de los datos en tiempo real mediante cuadros de mando interactivos que muestran indicadores como el progreso, el rendimiento, la satisfacción y el impacto del aprendizaje. Asimismo, posibilita analizar los datos mediante técnicas avanzadas, como el análisis predictivo o el aprendizaje automático. De esta forma, Bealink ayuda a la empresa a tomar decisiones basadas en datos y a mejorar continuamente su oferta y su calidad de aprendizaje.

Los alumnos pueden acceder a todos los recursos de aprendizaje de la empresa desde un solo punto de entrada en Bealink, lo cual simplifica en gran medida el acceso y la distribución. También les permite elegir entre diferentes tipos de contenidos, como módulos de e-learning, eventos de práctica, eventos sincrónicos, evaluaciones, podcasts, artículos, micro-contenidos, etc. Además, pueden personalizar su experiencia de aprendizaje según sus competencias, objetivos e intereses, y así les ayuda a crear sus propias listas de reproducción multimodales. La recomendación de contenidos relevantes es algo que también tienen en cuenta y es algo que anima a compartir su experiencia a los alumnos y a aprender de sus compañeros. Esto se realiza con el fin de promover el aprendizaje colaborativo y social entre todos los participantes.

Por otra parte, la empresa puede optimizar y diversificar su estrategia de aprendizaje, mejorar la eficiencia de costes, desarrollar nuevos enfoques pedagógicos y aumentar el compromiso de los alumnos con la plataforma. Se integra perfectamente con todos los sistemas de gestión del aprendizaje (LMS), los sistemas de gestión del talento (TMS), las herramientas de autoría y los catálogos de contenidos de la empresa. Además, les permite aprovechar el conocimiento colectivo de sus empleados y retener más experiencia en la organización. También promueve el desarrollo de una cultura de aprendizaje en la empresa, alineando el aprendizaje con el día a día y las necesidades reales de los empleados. De esta forma, la empresa puede tener una visión global y una fácil administración de todo el ecosistema de aprendizaje con el solo uso de esta plataforma.

Bealink también tiene en cuenta a los formadores y les permite visualizar y analizar los datos de aprendizaje de los alumnos, utilizando el estándar xAPI y un motor de análisis avanzado. De esta forma, pueden medir el impacto del aprendizaje en el rendimiento, identificar las fortalezas y debilidades de los alumnos, detectar las brechas de competencias y las oportunidades de mejora, y proporcionar retroalimentación personalizada. Por último, también les permite diseñar experiencias de aprendizaje multimodales y adaptativas, utilizando un editor intuitivo y flexible.

2.6.2 Core LRS

Core LRS [17] es un sistema de almacenamiento de registros de aprendizaje (LRS) acreditado por ADL que permite a las organizaciones recopilar, analizar y visualizar datos sobre las actividades de los alumnos en diferentes plataformas y formatos. Con el objetivo de visualizar los datos cuenta con varias pestañas, como un log de actividades que permite ver qué usuario ha hecho qué actividad en qué momento, un tablero para ver las peticiones a tu LRS por país o por navegador, una gráfica con las peticiones por día o semana, ver las declaraciones procesadas totales o las del día actual… Además, cuenta con un buscador para poder filtrar por ID de usuario, ID de actividad, el verbo o timestamp de la declaración.

Los alumnos pueden acceder a cursos y recursos de aprendizaje desde cualquier dispositivo y lugar, sin necesidad de un LMS. La plataforma soporta tanto el protocolo xAPI como el SCORM, lo que significa que los alumnos pueden interactuar con contenidos variados y dinámicos que se adaptan a sus gustos y necesidades. La información que se recoge de los alumnos incluye datos como el nombre, el correo electrónico, el rol, el grupo, la actividad realizada, el tiempo empleado, el resultado obtenido, el nivel de satisfacción, etc. También registra y almacena todos los datos sobre el rendimiento, el progreso y la experiencia de los alumnos, lo que les ayuda a tener un mayor control y seguimiento de su aprendizaje.

Respecto a la empresa le proporciona una herramienta fácil de usar para gestionar y optimizar su estrategia de aprendizaje. Core LRS recoge datos de múltiples fuentes y formatos, lo que permite tener una visión global y detallada del impacto del aprendizaje en la organización. También cuenta con paneles personalizables que crean

visualizaciones de datos para todos los interesados en el aprendizaje, desde los responsables de línea hasta el nivel ejecutivo. Además, ayuda a identificar las brechas de competencias y las oportunidades de mejora en la oferta formativa y permite crear informes personalizados sobre los indicadores clave de rendimiento (KPI) del aprendizaje, como la participación, la finalización, la satisfacción y el impacto en el negocio. Gracias a esto, la empresa puede tomar decisiones basadas en datos para mejorar la eficiencia y el retorno de la inversión del aprendizaje.

Por último, a los formadores les facilita el diseño y la entrega de contenidos de aprendizaje efectivos y atractivos, ya que ofrece un kit de desarrollo de software (SDK) que permite conectar fácilmente las aplicaciones con el sistema de la propia plataforma, habilitando funcionalidades clave, como generar URLs de lanzamiento, crear y consultar declaraciones xAPI, o gestionar el estado del aprendizaje. Asimismo, soporta protocolos nativos de lanzamiento, como xAPI, CMI5 y SCORM, lo que permite a los formadores integrar contenidos de diferentes tipos y fuentes sin problemas. Además, proporciona informes y análisis sobre el comportamiento, la participación, los resultados y las preferencias de los alumnos, lo cual les ayuda a evaluar la efectividad de sus contenidos y a mejorarlos continuamente.

2.6.3 Veracity Learning

Veracity Learning [18] es una plataforma de software acreditada por ADL para registrar y reportar datos de experiencia y rendimiento de aprendizaje usando la especificación Experience API (xAPI). Es una solución flexible y escalable que se adapta a las necesidades de cada organización. Está disponible como una opción de Software as a Service (SaaS) alojada en la nube, o como una opción On-Premises instalada en la propia red del cliente.

Respecto a la visualización y análisis de los datos, Veracity Learning cuenta con potentes herramientas para medir los resultados y las tendencias del aprendizaje. La plataforma ofrece paneles e informes predeterminados que muestran información relevante sobre las declaraciones xAPI generadas hasta el momento. Además, la plataforma permite crear gráficos e informes personalizados usando el constructor de

gráficos, las herramientas de consulta gráfica avanzada, el lenguaje de consulta Veracity (VQL) y el soporte especial para perfiles xAPI comunes.

Para poder realizar los análisis de datos recoge una gran cantidad de información sobre lo que hacen los alumnos, tanto dentro como fuera de la plataforma. Esta información incluye datos sobre las acciones, los contextos, los resultados y los estados emocionales de los alumnos. Después de reunir toda esta información, la utiliza para realizar análisis avanzados que permiten extraer conclusiones sobre el aprendizaje que pueden responder preguntas como: ¿Qué contenidos son más efectivos para el aprendizaje? ¿Qué factores influyen en el rendimiento y la motivación de los alumnos? ¿Qué competencias han adquirido o mejorado los alumnos?

A los alumnos les permite acceder a contenidos de aprendizaje en diferentes formatos y dispositivos, y enviar sus datos de experiencia a través de xAPI. Los alumnos pueden ver sus propios informes y gráficos para seguir su progreso y logros. Además, también soporta el lanzamiento de xAPI, lo que facilita el acceso a contenidos externos sin necesidad de iniciar sesión en múltiples plataformas. Asimismo, fomenta tanto el aprendizaje autónomo como el colaborativo, permitiendo a los alumnos interactuar con otros usuarios y compartir sus experiencias.

Por otro lado, a la empresa le ayuda a gestionar usuarios, clases y cursos, y a subir contenidos de aprendizaje a la plataforma. La empresa puede crear y asignar roles y permisos a los usuarios, y configurar las opciones de privacidad y seguridad de los datos. Por último, también ofrece una arquitectura de plugins que permite integrar funcionalidades adicionales o personalizadas a la plataforma. Gracias a esto, ayuda a la empresa a optimizar sus recursos de aprendizaje, mejorar la calidad de la formación y evaluar el impacto del aprendizaje en el rendimiento.

Capítulo 3 - Análisis en Python sobre las interacciones del usuario de Articoding

Para realizar nuestro proyecto hemos utilizado GitHub como sistema de control de versiones. Esto nos ha permitido poder programar en paralelo de forma mucho más sencilla y tener todo almacenado. También nos ha permitido recuperar versiones anteriores cuando nos ha hecho falta. El enlace a nuestro repositorio es el siguiente: https://github.com/angelhort/TFG_Articoding_22-23, donde se podrán encontrar todas las funciones que se mencionan en este apartado.

El propósito de recoger datos generados por Articoding es el de poder evaluar el conocimiento y el uso de juego de cada uno de los usuarios, así como poder comparar el rendimiento entre los propios alumnos y las sesiones de juego. Para ello es necesario realizar un análisis sobre los datos xAPI generados durante cada sesión de juego de forma que podamos obtener información relevante sobre el aprendizaje y uso del juego: qué niveles les han costado, qué conocimientos no han adquirido o qué jugadores van más retrasados y necesitan reforzar algunos conceptos. En una clase, esto permite al responsable principal tener control sobre la sesión, conocer cómo ha funcionado el juego y ayudar, posteriormente, a los alumnos a mejorar en los conceptos en los que necesitan un refuerzo.

Las trazas generadas por los jugadores están todas juntas en un array dentro de un archivo JSON. Este array se recorre una única vez, ya que es un archivo bastante pesado y recorrerlo varias veces supondría mucho tiempo de espera para el usuario.

Para extraer los datos se ejecuta un script en Python desde un servidor de Node.js, siguiendo lo explicado en [19]. Este script se ejecuta sólo la primera vez que el usuario intenta acceder a los datos de la sesión. Después, los datos extraídos son almacenados en distintos archivos en formato JSON para acceder a ellos de forma más rápida cuando el usuario hace una solicitud. Los tutores fueron los que nos propusieron utilizar Python debido al gran número de librerías que existen para analizar datos y que es un lenguaje muy sencillo con el que aprender algo nuevo. El extenso ecosistema de librerías de Python proporciona una amplia gama de herramientas y funciones que facilitan la

manipulación, el análisis y la visualización de datos. Además, su facilidad de uso y su legibilidad lo convierten en una opción popular entre los investigadores y analistas de datos.

Como se ha mencionado anteriormente, las trazas tienen un formato concreto. Estas tienen un verbo, un objeto, un actor y un timestamp, entre otra información sobre el contexto o extensiones específicas de la herramienta. La función que extrae los datos dentro del script se llama: *extraerTiemposPorNivelJugador*.

Para obtener los datos se itera sobre la lista de las trazas y se obtiene la siguiente información:

- Timestamps de inicio y de finalización, así como las estrellas obtenidas, de cada vez que un jugador accede a un nivel.
- Los intentos empleados por el jugador cada vez que inicia un nivel y si consigue superarlo.
- Los timestamps de inicio y fin del juego, para saber así cuánto tiempo ha jugado.
- Los errores que ha cometido una vez ha completado un nivel.

Una vez se han extraído los datos necesarios de las trazas, se analizan para sacar valores que sirvan como visualización del desempeño de los participantes dentro de la sesión. Esta parte del script es mucho más rápida, puesto que ya se tienen todos los datos almacenados en diccionarios y se intenta iterar lo menos posible los datos para no perjudicar la experiencia del usuario.

Respecto al preprocesamiento de los datos, se realiza una selección de las trazas a utilizar. En particular, se descartan las trazas correspondientes a los niveles creados por los jugadores, ya que estas no aportan información relevante para el análisis de los datos. Además, para llevar a cabo la distinción de alumnos en grupos (el clustering), se realiza un tratamiento de los valores atípicos que se encuentran por encima del tercer cuartil. Esta estrategia se basa en las referencias bibliográficas [20] y [21], donde se aborda la importancia de identificar y tratar los datos atípicos para obtener resultados más confiables y significativos en el análisis de clustering. Al aplicar este enfoque, se busca asegurar que los grupos generados sean representativos y se ajusten adecuadamente a los patrones de comportamiento de los jugadores en el contexto del

propio juego. En este proceso, la utilización de xAPI resulta de gran utilidad, puesto que proporciona un conjunto de especificaciones y directrices que permiten la captura y registro de datos de manera estandarizada. Esto facilita la identificación de las trazas relevantes y la aplicación de los criterios de filtrado y descarte necesarios para el preprocesamiento de los datos. Por ejemplo, siguiendo los ejemplos recogidos en la tabla del Apéndice A, se puede obtener siempre los mismos datos accediendo a los campos de las trazas de la misma forma.

3.1 Tiempo de cada jugador para completar un nivel desde que lo inicia

Con los tiempos que ha tardado cada jugador en completar un nivel podemos hacer la media de tiempo empleado en los niveles y visualizar qué niveles han sido los que más tiempo han requerido. De esta forma el profesor de la clase puede obtener qué niveles han sido los que más se han atascado en su clase y han requerido gastar más tiempo en poder ser resueltos por parte de los alumnos.

Cada vez que itera el bucle que lee las trazas se comprueba si la definición del objeto es un nivel. Una vez sabemos que la traza es de un nivel comprobamos si la acción ha sido de inicialización (*initialized*) o de finalización (*completed*). Si la traza, además, tiene un objeto *result* sabemos que ha sido que ha iniciado el nivel desde el menú. Sí no estuviese este objeto sabemos que el jugador ha reiniciado el nivel desde este mismo.

Almacenamos entonces cada vez que un jugador inicia un nivel y cada vez que lo finaliza (lo supera o sale del nivel), como se puede ver en la figura 3-1. Para almacenar estos datos se tiene un diccionario con clave el id del actor y valor otro diccionario cuya clave es el id del nivel y cuyo valor es una lista con los timestamps de cuando ha iniciado y finalizado el nivel y las estrellas obtenidas cada vez.

```
"actor": {
  },
"verb": {
  "id": "http://adlnet.gov/expapi/verbs/initialized"
  },
"object": {
  "id": "tutorials_1",
  "definition": {
    "type": "https://w3id.org/xapi/seriousgames/activity-types/level"
    }
  },
"result": {
  "extensions": {
    "code": "\r\n<xml>\r\n    <variables />\r\n    <block type=\"start_sta
    }
  },
"timestamp": "2022-11-30T15:30:26.861Z"
}
```


Figura 3-1: Ejemplo de traza donde un jugador inicializa un nivel

Las estrellas obtenidas por los jugadores en cada nivel las podemos extraer del objeto Resultado cuando este ha terminado el nivel (el id del verbo es *completed*), tal y como se observa en la figura 3-2:


```
"actor": {
  },
"verb": {
  "id": "http://adlnet.gov/expapi/verbs/completed"
  },
"object": {
  "id": "tutorials_1",
  "definition": {
    "type": "https://w3id.org/xapi/seriousgames/activity-types/level"
    }
  },
"result": {
  "success": true,
  "score": {
    "raw": 3
    },
  "extensions": {
    "steps": 1,
    "special_block": true,
    "minimum_steps": true,
    "no_hanging_code": true
    }
  },
"timestamp": "2022-11-30T15:30:36.898Z"
},
```


Figura 3-2: Ejemplo de traza donde se pueden obtener las estrellas conseguidas

En la función *getMediaTiempoPorNivel* del script se recibe el diccionario que contiene los timestamps de inicio y fin de un nivel junto con las estrellas obtenidas.

En esta función se recorre la estructura de diccionarios para recoger los timestamps ya extraídos. Se resta el timestamp de fin con el de inicio para sacar la diferencia de tiempo y se almacena, junto con las estrellas obtenidas, en un diccionario similar al de los timestamps. Tenemos así cuánto ha tardado cada participante cada

vez que ha intentado superarse un nivel. Con esto obtenemos el tiempo medio para superar cada nivel.

3.2 Número de veces que los jugadores intentan un nivel y si lo han superado con éxito

Al igual que con el tiempo, los intentos nos permiten saber qué niveles han resultado más difíciles. Estos datos suelen estar relacionados directamente. Ambos datos también sirven para generar diagramas de cajas y detectar valores atípicos. Los valores atípicos son aquellos valores que se alejan mucho de la mediana. Podemos así descubrir qué alumnos en concreto se han atascado en ciertos niveles. Esto permite mejorar el aprendizaje de alumnos concretos que han tenido dificultades obteniendo ciertos conocimientos.

Como hemos mencionado en el apartado anterior, si no aparece el objeto *result* en una traza de inicialización de un nivel es porque el jugador ha reiniciado el nivel. Llevamos entonces un diccionario cuya clave es el id del jugador y su valor es otro diccionario cuya clave es el id del nivel y su valor es una lista con los intentos empleados y si ha conseguido pasar el nivel cada vez que lo ha iniciado.

Obtenemos los intentos empleados por cada jugador en cada nivel sumando los intentos empleados cada vez que ha iniciado el nivel hasta completarlo, ya que estos datos habían sido obtenidos en la iteración de las trazas. Hacemos una media también de los intentos en todos los niveles para que el profesor pueda visualizar donde han tenido que probar más sus alumnos.

3.3 Último nivel alcanzado por jugador

Al conocer el último nivel alcanzado por los jugadores podemos ver hasta dónde ha llegado cada participante y así saber quién ha aprendido más. Podemos también comparar estos datos con el resto de las sesiones y ver si la sesión actual está dentro de unos valores normales. Permite también al profesor ver donde ha parado de jugar cada alumno y que conceptos no ha podido alcanzar.

Para obtener el último nivel alcanzado por cada jugador basta con ver cuál ha sido el último nivel del que se tiene en el diccionario devuelto por la función mencionada

en el apartado 3.1. Este método del script que obtiene el último nivel alcanzado se llama *getUltimoNivelAlcanzado*.

3.4 Tiempo total de juego por jugador

Con el tiempo total que ha jugado cada jugador podemos ver si el último nivel alcanzado comparado con este tiempo tiene sentido. Los jugadores que juegan más tiempo tienen más posibilidades de alcanzar un nivel más avanzado. Sirve entonces para dar coherencia a los datos anteriores y poder comparar el tiempo jugado con el último nivel alcanzado.

Podemos saber que un jugador ha iniciado el juego gracias a la siguiente traza en la figura 3-3:

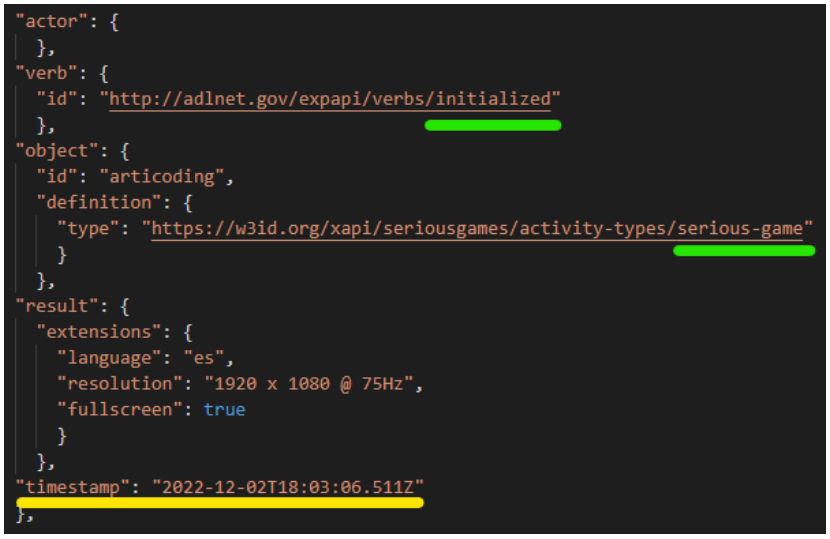

Figura 3-3: Ejemplo de traza donde un jugador inicializa el juego

Existe también una traza para saber si el jugador ha cerrado el juego. El problema es que algunos jugadores cierran la pestaña del juego directamente y como el juego no es orientado a conexión no podemos saber exactamente cuándo ha finalizado el juego. Para solucionar esto tomamos como traza de finalización de juego la última traza registrada por el jugador antes de iniciar de nuevo el juego.

Esta información se almacena en un diccionario con clave el id del jugador y valor una lista en la cual cada valor es un par con los timestamps de inicio y de fin de cada vez que el jugador inicia el juego.

Para obtener el tiempo que ha estado un jugador en el juego se restan los timestamps de inicio y de fin que se obtuvieron en la iteración de las trazas. Si el jugador ha entrado al juego más de una vez tendremos varios pares de inicio y fin. Basta con sumar todas las diferencias para obtener el tiempo total. Esto se consigue en la función del script *tiempoTotalJuego*.

3.5 Código XML con las tarjetas utilizadas por cada jugador en cada nivel

Cogiendo el código XML de cada jugador una vez completa un nivel podemos detectar malos hábitos de programación o si verdaderamente ha entendido los conceptos. Por ejemplo, muchos niveles en los que hay que utilizar variables pueden completarse sin necesidad de crearlas. Analizando estos códigos podemos detectar si ha creado las variables cuando hacía falta. También podemos ver si ha dejado código muerto. Esto sirve para mejorar el aprendizaje desde una etapa temprana y evitar que los participantes arrastren estos errores, que a nosotros nos parecen básicos, en su formación futura.

Cuando una acción sobre un nivel tiene como id de verbo *progressed* también trae un código XML con las tarjetas empleados por el jugador.

Este código se analiza por cada iteración de finalización de nivel para comprobar si el usuario ha cometido errores como crear variables cuando no hacen falta, no crearlas cuando es necesario o si tiene código muerto que nunca llega a ejecutarse.

Cuando encontramos un código XML podemos detectar el código muerto si hay más de un bloque colgando del bloque <xml>. Esto se debe a que los bloques que están encadenados se encuentran uno dentro del otro en el código XML, lo cual es posible analizar gracias a la librería ElementTree de Python [22].

Para detectar errores en la creación de variables se almacena el nombre de todas las variables declaradas. Esta información la obtenemos de la etiqueta <variables>, como se puede ver en la figura 3-4. Si la variable se ha utilizado menos de dos veces entonces no era necesario crearla.

Figura 3-4: Ejemplo de estructura XML

Para detectar si había que crear una variable se recogen todos los bloques de tipo "text" o "math_number". Estas tarjetas son las que los usuarios utilizan como valores literales. Si se utiliza alguna de estas tarjetas con el mismo valor más de una vez significa que el usuario debería de haber creado una variable y asignarle el valor para utilizar la tarjeta literal solo una vez.

La información sobre los errores es guardada directamente en archivos JSON dentro del servidor. Estos serán leídos desde el propio servidor para ser mandados a la página web.

Por último, también se generan diagramas de caja utilizando el tiempo e intentos que se extraen para los anteriores análisis. Para obtenerlos, se transforman los datos sobre el tiempo y los intentos empleados en un estilo aceptado por la librería Plotly, la cual nos ha permitido generar los diagramas de caja que se visualizan en la web. Estos diagramas de caja se guardan en formato JSON y son abiertos desde el servidor para colocarlos en la web.

En los siguientes subapartados se explican los análisis realizados y la información relevante obtenida con cada uno de ellos.

3.6 Clustering

Con el objetivo de realizar un clustering que permitiera diferenciar a los jugadores en grupos según su rendimiento en el juego, se usaron tanto los análisis mencionados previamente como los datos de edad y género obtenidos de un archivo CSV creado tras un cuestionario previo realizado a los alumnos antes de cada sesión.

Tras llevar a cabo numerosas pruebas con diferentes combinaciones de variables, como edad, género y tiempo o edad/género, tiempo e intentos, no se lograron obtener grupos con características significativas que permitieran una diferenciación clara entre los jugadores. Sin embargo, tras aplicar el Análisis de Componentes Principales (PCA, por sus siglas en inglés) y una previa estandarización [23], usando todos los análisis de tiempo, número de errores, intentos y nivel alcanzado, si que se consiguieron clústeres con valores significativos, como se puede ver en la figura 3-5. Finalmente se descartaron los valores de género y edad debido a que no aportaban valor a los clústeres generados.

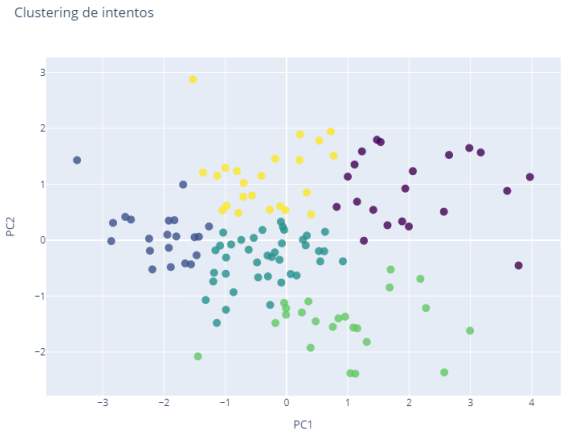

Figura 3-5: Clústeres tras aplicar el PCA y K-Means

Cabe destacar que se probaron con diferentes algoritmos de clustering para comprobar cuál ofrecía mejores resultados y poder ver sus diferencias, entre ellos K-Means, DBSCAN, Hierarchical Agglomerative Clustering y Spectral Clustering. De todos ellos, el que proporcionó clústeres más sólidos fue K-Means, puesto que los demás no consiguieron generar grupos significativos y bien diferenciados. Por lo tanto, para la plataforma se escogió únicamente realizarlo con este.

Además, este proceso se realiza para cada clase empleando los valores de los alumnos de las anteriores clases almacenadas en la plataforma, para así conseguir siempre los mismos grupos y poder darle siempre un mismo nombre a cada uno de ellos. Los datos acumulativos de las clases se guardan en un fichero CSV con los datos tras aplicar el PCA, evitando así almacenar todas las trazas de las clases, lo que sería muy poco eficiente. Los distintos grupos que hay son:

- Grupo 0: Jugadores con valores atípicos respecto a tiempo, intentos o errores. Son los que necesitan ayuda con más urgencia.
- Grupo 1: Clúster formado por los alumnos que han jugado menos niveles (y por lo tanto tienen menos errores), con un tiempo medio normal, pero con más intentos, por lo que no han jugado tantos niveles debido a ello.
- Grupo 2: Formado por los alumnos que más niveles se han pasado y con muy pocos intentos de media. Son los que tienen más errores al jugar más niveles. Su tiempo también está en la media.
- Grupo 3: Jugadores que han pasado bastantes niveles, pero no tantos como el anterior grupo, teniendo unos intentos similares, pero más tiempo de media. Tienen menos errores al jugar menos niveles que el grupo 2.
- Grupo 4: Son los que tienen menos errores, pero son los segundos por la cola respecto a número de niveles superados, solo por delante del grupo 1. Se los suelen pasar con pocos intentos, pero su tiempo medio es el mayor de todos.
- Grupo 5: Clúster formado por alumnos que se han pasado un número de niveles similar al grupo 3, pero con algo más de errores. Su tiempo es inferior, pero sus intentos son mayores.

Gracias a la existencia de estos grupos, los profesores podrán identificar a cuál pertenece cada alumno y poder actuar en consecuencia más fácilmente.

Capítulo 4 - Plataforma web de análisis de datos

Este apartado describe la creación de la aplicación web, explicando las tecnologías y herramientas utilizadas en su desarrollo, así como su arquitectura y diferentes funcionalidades. Además, se detalla el diseño de la página web y las diferentes funcionalidades que se han implementado mediante el análisis de los datos recopilados en las trazas generadas de Articoding.

4.1 Tecnologías usadas

En el siguiente apartado se describen las herramientas y lenguajes de programación que se han utilizado para la construcción de la aplicación web. En primer lugar, se ha utilizado Node.js para la gestión del servidor. Las vistas de la aplicación están creadas con el módulo de Node express js, un motor de plantillas que ha permitido el desarrollo de interfaces de usuario dinámicas.

La base de datos creada para almacenar la información de los usuarios es de tipo relacional MySQL. Para su gestión se ha utilizado phpMyAdmin, un gestor de bases de datos de software libre, que incorpora una interfaz web intuitiva y fácil de usar para la gestión de la base de datos. Para levantar el servidor de la base de datos durante el desarrollo de la aplicación se utilizó XAMPP.

Para acceder a la base de datos desde la aplicación, se ha utilizado el patrón DAO en JavaScript. El patrón DAO es un tipo de patrón de diseño que se utiliza para separar la lógica de negocio de la lógica de acceso a datos. En este caso hemos definido operaciones que nos permiten interactuar con la base de datos de forma segura.

Es importante mencionar que se ha utilizado XAMPP para levantar el servidor donde se aloja la base de datos durante el desarrollo de la aplicación, lo que ha facilitado la gestión del entorno y su configuración.

Como se explicó en el apartado anterior, los scripts que generan los análisis se han desarrollado en Python, leyendo un archivo en formato JSON, el cual contiene las trazas generadas durante los diferentes experimentos. Los resultados obtenidos son

almacenados en archivos JSON que luego son interpretados por la aplicación para mostrar los datos en tablas y gráficas a través del motor de plantillas de Express JS.

Para dar estilo a la página se ha utilizado Bootstrap 5.3. Bootstrap es un framework de diseño web que permite, de manera sencilla, dar un estilo personalizado y moderno, adaptado a diferentes dispositivos. También se ha utilizado JQuery y Ajax para actualizar en tiempo real los elementos de las páginas. JQuery es una biblioteca de JavaScript que permite manipular eventos y HTMLs mientras que AJAX es una técnica que nos permite realizar peticiones HTTP para actualizar una parte de una página sin tener que recargar la vista entera, mejorando el rendimiento y la experiencia del usuario.

En definitiva, la utilización de estas tecnologías y herramientas ha permitido el desarrollo de una aplicación web completa y funcional, con capacidad para analizar el comportamiento de los jugadores de Articoding.

4.2 Arquitectura de la aplicación

La arquitectura de la aplicación se ha diseñado con una estructura MVC (Modelo-Vista-Controlador) para separar la lógica de negocios, la presentación y el control de las peticiones de usuario.

La capa de presentación está formada por los archivos de vista (.ejs), las tablas y las gráficas generadas en la capa de negocio.

La capa de controlador está formada por diferentes módulos en JavaScript, que proporcionan la lógica y las reglas de negocio de la aplicación. Esta capa se comunica con la capa del modelo para obtener y almacenar información en la base de datos.

Los archivos de controlador de ruta (admin.js, profesor.js y app.js), reciben las diversas solicitudes de los usuarios (GET o POST) que posteriormente, presentan los resultados en forma de páginas web o actualizan los cambios realizados en la aplicación y los actualiza en la base de datos.

La capa de modelo está formada por los DAOs (Objetos de Acceso a Datos) en JavaScript, que se comunican con la base de datos para realizar operaciones CRUD (Crear, Leer, Actualizar y Borrar) sobre los datos de los usuarios y los institutos.

El script de Python que se utiliza para analizar las trazas se ejecuta desde el servidor. Para conseguir esto nos basamos en un tutorial creado por J. Santhanavanich [21]. Este script solo se ejecuta la primera vez que un usuario pretende accedes a los datos de una clase. El script genera archivos JSON que son almacenados en el servidor y se abren dentro de las rutas de los controladores permitiendo reducir considerablemente el tiempo de espera del usuario. Una vez se lee un archivo JSON se transforman los datos para dejarlos más bonitos y seleccionar solo los necesarios para pasárselos en la respuesta a la plataforma web.

Dentro de nuestro servidor tenemos los tres controladores mencionados anteriormente. El controlador app.js es accesible para usuarios sin sesión iniciada pero solo permite iniciar y cerrar sesión. Los otros dos controladores son unos routers de node cuyas peticiones solo son accesibles por gente con la sesión iniciada y cuyo rol corresponda al indicado en cada uno de ellos.

El router de profesor tiene peticiones que son principalmente de acceso a datos de los alumnos (peticiones GET). Algunas de estas peticiones son redirecciones a páginas y otras devuelven un JSON con datos del desempeño de los participantes. Estas últimas peticiones son llamadas desde JQuery utilizando AJAX para que las páginas no necesiten ser recargadas cuando se quiere cambiar los datos que se visualizan en estas. Este router también dispone de dos peticiones de tipo post para asignar un nombre a un alumno y para editar los datos del usuario.

La base de datos consta de 3 tablas, como se puede ver en la figura 4-1:

- Usuario: sirve para almacenar a todos los tipos de usuario que tenemos en la aplicación. Almacenamos su nombre, su contraseña (cifrada) y su rol. Se tiene, además, un id único para que sea más fácil acceder a sus datos.
- Instituto: sirve para almacenar el nombre y un id único de cada experimento realizado.
- Sessions: Esta tabla es generada automáticamente por el módulo de Node express-session.

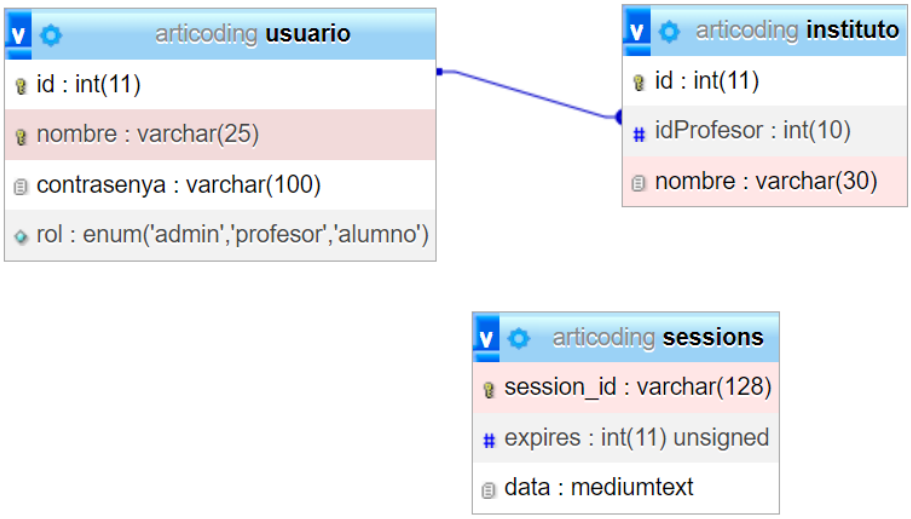

Figura 4-1: Tablas usadas para la base de datos

Para almacenar los datos en el servidor se tiene una carpeta llamada *datos,* la cual se puede ver en la figura 4-2. Esta carpeta tiene a su vez tantas carpetas como experimentos haya en la base de datos, el nombre de estas carpetas es su ID dentro de la base de datos. Dentro de cada una de estas carpetas se tiene el archivo que contiene las trazas, llamado trazasOrdenadas.json. Cuando se ejecuta el script se generan los archivos JSON mencionados anteriormente y son almacenados dentro de la carpeta correspondiente a su experimento.

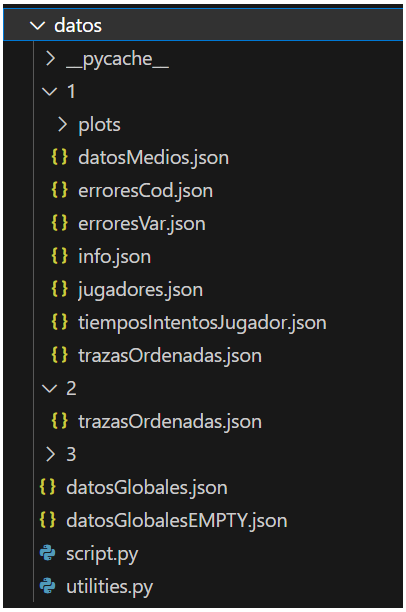

Figura 4-2: Estructura de la carpeta datos

Las subcarpetas 1, 2, y 3 son correspondientes a los experimentos realizados que se tienen almacenados. La carpeta 1 tiene varios archivos JSON y otra carpeta (plots) donde se encuentran más archivos JSON con la información de las gráficas que ha generado el script para ser luego visualizados en la web. La carpeta 2, en cambio, solo tiene el archivo *trazasOrdenadas.json.* Esto se debe a que el profesor del experimento con ID 2 todavía no ha accedido a los datos y no han podido ser generados. El archivo *datosGlobales.json* contiene datos medios de todas las clases y por eso está fuera de las carpetas de los experimentos. El archivo *datosGlobalesEMPTY.json* es un archivo con la misma estructura que el anterior pero vacío, por si hiciese falta reiniciar los datos globales. En esta carpeta también se aloja el script y un archivo con una clase utilizada por el script (utilities.py).

Por último, los diferentes textos de la plataforma se guardan en un archivo dentro del servidor y, en consecuencia, el texto mostrado en las diferentes pantallas se adapta al idioma predeterminado del navegador.

4.3 Diseño

Con el objetivo de realizar un primer boceto de la plataforma, llevamos a cabo un diseño en Figma, el cual se puede consultar en este enlace: https://www.figma.com/file/T0LgzIma845TIoVbuGWi7k/TFG_Articoding.

La página web de la aplicación tiene un diseño claro y conciso. Los colores utilizados son el blanco y una gama de azules ya que se complementan para dar una apariencia minimalista y clara. Además, la organización de la web está estructurada en dos partes diferenciadas: la vista del profesor, donde se muestran todos los análisis realizados y la vista del administrador, que contiene la gestión de usuarios y la gestión de las rutas que contienen los análisis. La página web está preparada para poderse ver en una pantalla móvil razonablemente bien. Esto ha sido posible gracias al sistema de cuadrículas (*grid*) de Bootstrap.

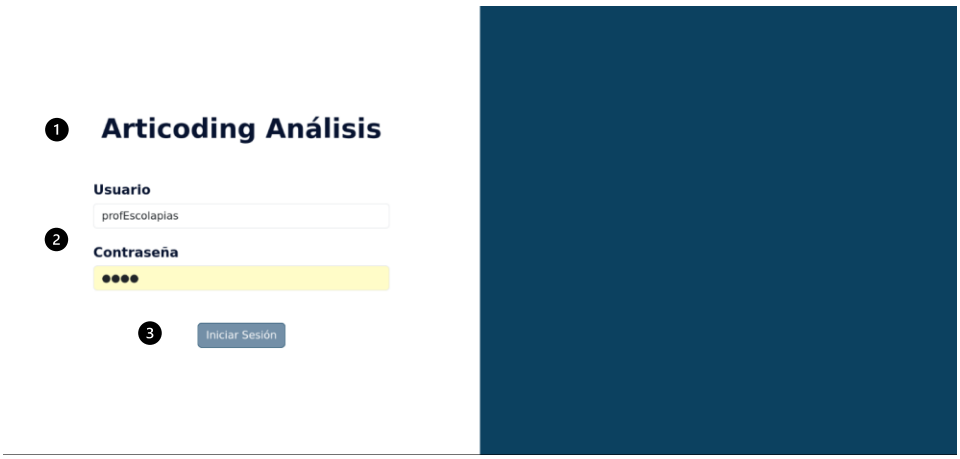

Figura 4-3: Pantalla de inicio de la plataforma

La primera vista que ve el usuario contiene un formulario de inicio de sesión, como se observa en la figura 4-3. La mitad izquierda de la pantalla contiene toda la información suministrada, con un fondo blanco, mientras que la derecha es un fondo azul. Este diseño, que divide en dos mitades la vista se va a repetir en la mayoría de las vistas de la página web. El título muestra el nombre de la aplicación (1), el formulario (2) permite introducir el nombre del usuario y su contraseña, que luego son comprobados al introducir el botón de inicio de sesión (3).

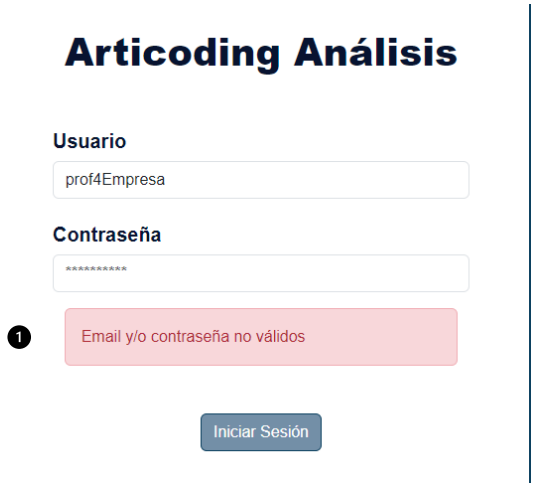

Figura 4-4: Ejemplo de fallo en el inicio de sesión

Tal y como se muestra en la figura 4-4, si hay un fallo a la hora de validar los datos de entrada se mostrará un mensaje de error (1).

Figura 4-5: Ejemplo de sesiones de un profesor

Si la cuenta de usuario que se ha introducido es la de un profesor, este será automáticamente dirigido a una vista que muestra las diferentes sesiones incluidas (siendo cada sesión un experimento con diferentes trazas), según se puede comprobar en la figura 4-5.

Cada sesión muestra el nombre del centro en el que se ha realizado el experimento, el número de participantes y un botón, el cual puede ser pulsado para generar los análisis de la sesión (1) la primera vez que se y, si los análisis ya se han generado previamente, se carga la sesión (2). En cualquiera de los dos casos una vez pulsado nos redirige a la siguiente pestaña (figura 4-6):

Figura 4-6: Pestaña de Resultados Generales

Después de seleccionar una sesión se muestra la vista general del profesor, la cual contiene tres características que son recurrentes en el diseño de la página web.

El primer elemento es la barra de navegación (1) de la aplicación, que muestra los diferentes tipos de análisis que puede consultar el profesor, además de poder cerrar

la sesión actual seleccionando el engranaje que se encuentra en la última posición a la derecha.

El segundo elemento es la gráfica (2), en este caso situada en la mitad izquierda de la vista, la cual contiene un histograma con información del número de jugadores que han completado cada una de las categorías del juego. Esto permite al usuario obtener información visual y fácil de entender y comparar con otros datos.

El tercer elemento de la vista es la tabla (3), la cual contiene información más específica, que es complementada con la más general obtenida con la gráfica. En el caso de la vista de resultados generales tenemos varias tablas, la primera de ellas contiene información del número de intentos y el tiempo medio de juego en cada categoría.

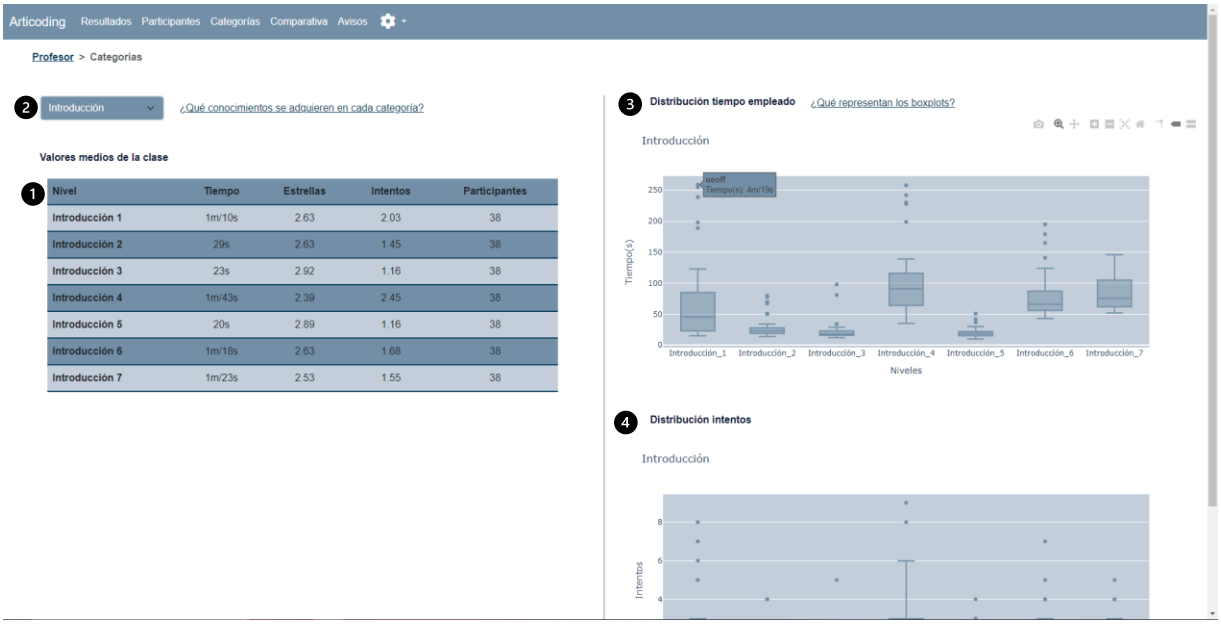

Figura 4-7: Pestaña de Categorías

Si el usuario selecciona en la opción de categorías se encontrará con una estructura similar a la anterior, como se refleja en la figura 4-7. En este caso la vista no es general, ya que los datos mostrados son de una categoría concreta.

La mitad izquierda muestra una tabla (1) con información media de cada nivel, como el tiempo jugado, las estrellas conseguidas o el número de jugadores que lo han

completado. Además, en la parte superior se encuentra un desplegable (2) con las diferentes categorías disponibles que, al ser seleccionado carga un seleccionable con las diferentes categorías.

La mitad derecha incluye varios *boxplots* (diagrama de caja y bigotes) que muestran representaciones gráficas de la distribución del tiempo empleado (3) y del número de intentos (4) en cada uno de los niveles de esa categoría, tal y como se refleja en la figura 4-8. Los *boxplots* además permiten al usuario conocer a simple vista valores atípicos.

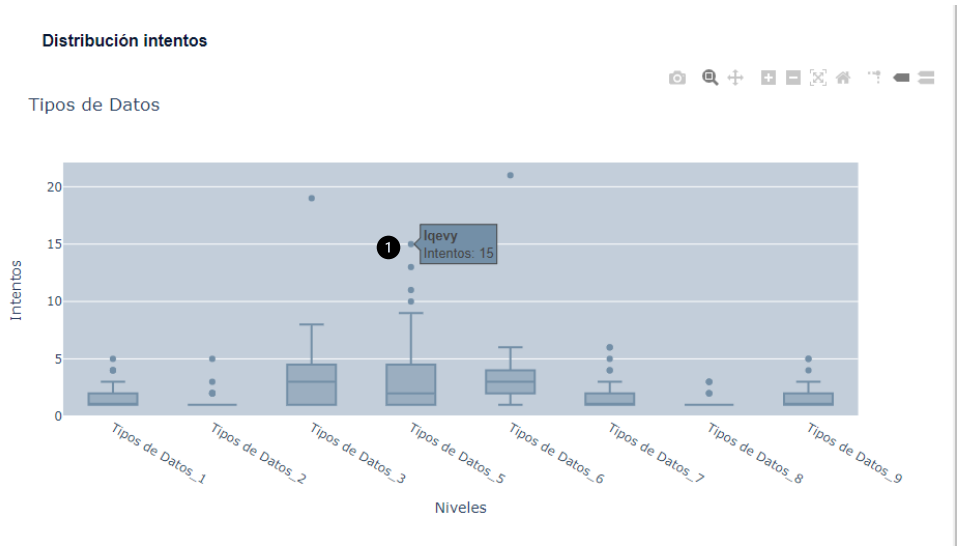

Figura 4-8: Ejemplo de boxplot para la distribución de intentos

Si el usuario coloca el cursor encima de la distribución de un nivel concreto, el *boxplot* mostrará información más específica, cómo la mediana (la línea contenida en la caja), su máximo, su mínimo o su rango intercuartílico (el contenido de la caja). Si el usuario coloca el cursor sobre algún valor atípico (1), se muestra el nombre del alumno y su valor correspondiente.

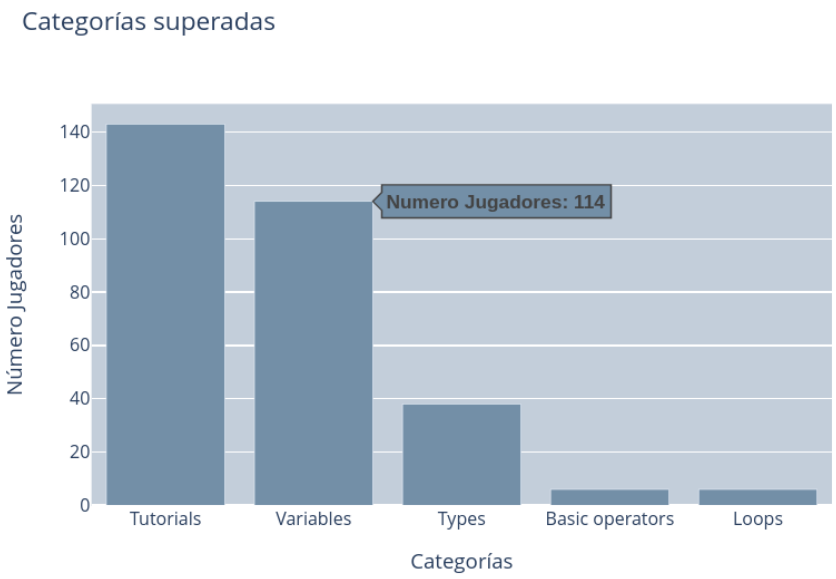

Figura 4-9: Histograma para las categorías superadas

Algo similar ocurre cuando el usuario coloca el cursor sobre una barra de un histograma, como se aprecia en la figura 4-9. En el caso de la vista general muestra el número de jugadores que han completado los niveles de una categoría concreta.

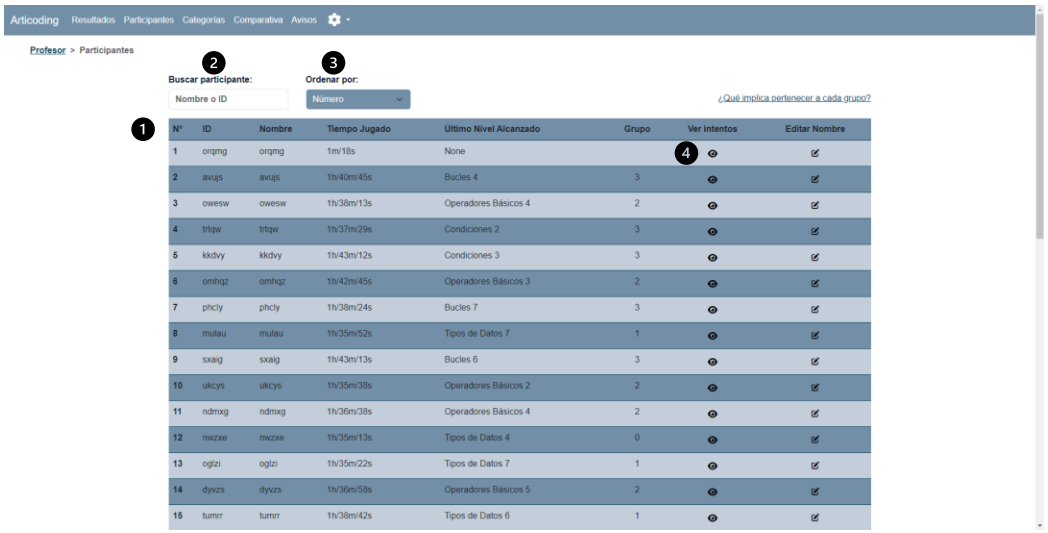

Figura 4-10: Pestaña de Participantes

Si el usuario accede a la vista de participantes se le mostrará una tabla detallada de usuarios (1) con información como su id, su nombre, su tiempo jugado o el grupo al que pertenece, como se puede constatar en la figura 4-10.

En la parte superior hay una barra de búsqueda (2) por id y un desplegable (3) que permite ordenar la tabla según una variable concreta (nombre, tiempo jugado, último nivel alcanzado, etc.). La sexta columna de la tabla de participantes contiene un icono (4) que, al ser presionado muestra una ventana emergente (figura 4-11).

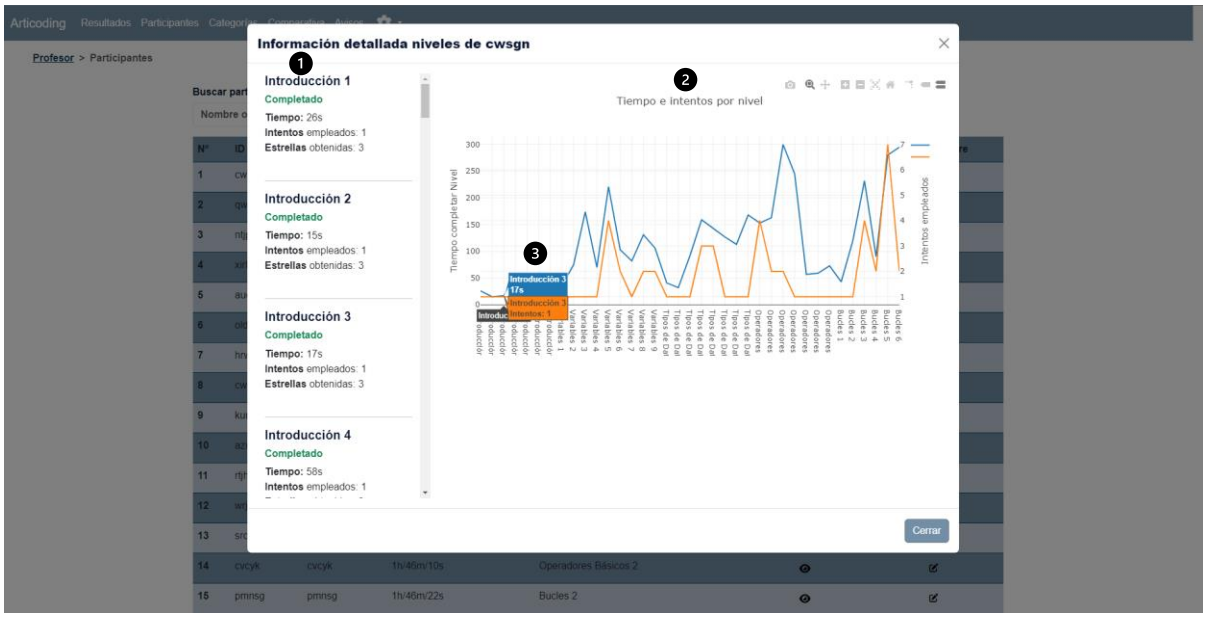

Figura 4-11: Información detallada de un alumno concreto

La ventana contiene información detallada (1) (tiempo de juego, número de intentos, estrellas conseguidas) de cada uno de los niveles jugados por el participante, tanto los completados, mostrados en verde, como los no completados, mostrados en rojo. Hay también una gráfica (2) que resume el desempeño del alumno durante la sesión. Esta gráfica también muestra los valores (3) cuando se pasa el ratón por encima.

Nº	ID	Nombre	Tiempo Jugado	Último Nivel Alcanzado	Grupo	Ver intentos	Editar Nombre
1	cwsgn	cwsgn	1h/48m/56s	Bucles 6	4		
2	qwwpu	Angel	1h/48m/48s	Bucles 1	5		
3	ntjpz	ntjpz	1h/48m/30s	Bucles 5	3		
4	xirlo	xirlo	1h/48m/7s	Operadores Básicos 4	1		

Figura 4-12: Ejemplo de edición del nombre de un alumno

Si el usuario selecciona el símbolo de la columna "Editar Nombre" (1) de un participante, aparecerá en el campo "Nombre" (2) la opción de modificarlo, como se aprecia en la figura 4-12. Esto no sólo se verá reflejado en la tabla, sino que también se

actualizarán todas las referencias a ese participante en las trazas con el nombre introducido. Hay que mencionar que para la realización de la edición en las tablas se utilizó un script obtenido de Internet [24].

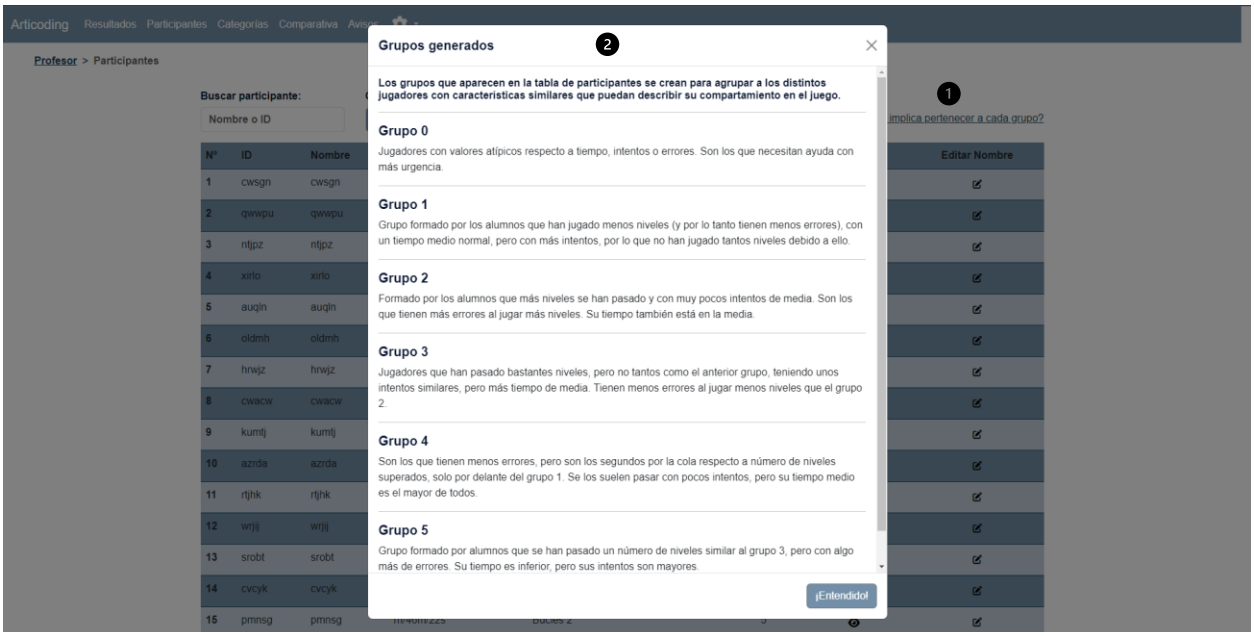

Figura 4-13: Descripción de los diferentes grupos

Asimismo, al pulsar el enlace en la parte superior de la tabla que dice ¨¿Qué implica pertenecer a cada grupo?¨ (1), se abre un modal (2) (figura 4-13) que explica las características de cada uno, formados gracias al clustering explicado en el apartado 3.6. Estos grupos que aparecen en esta tabla sirven para agrupar a los distintos alumnos con características similares que puedan describir su comportamiento en el juego.

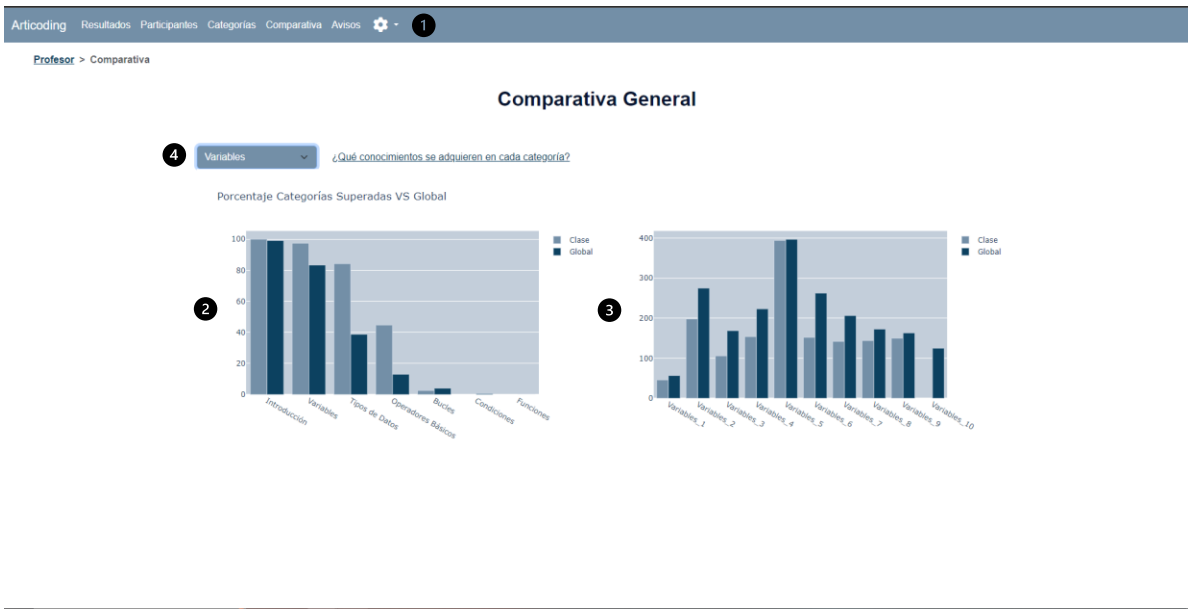

Figura 4-14: Pestaña de Comparativa

Si el usuario accede a la vista de "Comparativa" (1) (figura 4-14) se cargarán dos gráficas que comparan información de las trazas de una clase (información de esta sesión) con las globales (la media de todas las trazas obtenidas).

La gráfica que está situada en la mitad izquierda (2) compara el porcentaje de categorías superadas, y si el usuario acerca el cursor a una barra ésta mostrará su tipo (si es global/clase) y el porcentaje del número de participantes que han completado esa categoría.

En la mitad derecha (3) se muestra una gráfica que compara el número de participantes que han completado los niveles de una categoría concreta, la cual es seleccionada con el desplegable (4) disponible en la parte superior izquierda de la vista.

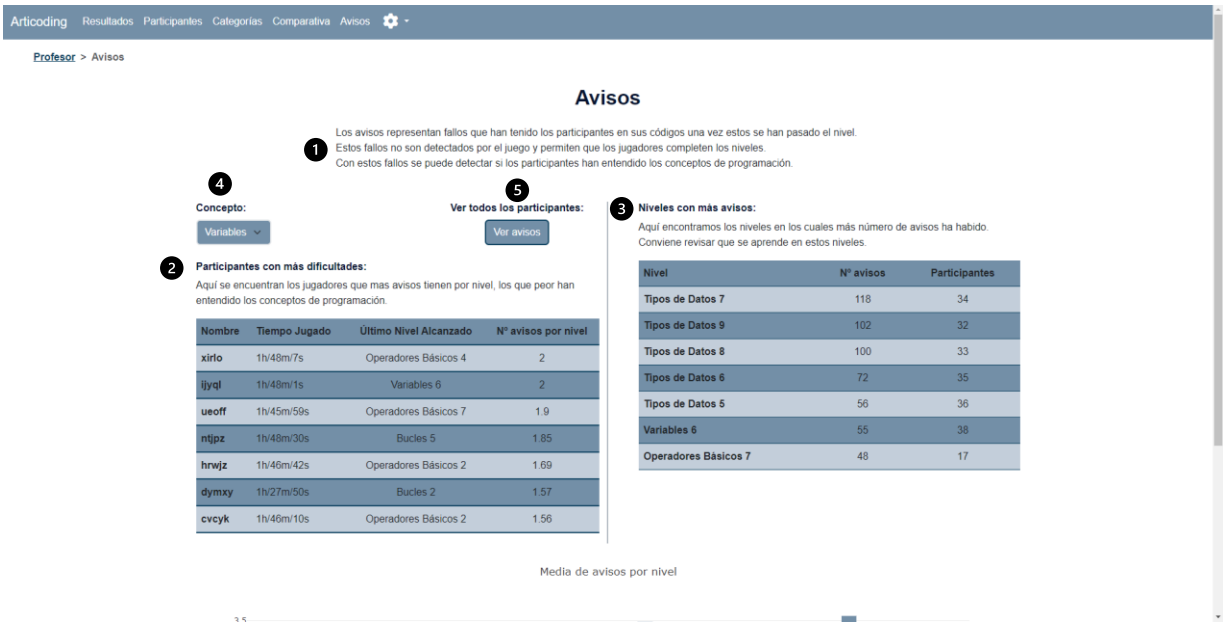

Figura 4-15: Pestaña de Avisos

Además de análisis del tiempo de juego y el número de estrellas conseguidas, la aplicación también muestra información de los diferentes fallos que han cometido los jugadores. Como es un concepto menos concreto hemos incluido una descripción del concepto de "Avisos" en la parte superior de la vista (1), como queda reflejado en la figura 4-15.

En la vista se muestran dos tablas, la primera contiene información de aquellos participantes que han cometido más fallos (2), incluyendo su nombre, tiempo de juego, último nivel alcanzado y el número de avisos recibidos por nivel y, la segunda tabla (3) se centra en los niveles en lugar de los participantes, indicando cuáles han sido los que han cometido más avisos/fallos. Además, la vista se actualiza con información del tipo de aviso (variables, código) seleccionado en el desplegable (4) de la parte superior.

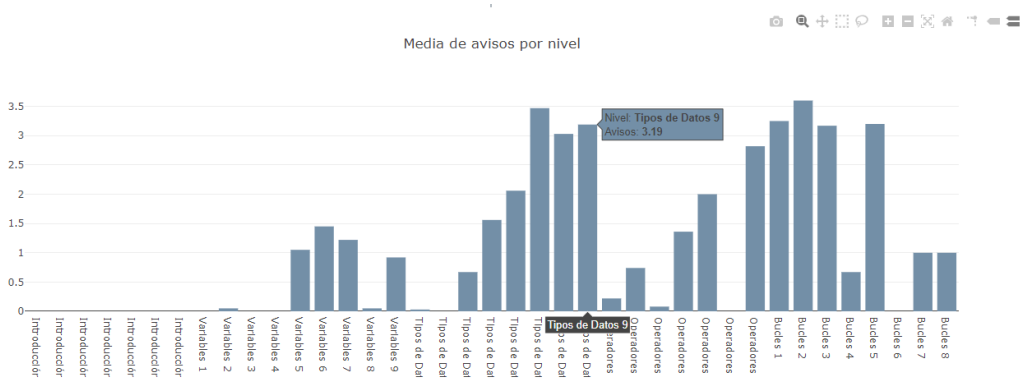

Figura 4-16: Histograma de la media de fallos por nivel

En la parte inferior de la vista hay un histograma que muestra la media de fallos que han cometido los jugadores a la hora de completar cada uno de los niveles, de acuerdo con lo ilustrado en la figura 4-16.

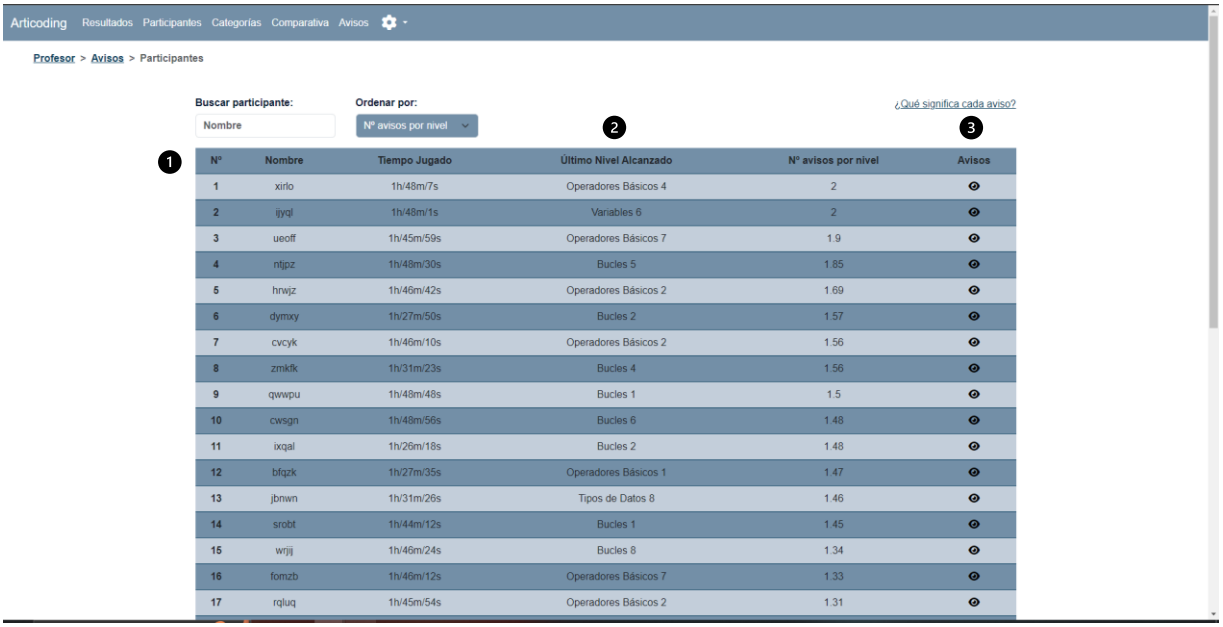

Figura 4-17: Pestaña con la tabla de todos los participantes en Avisos

Si el usuario pulsa el botón "ver avisos" (5) disponible en la parte superior izquierda de la vista de avisos mostrado en la figura 4.15 accederá a una tabla (1) con los usuarios que han cometido más avisos por nivel, incluyendo el último nivel (2) alcanzado por el jugador y un botón (3), que al ser pulsado muestra una ventana emergente con todos sus avisos cometidos (figura 4-17).

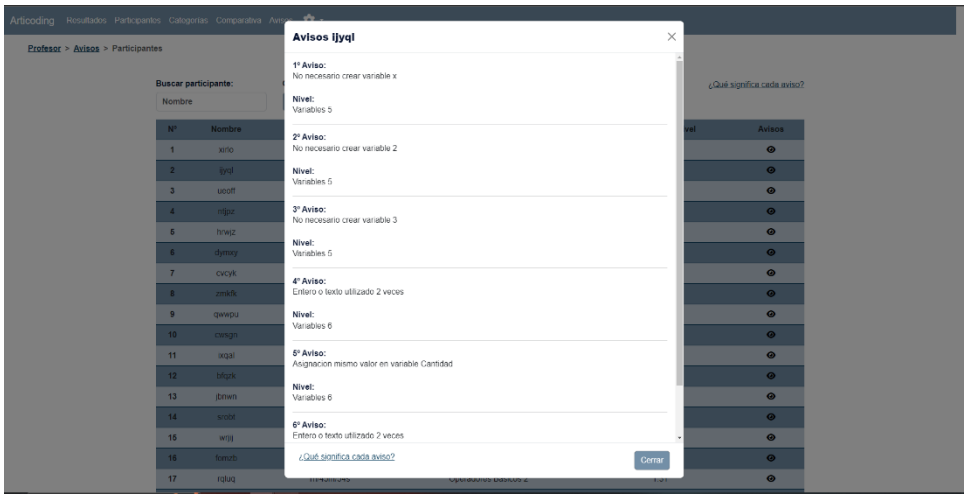

Figura 4-18: Detalle de los errores de un jugador determinado

Como se puede comprobar en la figura 4-18, los fallos cometidos por el jugador "tgwmn" son listados, incluyendo el tipo de aviso cometido y el nivel en el que ha ocurrido.

Si la cuenta que se ha introducido en el inicio de sesión corresponde a la de un administrador, se cargará su vista general, la cual contiene información de las diferentes funcionalidades que están implementadas para ese rol.

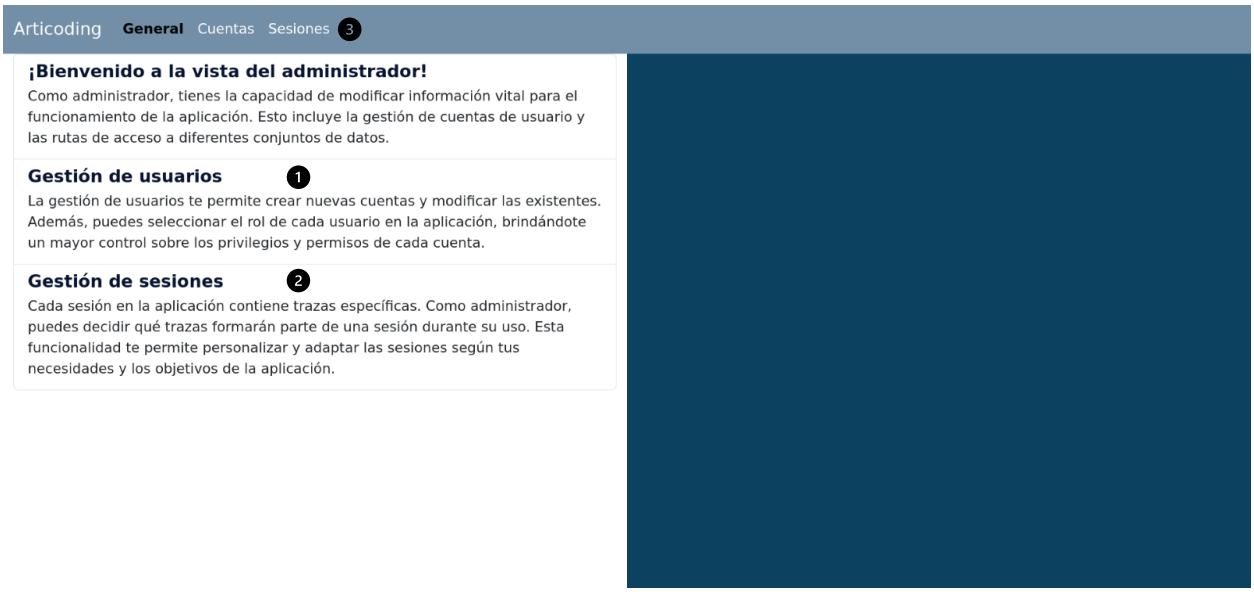

Figura 4-19: Vista general del administrador

La vista general que se puede ver en la figura 4-19 contiene explicaciones de las diferentes funcionalidades que puede realizar el administrador: la gestión de cuentas (1) y la gestión de sesiones (2). La parte de la aplicación que corresponde al administrador mantiene una barra de navegación (3) común.

La vista está dividida en dos mitades, la izquierda con el contenido de la página y la derecha con un fondo azul. Esto permite mantener una estructura similar a la de las vistas del profesor, dividiendo la aplicación en dos mitades.

En este caso, debido a que la gestión de usuarios no muestra mucha información por pantalla (como tablas o gráficas), no ha sido necesario incluir contenido en la mitad derecha, pero para mantener la estructura se ha utilizado el lado izquierdo para mostrar la información en las diferentes vistas.

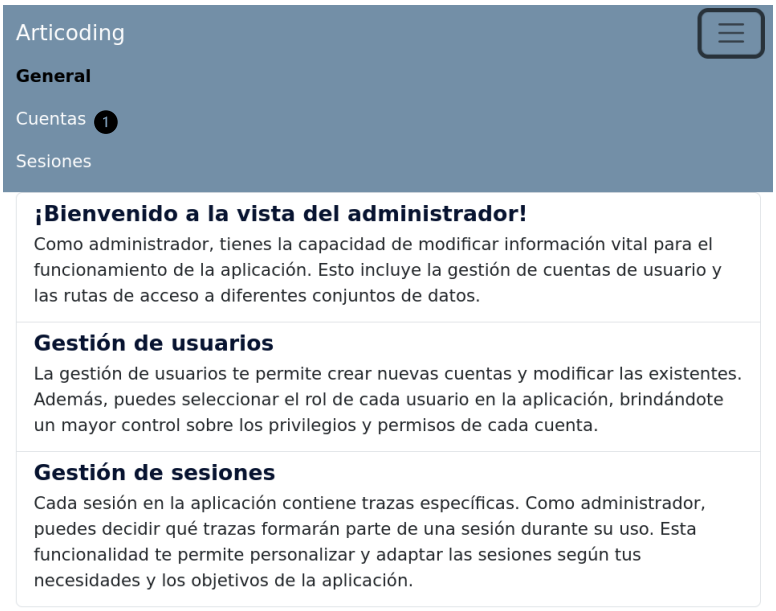

Figura 4-20: Vista del administrador en un formato de pantalla reducido

Como podemos comprobar en la figura 4-20, la vista del administrador también es responsive, mostrando la barra superior de navegación como un desplegable.

Si el usuario selecciona en la barra superior la opción de cuentas (1) se encontrará con la siguiente vista, como queda reflejado en la figura 4-21:

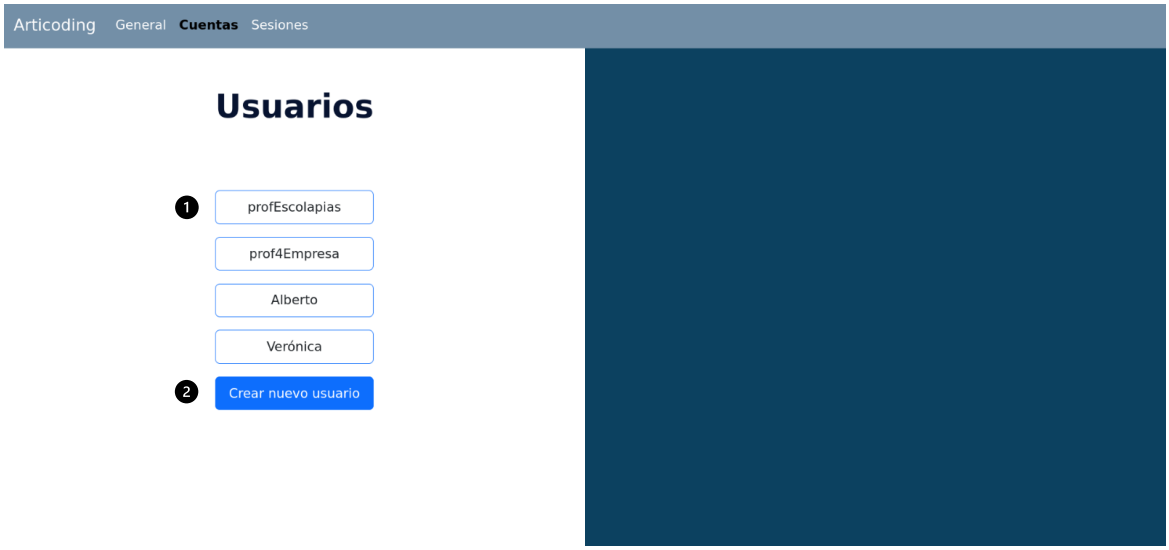

Figura 4-21: Pestaña de cuentas del administrador

Esta vista contiene una lista con todos los usuarios disponibles en la aplicación, pero, para poder distinguir a simple vista las dos funcionalidades de la gestión de usuarios: la modificación de un usuario existente (1) y la creación de uno nuevo desde cero (2), se han utilizado los dos colores principales de la aplicación, blanco y azul, respectivamente.

Si el administrador selecciona un usuario existente se encontrará con la siguiente vista reflejada en la figura 4-22:

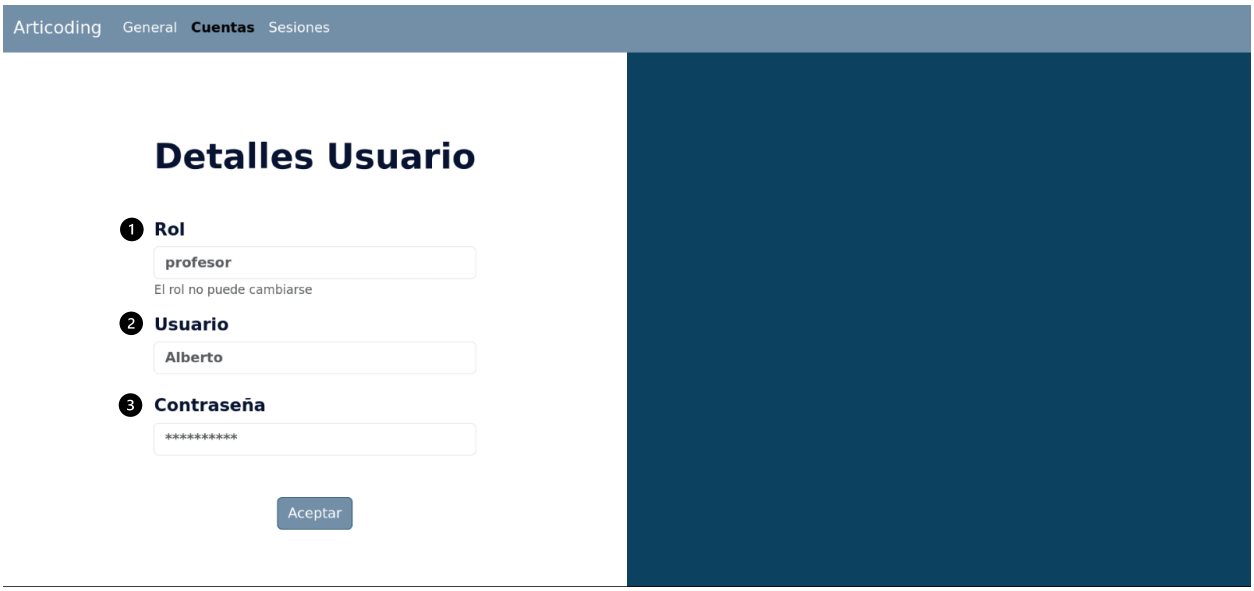

Figura 4-22: Pestaña de cuentas tras seleccionar un usuario existente

La vista de "Detalles Usuario" contiene un formulario con los siguientes campos: el rol (1), el cual no se puede modificar; el nombre de usuario (2) y la contraseña (3).

En el campo de rol y de usuario se pueden comprobar los valores actuales para que el administrador tenga esa información como referencia. Finalmente, la contraseña actual no se muestra por motivos de seguridad.

El formulario permite cambiar los campos de tres formas: permite cambiar sólo la contraseña, sólo el nombre de usuario, o ambos campos a la vez.

También se implementan varias comprobaciones una vez que se termine el formulario, evitando que el nuevo nombre de usuario se actualice si ya existe otro usuario con ese nombre, o evitando que se introduzca una contraseña débil sin el formato correcto.

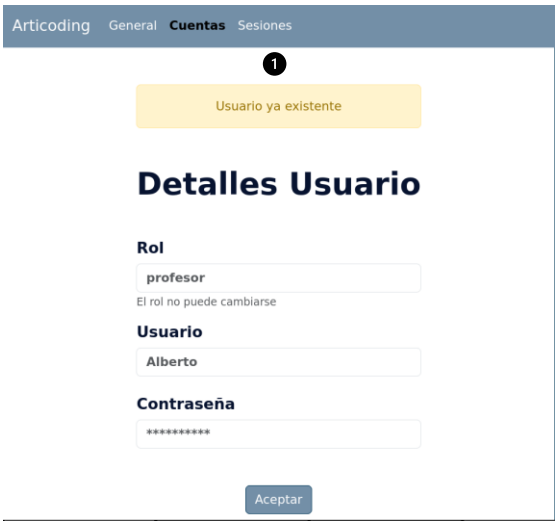

Figura 4-23: Error del formulario al intentar crear un usuario ya existente

Figura 4-24: Error del formulario debido al formato de la contraseña

Como podemos comprobar si el formulario no pasa los criterios de validación se cargará de nuevo la vista mostrar un mensaje (1, 2) explicando el motivo por el cual no se ha podido modificar el usuario, según se puede observar en las figuras 4-23 y 4-24.

Si el usuario vuelve a la lista de usuarios y selecciona el botón "crear nuevo usuario" se cargará la siguiente vista ilustrada en la figura 4-25:

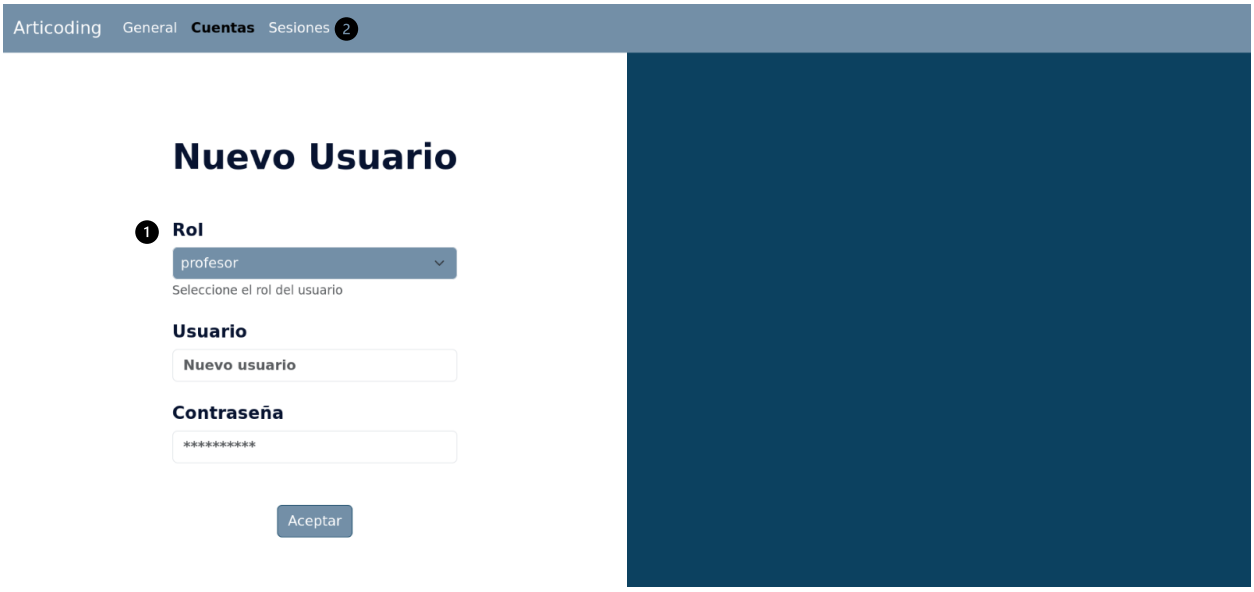

Figura 4-25: Vista para crear un nuevo usuario

La vista de nuevo usuario es similar a la anterior, la diferencia principal radica en que el usuario es creado desde cero, permitiendo la selección del rol mediante un desplegable (1) con la lista de roles disponibles.

La validación del formulario es la misma que la del formulario de edición, no permite introducir nombres existentes ni contraseñas que no tengan un formato válido.

Por último, si el usuario selecciona la vista de sesiones (2) se encontrará con la siguiente vista tal y como se muestra en la figura 4-26:

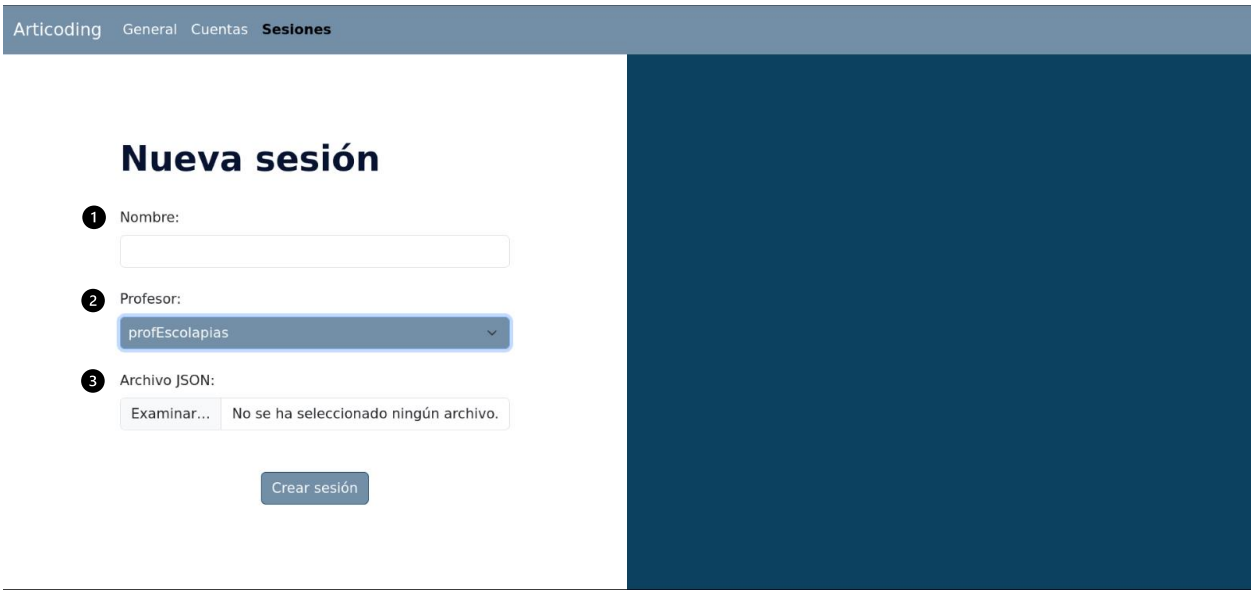

Figura 4-26: Pestaña de sesiones del administrador

El formulario de "Nueva sesión" contiene tres campos: un nombre (1), un profesor existente, el cual es seleccionado desde un desplegable (2) con todos los usuarios con rol de profesor, y un selector de archivos (3) locales.

A la hora de implementar el selector de archivos se han tenido varias cosas en cuenta: solo se permite la selección de ficheros que tengan un formato JSON, el archivo subido se renombrará con un nombre genérico, la información introducida en el formulario se insertará en la base de datos creando una nueva sesión y, finalmente se insertará el fichero en la dirección que contenga los datos de las sesiones, creando una nueva carpeta utilizando el nuevo id de la sesión como campo.

Insertando un nuevo experimento, como queda reflejado en las figuras 4-27 y 4-28, incorporaríamos las trazas del nuevo experimento con ID=4 en la carpeta de datos, la cual contiene información de las diferentes sesiones.

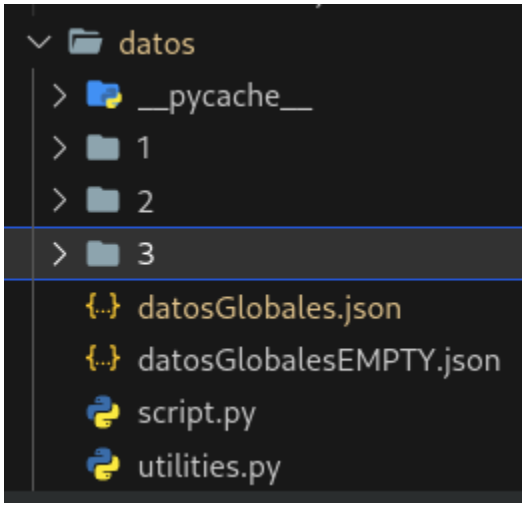

Figura 4-27: Esquema de archivos con tres experimentos

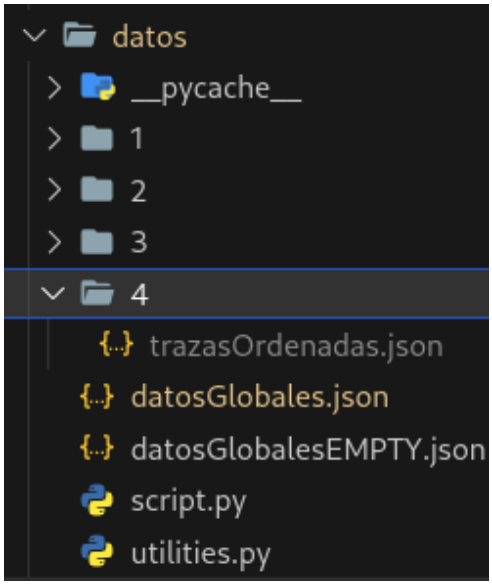

Figura 4-28: Esquema de archivos tras crear un cuarto experimento

4.4 Datos recogidos y tratamiento

El Reglamento General de Protección de Datos (RGPD) es un reglamento europeo encargado de poner normas a los datos que recogen las empresas sobre sus usuarios. Todo servicio que se ofrece dentro de la Unión Europea debe cumplir con estas normas dependiendo de que datos se recoja y que uso se les dé.

En nuestra página web no recopilamos ningún dato de carácter personal ni hacemos ningún tipo de tratamiento de datos. Lo único que almacenamos en nuestra base de datos son nombres de usuario y contraseñas, cifradas e introducidas por los propios usuarios y que sirven para iniciar sesión en el servicio. Aparte de esos datos, también guardamos un ID de sesión para que los usuarios puedan navegar por la web y volver a entrar a esta sin necesidad de tener que iniciar sesión. Este ID de sesión es la única cookie que almacenamos y es estrictamente necesaria para el funcionamiento de la página web y para dar un servicio solicitado por el usuario, siguiendo las pautas dictadas en [25]. En consecuencia, no es necesario solicitar el consentimiento de los

usuarios para almacenar la sesión, ya que, además, no usamos la sesión para rastrear la actividad de estos usuarios ni obtener ningún tipo de información.

Como no tratamos con datos personales no es necesario que cumplamos con todas las disposiciones legales del Reglamento General de Protección de Datos (RGPD). El RGPD es el reglamento europeo encargado de regular el tratamiento de datos de personas físicas y su circulación dentro de la Unión Europea. Aun así, debemos cumplir con ciertas obligaciones. Debemos incluir una política de privacidad que informe sobre qué información se recopila y qué uso se le da. Debemos también aplicar las medidas de seguridad necesarias para garantizar la seguridad de los datos, impidiendo accesos no autorizados y cifrando las contraseñas. Por último, tenemos también que poner a una persona encargada a disposición de las consultas o reclamaciones que puedan tener los usuarios.

Dentro del servicio que ofrecemos no guardamos ningún tipo de dato que pueda identificar a los participantes de las sesiones de Articoding. Al tratarse de menores, en la mayoría de los casos, sólo almacenamos datos demográficos de estos. Estos datos son su edad y alguna pregunta relacionada con la programación y el juego Articoding. No almacenamos nombres de los participantes. Cada participante tiene un id propio que no tiene ninguna relación con su nombre o con ningún dato suyo. Esta seudonimización de datos se hace en origen, cuando se generan las trazas, por lo que en ningún momento se conoce alguna información de los participantes que puedan identificarlos.

Capítulo 5 - Evaluación con usuarios

Con el objetivo de verificar y validar que la página web mostraba información entendible y útil para una persona que quisiese acudir a los diferentes análisis que se muestran en ella, recurrimos a dos voluntarios, los cuales se dedicarían a probar la página y comprobar que toda la información que se fuera mostrando se entendía y les ayudaba a comprender qué fallos habían cometido los alumnos de su clase. Cabe destacar que estos voluntarios son profesores de programación interesados en el juego para utilizarlo con sus alumnos en un curso 0 de una carrera de programación.

5.1 Metodología

La herramienta con la que se va a realizar la evaluación es un cuestionario de Google Forms que se encuentra en el Apéndice B, el cual se le requirió al voluntario rellenar al terminar para que nos permitiese conocer qué aspectos le habían parecido buenos o mejorables y saber si era capaz de encontrar determinada información dentro de la página web. Con este motivo, el formulario se dividió en tres secciones, siendo la primera la que contiene preguntas relacionadas con la demografía del encuestado, como su rango de edad, sus estudios o si había probado Articoding anteriormente.

La segunda sección incluye declaraciones que permiten al encuestado mostrar si está de acuerdo o no con ellas según la escala de calificación Likert, la cual es muy usada debido a que permite al encuestado valorar su respuesta en vez de simplemente responder con un sí o no. De este modo, las posibles respuestas a cada declaración se situaban en un rango entre 1 y 5, y a cada valor se le asignó una opinión siguiendo esta escala: 1-Muy en desacuerdo, 2-Algo en desacuerdo, 3-Ni de acuerdo ni en desacuerdo, 4-Algo de acuerdo, 5-Muy de acuerdo. Asimismo, aparte de las numerosas declaraciones, se incluyeron un par de preguntas de respuesta libre para que se pudiese dar una opinión o ser más específico en alguna cuestión relevante [26].

Por último, la tercera sección estaba formada por preguntas de respuesta corta donde el encuestado debía encontrar datos concretos en la página web para así comprobar que toda la información es accesible y fácil de encontrar.

Para comenzar con las evaluaciones, se requirió a los voluntarios iniciar sesión y realizar la prueba. Durante este proceso, se les pidió que hablaran en voz alta mientras revisaban la información que se les presentaba en la página web. A través de esta interacción, los voluntarios comentaron diversos aspectos relacionados con la maquetación o la generación de tablas y gráficas, identificando posibles mejoras que podrían implementarse.

5.2 Primera Evaluación

La persona seleccionada estuvo presente en un experimento usando Articoding con unos participantes el día 28 de marzo. La reunión con el voluntario se produjo el día 17 de abril en uno de los despachos de la Facultad de Informática, lo cual también le permitió obtener y analizar de primera mano los resultados de los participantes unos días atrás. Mientras revisaba la plataforma, comentó una serie de mejoras que se podían realizar.

Nada más entrar en la aplicación comentó varias posibles mejoras a nivel general. Primero comentó que en la barra superior es recomendable resaltar la página en la que nos encontramos, otra opción podía ser la implementación de un pie de página. También comentó que era mejor mantener los nombres mismos nombres que hay para cada categoría en el juego, es decir cambiar elementos como "tutorials" a "introducción", además de resaltar algún fallo ortográfico.

En la vista de alumnos nos recomendó cambiar el nombre de alumnos a participantes, ya que el experimento podría realizarse fuera de un entorno escolar. Además, nos recomendó que numeráramos la tabla de alumnos.

En la vista de categorías nos recomendó incluir el número de usuarios que había completado cada nivel, ya que las otras métricas no lo mostraban, siendo varias medias.

En "Comparativa general", debido a que no había unidades de medida, nos sugirió que las incluyéramos (segundos, número de intentos, etc.). Además, nos comentó que podía ser buena idea mostrar todas las vistas de categorías en una misma vista una seguida de otra en formato vertical, aunque fuera un cambio más secundario.

La vista que más cambios necesitaba fue la de errores. Para empezar el nombre era incorrecto, ya que nuestra idea era mostrar no excepciones en el código (lo que se suele entender cómo errores), si no detectar imperfecciones en el código y sugerir posibles mejoras, con lo cual nos recomendó que renombráramos la vista como "mejoras".

También nos recomendó incluir un texto descriptivo que explicara que datos representan las gráficas. También nos recomendó separar la vista actual en dos, una con los errores de un jugador concreto y otra con errores a nivel global, con los fallos más comunes, categorías con más fallos, etc.

Otros posibles cambios en la vista de errores eran la inclusión de funcionalidades nuevas, como mostrar los usuarios que cometieran más errores explicando por qué, o conocer que estrellas son más difíciles de conseguir analizando los errores cometidos en ese nivel. También nos recomendó cambiar el botón de errores detallados de un usuario ya que el diseño actual (un ojo) no era muy intuitivo e incluir gráficas que complementaran a las tablas.

En la vista detallada de errores nos recomendó enumerar los errores, cambiar el nombre de varios errores, ya que no estaban bien explicados, además de incluir un texto descriptivo de cada tipo de error, e incluir filtros que pudieran permitir al usuario buscar elementos en la tabla y reordenarla de diferentes maneras (número de errores, último nivel conseguido, etc.

Otras implementaciones posibles para la vista detallada eran la inclusión de gráficas que mostraran el número de fallos cometidos en cada nivel por el usuario e incluir múltiples textos descriptivos, por ejemplo, en el selector de categorías.

Tras la realización del cuestionario, nos recomendó hacer algunos cambios en el mismo, como cambiar algunos nombres, como, por ejemplo, cambiar alumnos por participantes o "grupos de niveles" por "categorías de niveles", además de eliminar alguna pregunta redundante.

Después de analizar los datos recogidos en el formulario, podemos conocer que su edad está en el rango de 35 a 44 años, que su nivel de estudios es universitario, centrados en la Informática, y que ya había jugado a Articoding.

Gracias a las respuestas respecto a la usabilidad, se ha obtenido una media de 4.08, lo que nos indica que en general la página web es intuitiva y fácil de interpretar. Para este cálculo se ha tenido en cuenta que un 1 en las declaraciones "La herramienta me desborda con demasiada información" y "Echo de menos algún tipo de análisis en la página" se interpretan como un 5 al estar formuladas en negativo. Más tarde, viendo individualmente cada una de las respuestas dadas, se ha comprobado que la mayor parte de la información recogida en la página web es de utilidad para el encuestado, dándole la mayor puntuación posible a los análisis sobre el tiempo e intentos medios de los jugadores para superar cada categoría. Además, está algo de acuerdo en que la vista general de participantes le ayuda a evaluar individualmente a cada uno, que los jugadores mostrados en cada página son los adecuados, y que sabe cómo se puede cambiar el nombre de estos.

Respecto a la pestaña de categorías, expone que está algo de acuerdo en que conocer el tiempo, intentos y estrellas medios de los participantes por cada nivel le ayudan a conocer qué problemas pueden tener y en que sabe cómo filtrar por cada categoría de niveles. Asimismo, indica que los boxplots que representan la distribución de tiempo empleado por los jugadores le son muy útiles. El aspecto en el que no está muy de acuerdo es en que los puntos que están más apartados en los gráficos le ayudan a la hora de detectar quienes han tenido más dificultades en niveles concretos.

Las respuestas sobre la pestaña de comparativa nos permiten observar que está de acuerdo en que es capaz de ver las diferencias entre los resultados de su grupo de participantes y los datos globales de todos los experimentos realizados y que esto mismo le parece de utilidad. Sin embargo, responde que no está de acuerdo ni en desacuerdo con que entienda los datos proporcionados por las gráficas de esta pestaña.

En las declaraciones de la pestaña de errores podemos observar que está muy de acuerdo con que le parece útil poder ver en la tabla de todos los participantes los errores concretos que ha tenido cada uno, además de que es capaz de ver quien necesita más ayuda con cada tipo de error, pareciéndole esto también de utilidad. No obstante, en las respuestas comenta que no tiene muy claro cómo identificar qué niveles son los que tienen más errores ni cómo filtrar por los tipos de errores de esta pestaña.

Por último, se le hicieron unas preguntas más generales con el fin de obtener su opinión sobre la página web en completo. Sus respuestas nos permiten saber que le parece razonable el tiempo de carga al iniciar sesión, que la página web es intuitiva, con el detalle de añadir información que permita saber en qué página se encuentra en cada momento, y que se la recomendaría a otros profesores que enseñan programación. Asimismo, está muy de acuerdo en que la herramienta le ayuda a identificar los problemas con los que los jugadores tienen más complicaciones, que le gustaría poder usarla como ayuda en sus clases y que no le desborda con demasiada información. Sin embargo, comenta que echa de menos algunos análisis en la página web, como las estrellas no conseguidas, el tiempo empleado en cada nivel, si hay inactividad, el uso de manuales o el tiempo de visualización de los mensajes de ayuda.

Respecto a las respuestas obtenidas en la tercera sección, todas las respuestas son correctas, por lo que podemos entender que la información que se muestra en la página web es accesible y fácil de encontrar.

Teniendo en cuenta las respuestas dadas por parte del encuestado y sus comentarios en voz alta, se han modificado los siguientes aspectos en el formulario para modificarlo y usarlo en las próximas evaluaciones. Primero se eliminó la declaración "Tengo claro cómo filtrar por grupos de niveles" repetida en las pestañas de Categorías y Comparativa, dejando solo la situada en la primera de ellas. Después se modificó la declaración "Tengo claro cómo filtrar por grupos de niveles" y pasó a ser llamada "Tengo claro cómo filtrar por categorías de niveles" para así tener cohesión con la página web. Finalmente, se eliminaron todas las referencias a "alumnos" en las preguntas o declaraciones, pasando a ser "jugadores" o "participantes".

La página web también sufrió unos cambios para conseguir una mejor comprensión y experiencia del usuario. Incorporamos varias funcionalidades nuevas, por ejemplo, incluimos una página de selección de experimentos (sesiones) después de la vista de inicio de sesión, ya que un mismo profesor puede participar en varios experimentos. En la tabla de categorías incluimos una pestaña que mostraba cuantos jugadores habían completado cada nivel.

También añadimos una barra de búsqueda en la tabla de participantes, lo cual le permite filtrar por nombre e id. En la misma tabla incluimos la opción de poder ordenar

a los participantes según su nombre o último nivel jugado. Realizamos mejoras visuales y de la interfaz, como añadir migas de pan (breadcrumbs) en todas las páginas, marcando en la barra de navegación la vista que se esté mostrando, lo cual facilita a la navegación de la página web. Además, cambiamos los botones de la vista de errores a unos más intuitivos.

Asimismo, incluimos nueva información para ayudar a los usuarios a entender mejor los datos mostrados. En la pestaña de errores enumeramos los errores de cada jugador, cambiamos el nombre de "errores" a "avisos", ya que se acercaba más los conceptos que queríamos describir. También arreglamos varias erratas, acentos e incluimos múltiples explicaciones de conceptos en cada una de las vistas principales.

5.3 Segunda Evaluación

La segunda evaluación se realizó el día 26 de abril en uno de los despachos de la Facultad de Informática. Todos los cambios mencionados para modificar en el anterior apartado ya fueron incluidos al momento de realizar esta evaluación. Al evaluador le fue requerido rellenar, de misma manera que para el anterior, el Google Forms de tres secciones listado en el apartado anterior, realizando los cambios pertinentes también presentados para mejorar cohesión entre la página web y el propio formulario.

Se comenzó de igual forma que en la anterior evaluación, pidiendo al encuestado iniciar sesión en la página web y hablase en voz alta mientras iba repasando la información que se le mostraba. Mientras llevaba a cabo esto, comentó numerosas mejoras que se podían realizar, las cuales se comentan a continuación.

El primer cambio que nos recomendó el evaluador fue la inclusión del logo de Articoding en la web, para ayudar a asociar la aplicación al juego. Además, nos recomendó que incluyéramos una pequeña descripción de la web en la página inicial y que implementáramos información visual, para ayudar en la navegación de la web.

Para conocer mejor qué niveles son más difíciles nos recomendó que los indicáramos en la vista. Además, el concepto de "intentos" no era muy claro y nos sugirió que incluyéramos una explicación del concepto. Otro concepto que podía ser

explicado al usuario fue el significado de cada una de las diferentes categorías, incluyendo un pop up con sus descripciones.

Otra recomendación que nos hizo el evaluador fue la modificación del almacenamiento y lectura de mensajes en la aplicación, almacenándolo en documentos JSON, esto mantendría el código más limpio y permitiría implementar diferentes traducciones del contenido de los mensajes utilizando internacionalización (I18N).

En la vista de comparativa general nos recomendó incluir el tamaño de las muestras, ya que mejoraba mucho la interpretación de los datos al incluirlo. También nos recomendó que incluyéramos unidades de medida para todos los elementos representados en las gráficas, además de incluir un título descriptivo de la función que cumple cada una. Un dato importante que resaltó fue la falta de información en la aplicación, como el tiempo jugado o qué niveles se han intentado

En la vista de avisos nos comentó que nos recomendaba incluir una descripción del concepto, además de incluir varias funcionalidades, como el número de fallos cometidos en cada categoría, pudiendo mostrarlos al pasar el ratón por encima de la tabla. También nos indicó que podría ser útil incluir el número de tarjetas no se habían ejecutado al mostrar el aviso "código muerto".

Además, nos recomendó incluir nueva información visual, como histogramas que pudieran mostrar hasta qué nivel ha llegado una clase y en qué cuartil se encuentra un usuario concreto, incluyendo en la tabla de participantes, junto con sus nombres, el número de interacciones realizadas y sus tiempos de juego. Todo esto comparándolo con la media de la clase, para conocer qué grado de implicación tuvo cada uno.

Para facilitar la visibilidad de la aplicación nos recomendó que utilizáramos un azul más claro. Sobre el formulario nos recomendó actualizarlo, ya que ya no teníamos una vista de "errores" sino de "avisos".

Después de la evaluación también decidimos realizar un compendio de todos los cambios pendientes en la memoria, para corregirlos y revisarlos, además de aumentar la frecuencia de nuestras reuniones.

Analizando los datos recogidos en sus respuestas al formulario, podemos observar que el evaluador pertenece al rango de entre 45 y 54 años de edad, su nivel de estudios es universitario, centrado en juegos serios y visualización de información y si ha jugado a Articoding previamente.

Respecto a la sección de usabilidad, se ha obtenido una media de 4.52 usando los mismos cálculos que la anterior vez, lo que refleja que la página web ha mejorado y que la información mostrada es aún más fácil de interpretar y accesible. Analizando las preguntas individualmente, le parece muy útil conocer tanto el tiempo como los intentos medios de los participantes. En la pestaña de Participantes indica que sabe cómo cambiar el nombre de estos y que el número de participantes en la tabla por página es el adecuado, pero no está de acuerdo ni en desacuerdo en que esta pantalla le permite evaluar individualmente a cada participante.

En la pestaña de Categorías se obtienen muy buenos resultados, ya que les da la mayor puntuación a las cuatro declaraciones, por lo que opina que conocer el tiempo, intentos y estrellas medias de los participantes le es útil, así como los gráficos de vela y sus outliers mostrados también en esta sección. También indica que tiene claro cómo filtrar por categorías de niveles. En las declaraciones de la pestaña de errores se vuelve a obtener la puntuación máxima en todas ellas, lo cual nos permite conocer que es capaz de ver las diferencias entre los participantes del experimento y los datos globales, expresando que esto mismo es de gran utilidad, y que entiende los datos que le proporcionan las gráficas.

Las respuestas sobre la pestaña de errores muestran que sabe cómo identificar qué niveles son los que más tienen errores, qué jugadores necesitan más ayuda con un tipo de error concreto y cómo filtrar por tipos de errores. Asimismo, opina que es útil conocer qué porcentaje de errores por nivel tienen los participantes y poder ver que errores han tenido cada uno de ellos.

Por último, las preguntas más generales nos permiten saber que está muy de acuerdo con que la herramienta le ayuda a identificar los problemas que tienen los participantes, que le gustaría poder usarla como ayuda en sus clases y que se lo recomienda a otros profesores que enseñan programación. Además, indica que la navegación es intuitiva, a pesar de ser mejorable, que no le desborda la cantidad de

información mostrada y que echa de menos en la página un análisis individual de los participantes pudiendo comparar sus resultados con sus compañeros.

Después de revisar las respuestas obtenidas en la tercera sección, todas las respuestas son correctas, por lo que deducimos que la información que se muestra sigue siendo tan accesible y fácil de encontrar como para el anterior evaluador.

Considerando las respuestas dadas por parte del encuestado y sus comentarios realizados en voz alta durante la evaluación, se comentarán a continuación los siguientes aspectos de la página web.

Cambiamos el nombre de las categorías para que se mostraran con el mismo nombre con el que aparecían en Articoding. También añadimos la columna "ver intentos" en la tabla de participantes para mostrar que habían realizado en cada uno. En esa tabla incluimos un filtro que ordenaba por tiempo jugado.

En todas aquellas gráficas que mostraban información de tiempo jugado se cambiaron las métricas de segundos, a horas, minutos y segundos.

En la vista de avisos incluimos una gráfica que mostraba la media de avisos cometidos por nivel, para dar más información que permitiera al usuario conocer la evolución de dificultad en cada categoría.

Finalmente, pasamos los textos del código a archivos JSON, permitiendo mantener el código limpio y aplicar I18N.

Capítulo 6 - Conclusiones y trabajo futuro

El presente Trabajo de Fin de Grado ha abordado la importancia de la programación y su enseñanza en el ámbito de la educación primaria y secundaria, así como el papel de los juegos serios como herramienta educativa. También se ha evidenciado la falta de plataformas para que los profesores puedan evaluar el conocimiento adquirido por los estudiantes durante el juego.

El desarrollo de la plataforma de evaluación utilizando Articoding y la implementación de la plataforma web han demostrado ser un paso importante hacia la mejora de la evaluación en el contexto de los juegos serios de programación. Se ha logrado proporcionar a los profesores plataformas para analizar y visualizar los datos generados durante el juego, permitiéndoles obtener una visión clara del progreso y desempeño de sus estudiantes.

Además, las evaluaciones realizadas han sido fundamentales para identificar áreas de mejora y realizar ajustes en la plataforma con el objetivo de lograr una experiencia intuitiva y efectiva para los usuarios. A partir de estas evaluaciones, se han extraído conclusiones relevantes que orientan el desarrollo de futuras plataformas de análisis de juegos serios para la enseñanza de la programación.

En base a la retroalimentación recibida, se han identificado algunas características clave que una plataforma de análisis de juegos serios para programación debería cumplir. En primer lugar, es fundamental que la plataforma permita recopilar y visualizar de manera clara y concisa los datos relevantes del juego, como el progreso individual de los estudiantes, las decisiones tomadas durante el juego y el rendimiento en diferentes áreas de conocimiento. Esto proporciona una visión completa y detallada del desempeño de los alumnos, facilitando la evaluación y el seguimiento de su aprendizaje. Para obtener esto habría que realizar más análisis y aplicar técnicas de aprendizaje automático para obtener comportamientos y poder dar unos resultados con menos necesidad de interpretación.

En cuanto a la extensibilidad y adaptabilidad de la plataforma, es importante considerar la posibilidad de que el juego de programación cambie o se quiera aplicar

la plataforma a otro juego similar. En caso de que se añadieran nuevos niveles o categorías, no haría falta modificar nada, ya que esa información se obtiene de las propias trazas. No obstante, si se añadiesen nuevas funcionalidades que modificasen los tipos de trazas ya existentes reflejadas en la tabla del Apéndice A, si sería necesario modificar la infraestructura de la plataforma para que las tuviese en cuenta, por ejemplo, nuevos campos dentro del objeto resultado al completar un nivel. Por otra parte, si se quisiera usar esta plataforma para otro juego de programación habría que asegurarse de que el juego tenga una estructura de las trazas similar a las de Articoding. Utilizando el formato xAPI esta estructura es muy parecida y deberían de poder hacerse los análisis básicos. El análisis de los errores en el código probablemente no pudiese visualizarse ya que otro juego no tiene por qué seguir la misma estructura ni tener este concepto.

Para evitar errores y garantizar la calidad de los datos recopilados y mostrados, es esencial seguir estándares y buenas prácticas establecidos en la industria. Una plataforma debe asegurar la coherencia y confiabilidad de los datos, aplicando filtros y validaciones adecuadas para garantizar su integridad. También se debe evitar la sobreexposición o interpretación errónea de los datos, presentándolos de manera clara y comprensible, evitando información redundante o innecesaria que pueda generar confusión. En nuestro caso, hemos seguido los estándares y buenas prácticas recomendados por expertos en el campo de la recolección y visualización de datos, tales como validación de datos, utilización de nombres descriptivos y significativos para las variables, cifrado de contraseñas o el uso de principios de diseño de experiencia de usuario, como la prevención de errores o la consistencia con otras plataformas. Al seguir estos estándares, buscamos asegurar la precisión y la relevancia de los datos presentados en la plataforma.

Asimismo, este trabajo ha sentado las bases para futuras líneas de investigación y desarrollo en el campo de la evaluación del conocimiento adquirido mediante juegos serios de programación. A continuación, se presentan algunas áreas de trabajo futuro que podrían ser exploradas para mejorar y ampliar el alcance de este proyecto:

1. Mejora de la plataforma web: Aunque se ha desarrollado una plataforma web funcional para la visualización de los datos y análisis ya explicados, existen

oportunidades para mejorar su interfaz de usuario, agregar nuevas funcionalidades y optimizar su rendimiento. Esto podría incluir la implementación de características de personalización, la integración con otras plataformas de evaluación o la mejora de la capacidad de visualización de datos en tiempo real.

2. Ampliación de fuentes de datos: Además de Articoding, existen otros juegos serios de programación disponibles en el mercado. Sería interesante investigar y adaptar estos juegos para utilizarlos como fuentes adicionales de datos en el proceso de evaluación. Esto permitiría comparar los resultados obtenidos con diferentes juegos y evaluar su efectividad en la enseñanza de la programación. Cabe resaltar que, lamentablemente, la mayoría de estos juegos no cuentan con código libre que permita su modificación, lo cual limita la posibilidad de adaptarlos según nuestras necesidades.
3. Análisis de datos avanzado: Si bien se han implementado herramientas de análisis y visualización de datos en este proyecto, se pueden explorar técnicas más avanzadas para obtener información más detallada sobre el desempeño de los estudiantes. Esto podría incluir el uso de algoritmos de aprendizaje automático para identificar patrones de comportamiento, realizar análisis predictivos o detectar áreas de mejora específicas para cada estudiante.
4. Evaluación de la efectividad pedagógica: Además de evaluar el conocimiento adquirido por los estudiantes, sería beneficioso realizar investigaciones para evaluar la efectividad pedagógica de los juegos serios de programación en general. Esto podría involucrar estudios comparativos con métodos de enseñanza tradicionales, análisis de retroalimentación de los estudiantes y recopilación de datos cualitativos para comprender mejor el impacto de estos juegos en el aprendizaje.
5. Análisis en tiempo real: Nuestra aplicación necesita que el administrador introduzca un archivo con las trazas para poder realizar los análisis. Una ampliación bastante interesante podría ser realizar estos análisis en tiempo real mientras se lleva a cabo una sesión. Se podrían analizar los datos cada pocos minutos, ya que estos tardan al menos 15 segundos en analizarse, y actualizar la

página dinámicamente. Lo más complicado de esto sería poder conectar el servicio que recoge los datos de Articoding con el nuestro.

Introduction

Motivation

Programming has become an increasingly relevant skill in today's society. With the advancement of technology, the ability to program has become essential in various fields, from task automation and simplification to the development of software and innovative solutions. For this reason, teaching programming at early stages of education, such as primary and secondary school, has gained increasing importance.

Moreover, it is important to note that programming fosters cognitive and critical thinking skills in students. Through programming, students are encouraged to develop logical and algorithmic thinking as they learn to break down complex problems into smaller steps. Additionally, programming promotes creativity, intellectual competence, problem-solving, collaboration, and decision-making skills, which are key in today's digital society [1].

In this context, video games have emerged as an effective educational tool for teaching programming. Known as "serious games," these games leverage the dynamics of video games to motivate students and provide them with an interactive and playful environment for learning. These games allow students to acquire programming knowledge while having fun and actively participating in the learning process.

However, teachers who want to use video games as educational tools face challenges that can discourage their use. One of these challenges is the difficulty of knowing what happens while students play. Teachers may struggle to monitor the individual progress of each student that the game transfers. The lack of adequate monitoring tools to assess the knowledge acquired during the game hinders the implementation of the tool, as well as the feedback and accurate evaluation of student learning by the teacher.

Goals

The main objective of this work is to provide effective platforms for evaluating the knowledge acquired by players participating in serious games focused on programming.

To achieve this objective, the study and development of data analysis and visualization platforms will be addressed, using Articoding as a case study. Articoding is an open-source game that utilizes standards for data collection.

The specific subobjectives pursued are as follows:

1. Understanding Articoding and its context of use: The first subobjective focuses on acquiring a deep understanding of the serious game Articoding, a work carried out by students from another year [2] and [3] with the aim of teaching basic programming concepts. This task involves understanding its context of use in programming education. The mechanics and functioning of the game, the programming concepts addressed, and the learning objectives pursued through its use will be investigated. This knowledge will allow an understanding of the game's characteristics and the process it follows to address its educational objective.
2. Learning xAPI: The second subobjective focuses on learning and mastering the xAPI (Experience API) standard. xAPI is a protocol that enables data collection on player interactions in games and stores them in a structured format. By studying xAPI, the data collection and structuring of Articoding can be understood, as it utilizes this standard. This will facilitate the subsequent analysis and evaluation of player data.
3. Developing data analysis and visualization tools: The third subobjective involves the development of data analysis and visualization tools based on the interaction data provided by Articoding and its implementation of xAPI. These tools will allow for a detailed examination of the actions and decisions made by players during the game. Data analysis techniques, such as those used in [4], will be explored, and effective visualizations will be employed to extract relevant information for teachers. These tools will serve to evaluate the knowledge acquired by players in a more precise and efficient manner. The main objective of this work is to provide effective platforms for assessing the knowledge acquired by players participating in serious games focused on programming. To achieve this goal, the study

and development of data analysis and visualization platforms will be addressed, using Articoding as a case study, an open-source serious game that uses standards for data collection [2] [3].

Work plan

The work plan for the development of this project was divided into several stages, addressing each of the previously stated subobjectives. Firstly, we began by playing Articoding to familiarize ourselves with its mechanics. During this stage, we identified the relevant information for a teacher when evaluating students' knowledge.

In the second stage of the work plan, we focused on acquiring a deeper understanding of the xAPI standard, familiarizing ourselves with key principles and concepts such as statements, actors, verbs, and objects. Additionally, we analysed the information generated by the game in this format and its representation in JSON format. We also conducted a comprehensive study of similar platforms used for evaluating learning in serious programming games. This research allowed us to better understand the features and functionalities that such platforms offer, as well as identify best practices and aspects to consider in the development of our own platform.

As the third stage, we collected user interaction data with the game and focused on creating a web platform where teachers could access and visualize the generated analyses. This platform would enable teachers to view different game sessions conducted with their students and access specific analyses, graphs, and tables with relevant session information. The goal was to provide an intuitive and user-friendly interface that would give teachers a clear view of their students' progress and performance.

After completing the implementation of the web platform, the final stage involved conducting user evaluations. For this purpose, we invited two university professors who had previously used the Articoding game to test the developed platform. During these evaluations, the professors used the platform and provided valuable feedback and suggestions for improvement. We carefully analysed their comments and decided to implement the changes and details that we deemed most necessary and beneficial for the functioning and user experience.

The work plan and each of its stages were approached iteratively. As we progressed in each stage, we reviewed and improved the work done so far to ensure that the platform met the established requirements. This iterative approach allowed us to make continuous adjustments and enhance the functionality and usability of the platform.

Conclusions and future work

The present Bachelor's Thesis has addressed the importance of programming and its teaching in primary and secondary education, as well as the role of serious games as an educational tool. It has also highlighted the lack of platforms for teachers to assess the knowledge acquired by students during the game.

The development of the evaluation platform using Articoding and the implementation of the web platform have proven to be significant steps towards improving assessment in the context of serious programming games. It has successfully provided teachers with platforms to analyse and visualize the data generated during the game, enabling them to gain a clear understanding of their students' progress and performance.

Furthermore, the conducted evaluations have been crucial in identifying areas for improvement and making adjustments to the platform to achieve an intuitive and effective user experience. Relevant conclusions have been drawn from these evaluations, guiding the development of future platforms for analysing serious games for programming education.

Based on the received feedback, several key characteristics that a platform for analysing serious programming games should fulfil have been identified. Firstly, it is essential for the platform to allow for the collection and clear visualization of relevant game data, such as individual student progress, decisions made during the game, and performance in different knowledge areas. This provides a comprehensive and detailed view of student performance, facilitating evaluation and monitoring of their learning. To achieve this, further analysis and the application of machine learning techniques would be necessary to extract behaviours and provide results with less need for interpretation.

Regarding the extensibility and adaptability of the platform, it is important to consider the possibility of the programming game changing or applying the platform to another similar game. If new levels or categories were added, no modifications would be necessary since that information is obtained from the game logs. However, if new functionalities were added that modified the types of existing logs reflected in the table

of Appendix A, it would be necessary to modify the platform's infrastructure to accommodate them, for example, new fields within the result object when completing a level. On the other hand, if this platform were to be used for another programming game, it would be necessary to ensure that the game has a log structure similar to that of Articoding. Using the xAPI format, this structure is very similar, and basic analyses should be possible. The analysis of code errors probably could not be visualized since another game may not follow the same structure or have this concept.

To avoid errors and ensure the quality of the collected and displayed data, it is essential to follow industry-established standards and best practices. A platform should ensure the coherence and reliability of the data by applying appropriate filters and validations to guarantee its integrity. Overexposure or misinterpretation of data should also be avoided by presenting it in a clear and understandable manner, avoiding redundant or unnecessary information that may generate confusion. In our case, we have followed the standards and best practices recommended by experts in the field of data collection and visualization, such as data validation, the use of descriptive and meaningful variable names, password encryption, or the application of user experience design principles, such as error prevention or consistency with other platforms. By adhering to these standards, we aim to ensure the accuracy and relevance of the data presented in the platform.

Future Work

Likewise, this work has laid the foundation for future lines of research and development in the field of assessing knowledge acquired through serious programming games. Here are some areas for future work that could be explored to improve and expand the scope of this project:

1. Improvement of the web platform: Although a functional web platform has been developed for data visualization and analysis as explained earlier, there are opportunities to enhance its user interface, add new functionalities, and optimize its performance. This could include implementing customization features, integrating with other assessment platforms, or improving real-time data visualization capabilities.

2. Expansion of data sources: In addition to Articoding, there are other serious programming games available in the market. It would be interesting to research and adapt these games to use them as additional data sources in the evaluation process. This would allow comparing the results obtained from different games and assessing their effectiveness in teaching programming. It should be noted that unfortunately, most of these games do not have open-source code that allows for modification, which limits the possibility of adapting them to our needs.
3. Advanced data analysis: While data analysis and visualization tools have been implemented in this project, more advanced techniques can be explored to obtain more detailed insights into student performance. This could involve the use of machine learning algorithms to identify behaviour patterns, perform predictive analysis, or detect specific areas for improvement for each student.
4. Evaluation of pedagogical effectiveness: In addition to assessing students' acquired knowledge, it would be beneficial to conduct research to evaluate the pedagogical effectiveness of serious programming games in general. This could involve comparative studies with traditional teaching methods, analysis of student feedback, and collection of qualitative data to better understand the impact of these games on learning.
5. Real-time analysis: Our application currently requires the administrator to input a file with the logs to perform the analysis. An interesting extension could be to carry out real-time analysis during a session. The data could be analysed every few minutes, as it takes at least 15 seconds to analyse them, and dynamically update the page. The most challenging aspect of this would be to connect the service that collects data from Articoding with our platform.

CONTRIBUCIONES PERSONALES

David Gavilanes de Dios

Antes de comenzar el proyecto, revisé trabajos anteriores de asignaturas pasadas que también realizaban análisis de datos utilizando Python. Esto me permitió recordar ideas y no empezar desde cero, lo que me posibilitó contribuir desde el principio. Además, compartí estos trabajos con mis compañeros para que también pudiesen revisarlos y ver ejemplos.

En la reunión de presentación del proyecto, nuestros tutores nos asignaron una investigación inicial a cada uno. En mi caso, me encargué de realizar la lectura del libro referenciado como [10], para así empezar a entender la estructura de xAPI, que es la que usa Articoding para guardar sus trazas. También leí los anteriores TFG sobre Articoding, [2] y [3], para comprender como estaba creado, y jugué al propio juego para empezar a pensar que análisis se podían realizar. Estos posibles análisis los anoté en un documento compartido con mis compañeros para generar ideas de qué análisis útiles se podrían crear para la posterior plataforma web. Todas las referencias que usamos a partir de ese momento las guardé gracias al gestor de referencias Mendeley.

Después de su lectura y de que mis otros dos compañeros también acabaron sus tareas, redactamos cada uno un documento sobre el tema que habíamos investigado. El mío se corresponde con lo que ahora en la memoria es el apartado 2.2. Asimismo, revisé los otros documentos que generaban mis compañeros para comprobar que lo que habían escrito estaba bien redactado y tenía sentido. Esta revisión mutua por parte de los demás compañeros se realizó después para todos los documentos y código generado.

Después de la revisión por parte de los tutores, también agregué al documento anterior un apartado respecto a cómo se usa xAPI en Articoding (apartado 2.3), un apartado comparándolo con SCORM (apartado 2.2.8), el anterior estándar a xAPI, y varias imágenes para ilustrar un poco esas secciones que después se incluirían en la memoria.

Más tarde, comencé a generar código en mi propia rama del repositorio de GitHub para llevar a cabo los análisis que teníamos apuntados en el documento. Los primeros que realicé se centraron en obtener las estrellas que había obtenido un jugador al completar un nivel, verificar el valor de "first_execution" para comprobar si se había pasado el nivel a la primera y obtener cuántos intentos había necesitado el jugador para pasárselo. Cuando ya estuvieron listos, se pasaron a la rama de Ángel Hortelano Pérez, donde él, junto con otros análisis que tenía preparados, realizó otros más complejos usándolos.

Unas semanas más adelante, me revisé un documento sobre Articoding, en el que aparte de propiamente realizar la revisión, añadí información del propio juego, como varios párrafos que se encuentran en el apartado 2.1 y antes de comenzar el 2.1.1.

Después de esto, me centré en realizar un análisis más complejo sobre las tarjetas creadas, movidas y eliminadas por los jugadores en cada nivel, para así tener un ranking de los niveles donde los jugadores enfrentaron mayores dificultades al realizar acciones muy por encima de la media.

También escribí un documento sobre otros juegos y plataformas que empleaban xAPI, para lo cual tuve que leer distintos artículos científicos y páginas web, como las referenciadas en [13] y [14], para así también conocer la clasificación de juegos serios existente hoy en día.

Otro documento que redacté fue todo el apartado 2.6, que habla sobre otras plataformas similares a la que hemos desarrollado. Para ello, investigué sobre plataformas certificadas por ADL que fuesen tanto de código abierto como código cerrado, aunque tras no encontrar ninguna con su código fuente disponible libremente, los tres subapartados actuales se centran sobre plataformas de código cerrado.

Dado que veíamos necesario un análisis más complejo en la plataforma, decidimos realizar un clustering sobre los jugadores que nos pudiese indicar a que grupo pertenecía cada uno, de lo cual me encargué yo. Con el fin de llevarlo a cabo, probé distintas combinaciones de variables, como tiempo, edad y género o edad/género,

tiempo e intentos, pero no se obtenían clústeres de los que se pudiesen obtener información significativa. Fue entonces cuando empleé el PCA (Principal Component Analysis) para usar todos los análisis que ya quedan explicados en el Capítulo 3. Una vez que ya obtuve clústeres mejor definidos, fue cuando los incorporé a la plataforma web. Asimismo, el anterior análisis que realicé sobre las tarjetas fue descartado, pero en su lugar en la pestaña de Avisos se quería añadir un análisis que tenía en cuenta las acciones sobre las tarjetas en vez de realizar el ranking de niveles. No obstante, no fue incluido en la plataforma por falta de tiempo para poder implementarlo correctamente.

En relación con la plataforma web, los análisis que realicé al principio del proyecto se incluyeron junto a los de mis compañeros. Además, participe en la creación a nivel de código backend de algunas tablas que aparecen en las distintas pantallas, como la que muestra los tiempos, intentos y estrellas medios de cada categoría en la pestaña de Resultados generales y la que muestra información similar, pero de cada nivel en la pestaña de Categorías. También participé en el diseño en Figma de la plataforma web, así como permitir que fuese visible en dispositivos móviles usando Bootstrap para conseguir responsividad al tamaño de pantalla.

Asimismo, redacté los diferentes textos que se pueden visualizar en la plataforma, como el que explica qué se aprende en cada categoría o el que muestra qué se puede ver en un diagrama de vela (boxplot).

En cuanto a las evaluaciones realizadas, investigué cómo se deberían hacer las propias evaluaciones, qué preguntas hacer y cómo hacerlo, consultando artículos como el referenciado en [23]. Gracias a esto, pude redactar el Google Forms empleado y realizar las modificaciones necesarias entre una evaluación y la otra. Asimismo, como último documento, me encargué de redactar lo que se corresponde al Capítulo 5 de la memoria, donde muestro todo el proceso de evaluación seguido, los comentarios recibidos de los evaluadores y los cambios que se llevaron a cabo finalmente.

Finalmente, respecto a la memoria, me encargué de transferir todos los documentos a la plantilla de la memoria, organizándolos en el orden actual, y redactando los apartados que faltaban, como la Introducción, las Conclusiones y

trabajo futuro y la Bibliografía. También me encargué de traducir los apartados necesarios a inglés y de agregar títulos a las figuras y tablas.

Ángel Hortelano Pérez

Cuando los tutores nos presentaron el proyecto nos explicaron que íbamos a tener que hacer un análisis de datos utilizando Python. Lo primero que nos mandaron fueron un conjunto de trazas generadas en un experimento del año anterior, el juego Articoding y el TFG del año anterior para que nos fuésemos familiarizando con el contexto del TFG. Mientras estudiaba la estructura de las trazas y jugaba al juego para comprender como funcionaba también tuve que ponerme a aprender Python ya que para mí fue un lenguaje nuevo. Después de comprender el contexto de nuestro TFG los tutores nos pidieron que pensásemos en posibles análisis que podíamos sacar y entre los tres hicimos una lista de análisis.

Cree un repositorio en GitHub para poder compartir el trabajo con los compañeros y los tutores. Baltasar nos pidió también que fuésemos guardando todas las referencias y para ello nos recomendó usar Mendeley que, gracias a la UCM, tenemos licencia gratuita. Me puse a investigar cómo se instalaba y utilizaba para poder explicárselo a mis compañeros. Cree también un documento con los pasos necesarios para crearse una cuenta, instalarlo y guardar referencias. El uso de esta herramienta ha facilitado mucho el desarrollo de la memoria.

Empecé entonces a hacer análisis en Python. Empecé extrayendo los timestamps de inicio y fin de cada nivel de cada alumno. Extraje también las estrellas que habían obtenido y los intentos que habían empleado cada vez que iniciaban el nivel. Mientras realizaba esta extracción de datos fui redactando en un documento como lo había y hecho y que objetivos tenían estos análisis. Este documento fue finalmente incluido en la memoria en el apartado 3. Completé también el documento sobre xAPI para que tuviese toda la información necesaria para poder entenderlo.

Mientras iba realizando estos análisis y sus respectivos documentos los tutores nos recomendaron ir redactando algún documento más que se fuese a incluir en la memoria con el fin de llevar el trabajo al día y que no se hiciese pesado redactar la

memoria entera. Me puse entonces a redactar un documento sobre el juego Articoding, cómo funciona, que categorías de niveles tiene y que conceptos se introducen en cada una de ellas. Este documento fue incluido en la memoria.

Cuando terminaba de redactar un documento se lo pasaba a mis compañeros para que lo revisasen y completasen. Ellos, a su vez, me pasaban a mí los documentos que iban terminando para poder completarlos y corregirlo, como hice con el documento sobre xAPI, por ejemplo.

Mis análisis fueron avanzando y ya pasé a analizar los datos que había extraído. Comencé sacando la diferencia de tiempo entre los timestamps de inicio y de fin de cada nivel para sacar el tiempo empleado por cada alumno en cada nivel. Conseguí también sacar la traza de inicio y fin de juego con el objetivo de saber cuánto tiempo habían estado jugando los participantes. Con estos datos y los extraídos anteriormente comencé a realizar medias y a obtener los valores atípicos. Conseguí generar algunas gráficas utilizando la librería de Python *Pyplot*. Esta librería fue sustituida por Plotly más tarde ya que no me permitía poner valores para cuando el usuario pasase el ratón por encima ni exportarlo a un formato JSON de forma sencilla.

Teniendo estos primeros datos y gráficas generadas comencé a hacer el diseño de nuestra página web. Hice un proyecto en Figma donde cree un boceto con el estilo que quería darle a la página para mandárselo a los profesores y que me diesen el visto bueno [24]. El diseño de Figma no fue calcado para realizar la página web, pero el estilo y los colores sí que son los mismos. Teniendo el visto bueno de los tutores comencé a montar el servidor de Node.js. Instalé los módulos necesarios y cree la página de login. Generé también la base de datos que es utilizada por la página web. Cree también el sistema de sesiones para que el usuario no tuviese que estar constantemente iniciando sesión en la página.

Me he encargado de crear toda la parte de la web del usuario, el profesor. Cree el script que analiza los datos y genera todos los archivos JSON que son utilizados en la página. Hice también todas las vistas del profesor dentro de la página web utilizando Bootstrap 5.3 y haciéndolas responsive para que puedan visualizarse desde un terminal

móvil razonablemente bien. Comencé maquetando las páginas de: resultados, participantes, categorías y comparativa. Una vez estaba la estructura de la página hecha comencé a crear las rutas dentro del router profesor.js para traer todos los datos a la página y poder hacer las redirecciones. Me he encargado de hacer todas y cada una de las peticiones de profesor.js para hacer la redirección entre páginas y para dar respuesta a las consultas realizadas por AJAX para que la web fuese dinámica.

Habiendo ya creado las vistas principales con gráficas y tablas fui añadiendo funcionalidades a la página. Añadí que se pudiese asignar un nombre a los alumnos para que el profesor pudiese identificarlos y que este nombre apareciese desde ese momento cada vez que se hace referencia al alumno. Hice también las páginas dinámicas y que se pudiesen cambiar los gráficos según la categoría seleccionada sin necesidad de recargar la página. La página de alumnos todavía no tenía la opción de ver en detalle los intentos de cada alumno.

Según iba avanzando con esto los tutores nos recomendaron hacer algún análisis más. Fue ahí cuando comencé a analizar los códigos XML de los alumnos cuando terminaban un nivel para obtener que fallos habían cometido. Comencé añadiendo al script un análisis del código XML que detectaba cuando un jugador se había dejado código muerto y cuando un jugador no nombraba adecuadamente a una variable. Investigué el proyecto de Dr. Scratch [4] para obtener algunas ideas sobre análisis. Se me ocurrió entonces hacer un análisis sobre el uso de variables. Conseguí detectar cuando el usuario ha creado una variable de más y cuando no ha creado una variable que le hubiese ahorrado utilizar algún valor literal.

El análisis con los errores lo incluí en una nueva pestaña de la vista de profesor, la pestaña avisos. Hice el proceso de añadir el análisis al script, crear la vista y añadir las peticiones necesarias al servidor para conectar todo. Para añadir un poco más de información añadí la columna *Ver intentos* a la tabla de la vista de participantes. De esta forma el profesor podía ver en detalle todo lo que había hecho cada alumno durante la sesión.

Mientras iba probando la web y arreglando errores cree un documento sobre el reglamento general de protección de datos (RGPD) que me había recomendado Baltasar que redactase para incluirlo en la memoria final.

También he ayudado a completar el apartado sobre la web a mi compañero Pablo añadiendo las cosas correspondientes a mi parte y a la estructura general del proyecto y la base de datos.

Pablo Rabadán Arza

Al empezar el curso lo primero que hicimos fue tener una reunión con nuestros tutores en la que se nos explicó el objetivo de la aplicación que tendríamos que realizar y, qué referencias necesitábamos comprender antes de empezar a desarrollar la aplicación. Debido a que la aplicación iba a analizar trazas generadas durante experimentos en los que usuarios completaban el juego serio Articoding, lo primero que tuvimos que realizar fueron documentos que explicaran el funcionamiento del juego, sobre el estándar en el que se almacenaban las trazas (XAPI) y otro, de manera más específica sobre su aplicación en Articoding.

Lo primero que hice fue completar el juego mientras me iba familiarizando con su estructura (tipos de niveles, funcionamiento de las tarjetas, navegación por la aplicación, etc.). Paralelamente los tutores nos enviaron trazas de un experimento anterior realizado en mayo de 2022 para familiarizarnos con su estructura, además nos compartieron dos documentos que explicaban el funcionamiento y formato de las trazas y otro con análisis realizados en 2021.

Para poder comprender bien las trazas nos recomendaron que realizáramos análisis, con Python, intentando realizar análisis de tiempos. Debido a que no todos habíamos tenido contacto previo con Python me dediqué a buscar información y documentos que explicaran su estructura, más específicamente la aplicación de Python para la Ciencia de Datos en cuadernos de Jupyter.

Una vez obtenida suficiente información realicé un documento de Jupyter de Introducción a Python, al ser este un entorno interactivo pude realizar un índice con varios apartados y, en cada uno explicaba un concepto diferente del lenguaje de

programación, como los elementos básicos del lenguaje, estructuras de datos y lectura de ficheros.

Una vez realizado este documento, debido a que las trazas estaban contenidas en un documento Python y la información del código de cada intento se encontraba en formato XML realicé otro documento de Jupyter denominado "Formatos de datos para la web" en el que expliqué información sobre lectura y manipulación de documentos JSON, XML y CSV.

Mientras realizaba estos documentos, me dediqué a realizar análisis de tiempo con las trazas que nos habían compartido los tutores. Lo primero que tuve que realizar fue obtener los timestamps de inicio y fin de nivel, no sin antes ordenar las trazas por tiempo. Con esa información pude sacar el tiempo de juego de cada intento, agrupando los tiempos en categorías, luego en niveles y finalmente en usuarios. Esto me permitió poder obtener datos como el tiempo medio de cada nivel y el tiempo medio de juego, obteniendo también información de las estrellas.

Mientras realizaba estos documentos y análisis mis compañeros también realizaban análisis de tiempo con Python para familiarizarse con la aplicación y redactaron documentos sobre el uso de XAPI y Mendeley. Mientras avanzábamos en nuestros respectivos campos revisábamos los documentos de que íbamos redactando de manera colectiva, añadiendo información extra y corrigiendo errores.

Una vez obtenidos los datos de los tiempos empecé a realizar un documento sobre la estructura de los documentos XML, y más específicamente orientado a entender la estructura del código de cada intento. Paralelamente empecé a desarrollar análisis que permitieran obtener información del comportamiento de los jugadores mediante el análisis de los documentos XML. Para recorrer los intentos tuve que utilizar la librería ElementTree, una API que permite procesar documentos XML como árboles de búsqueda.

Antes tuve que separar las trazas en dos listas, una con los intentos completados y otra con intentos de niveles fallidos o incompletos. Una vez separados los datos analicé el número de tarjetas que contenía intento completado, el número de variables que había creado el jugador y cuántas veces había utilizado cada una.

Paralelamente hice la parte de administración de la web, inicialmente incluí una vista principal y la lista de usuarios, no sin antes crear con MySQL una base de datos con los usuarios, roles e institutos de la aplicación. Más adelante incluí las vistas detalladas de usuario, que permitía comprobar sus datos, como el nombre, rol e instituto asociado y, añadí un formulario de creación de usuario. Para realizar todo esto tuve que realizar un router admin.js que redireccionara las diversas vistas de usuarios e incorporar peticiones CRUD a mediante DAOs de profesor, y posteriormente los experimentos que obtuviera información y mostrara de forma dinámica la información de los formularios y luego pudiera insertar esa información en la Base de Datos.

Mientras fue avanzando el desarrollo de la aplicación fui incluyendo otras funcionalidades, como la posibilidad de modificar datos de usuarios actuales o la creación de una ventana llamada "rutas" en la que se mostraba la dirección de la carpeta con las trazas. También fui ampliando el documento XML explicando la estructura básica de los diferentes bloques de código que analizábamos, como las variables o los bucles.

Más adelante realizamos un experimento en el cual acudieron varios alumnos para probar la aplicación, y de paso rellenaron unos formularios que redactaron mis compañeros. Después del experimento tuvimos varias evaluaciones del funcionamiento de la aplicación y, creando un cuestionario para incorporar futuras mejoras. Con esta información creamos un documento e incorporamos todas las mejoras, para poco a poco ir mejorando las funcionalidades de la web.

Muchas de esas recomendaciones las fuimos aplicando. En mi caso incorporé avisos y validaciones en los formularios de usuario o la posibilidad de tener información visual en la barra de navegación, resaltando la vista en la que se encontraba el administrador. También

Uno de los últimos cambios realizados en la vista del administrador fue la sustitución de la vista de rutas por la de sesiones, pudiendo asociar unas trazas determinadas a un profesor, y subiendo ese documento al proyecto. Así se podían

seleccionar diversas sesiones en la vista del profesor asociadas a aquel profesor que haya iniciado sesión de forma dinámica.

Asimismo, redacté dos documentos, uno sobre las tecnologías utilizadas en la aplicación, su propósito en la misma y cómo se incorporaba dentro de la estructura general del proyecto y otro explicando las funcionalidades de la página web y su diseño, detallando la estructura de cada vista y cómo se muestran de forma dinámica las tablas y graficas en la vista del profesor o cómo se insertan nuevos usuarios en la base de datos y se suben nuevas sesiones a la ruta que contiene los datos del proyecto. También fui incorporando imágenes con las diferentes variaciones de la página web, como avisos o pop-ups.

Por último, revisamos la aplicación para eliminar y arreglar posibles excepciones, fallos visuales y, paralelamente incorporamos los diferentes documentos que redactamos a lo largo del curso en la memoria, revisando erratas, errores de puntuación, incluyendo referencias y adaptándolo a un formato deseado.

BIBLIOGRAFÍA

[1] B. Marcano Lárez, «Juegos serios y entrenamiento en la sociedad digital,» *Teoría de la educación : educación y cultura en la sociedad de la información*, vol. 9, nº 3, pp. 93-107, 2008.

[2] D. Faouaz Santillana, A. García Cárdenas y Á. Poyatos Morate, «Juegos Serios para Promover el Pensamiento Computacional y la Programación,» Trabajo Fin de Grado, Departamento de Ingeniería del Software e Inteligencia Artificial, UCM, Madrid, 2021.

[3] T. Duarte Balvís, A. Martín Sánchez y P. Martínez Martínez, «Uso de juegos serios para mejorar el aprendizaje de la programación en la escuela,» Trabajo Fin de Grado, Departamento de Ingeniería del Software e Inteligencia Artificial, UCM, Madrid, 2022.

[4] Dr. Scratch, 2022. [En línea]. Available: http://www.drscratch.org/. [Último acceso: 21 diciembre 2022].

[5] A. Vahldick, A. Mendes and M. Marcelino, «A review of games designed to improve introductory computer programming competencies,» *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, Madrid, Spain, pp. 1-7, 2014, doi: 10.1109/FIE.2014.7044114.

[6] M. Miljanovic and J. Bradbury, «A Review of Serious Games for Programming,» *Springer International Publishing*, vol. 11243, 2018.

[7] Skillsoft Percipio, «Percipio Integration Overview,» 2022. [En línea]. Available: https://documentation.skillsoft.com/en_us/pes/Integration/glossary.htm. [Último acceso: 17 octubre 2022].

[8] I.R. Chenoweth, J. Abril-García y I. Meza-Ibarra, «Adoptar nuevas tendencias de elearning xAPI y LRS,» *Revista de Ciencias de la Educación*, vol. 1, nº 2, pp. 63-73, 2017.

[9] T. Hennessey, «Uso de juegos serios para mejorar el aprendizaje de la programación en la escuela,» 2022. [En línea]. Available: https://evolve-sg.com/tracking-learner-engagement-and-data-in-xr-learning-experiences/. [Último acceso: 29 noviembre 2022].

[10] J. Laane Effron y S. Putman, Investing Performance Design and Outcomes with xAPI, BookBaby, ISBN 978-1-48359-629-7, 2017.

[11] Ispring, «¿Qué es xAPI (Tin Can)?,» 2021. [En línea]. Available: https://www.ispring.es/blog/xapi-tin-can. [Último acceso: 25 enero 2023].

[12] S. Serrano, «LXP: beneficios de utilizar esta plataforma en el proceso de aprendizaje de tus colaboradores,» 2022. [En línea]. Available: https://www.crehana.com/blog/gestion-talento/lxp/. [Último acceso: 25 enero 2023].

[13] DL, «LMS y LXP: ¿qué son y cómo funcionan?,» 2022. [En línea]. Available: https://www.dl.cl/recursos-blog/lms-y-lxp-que-son-y-como-funcionan/. [Último acceso: 25 enero 2023].

[14] C. López Raventós, «El videojuego como herramienta educativa. Posibilidades y problemáticas acerca de los serious games,» *Apertura, Revista de Innovación Educativa*, Vol. 8, nº 1, 2016.

[15] A. Bell, «uBlockly, » GitHub. [En línea]. Available: https://github.com/imagicbell/ublockly. [Último acceso: 25 mayo 2023].

[16] Bealink, 2021. [En línea]. Available: https://bealink.io/en/. [Último acceso: 21 diciembre 2022].

[17] Core LRS, 2020. [En línea]. Available: https://www.corelrs.com/. [Último acceso: 21 diciembre 2022].

[18] Veracity Learning, 2023. [En línea]. Available: https://lrs.io/home. [Último acceso: 21 diciembre 2022].

[19] J. Santhanavanich, «Python & Pandas in Node.js Application,» 2022. [En línea]. Available: https://towardsdatascience.com/python-pandas-in-node-js-application-47a76ae42a2a. [Último acceso: 15 febrero 2023].

[20] NIST, «What are outliers in the data?,» 2022. [En línea]. Available: https://www.itl.nist.gov/div898/handbook/prc/section1/prc16.htm. [Último acceso: 29 noviembre 2022].

[21] A. Atkinson, F. Ariza, and J. García-Balboa, «Estimadores robustos: una solución en la utilización de valores atípicos para el control de la calidad posicional,» *GeoFocus*, nº 7, pp. 171–187, 2014.

[22] Python Organization, «The ElementTree XML API,» 2022. [En línea]. Available: https://docs.python.org/3/library/xml.etree.elementtree.html. [Último acceso: 28 diciembre 2022].

[23] J. de Wu, «Análisis de Componentes Principales: Implementación en Python,» 2021. [En línea]. Available: https://blog.damavis.com/analisis-de-componentes-principales-implementacion-en-python/. [Último acceso: 24 abril 2023].

[24] JQueryScript.net, «Create Dynamic CRUD Bootstrap Tables With BSTable Plugin,» 2021. [En línea]. Available: https://www.jqueryscript.net/table/crud-bstable.html. [Último acceso: 10 febrero 2023].

[25] AEPD, «Guía sobre el uso de las cookies,» 2022. https://www.aepd.es/es/documento/guia-cookies.pdf.

[26] L. Ali, M. Asadi, D. Gasevic, J. Jovanovic y M. Hatala, «Factors influencing beliefs for adoption of a learning analytics tool: An empirical study,» *Computers & Education*, vol. 62, pp. 130-148, 2013.

APÉNDICES

Apéndice A - Tabla de los diferentes tipos de trazas en Articoding

Información recogida	Verbo xAPI usado	Tipo objeto xAPI	Extensiones
Inicio del juego	Initialized	Game	● Resolución ● Pantalla completa ● Idioma
Finalización de un nivel de juego	Progressed	Game	
Finalización del último nivel del juego	Completed	Game	● Mismas extensiones que en un nivel de juego
Acceso a la sección de Jugar (Variables, Bucles...)	Accessed	Screen	● Categoría elegida (si la hay)
Acceso a la sección de Tutoriales (en el menú)	Accessed	Screen	● Desde menú ● Categoría elegida (si la hay)

Acceso a la sección de Modo Creación	Accessed	Screen	
Inicio de un nivel de juego (con estado inicial del código)	Initialized	Level	• Código inicial
Pulsado botón de ejecución del código	Interacted	GameObject	
Ejecución del código (con estado del código)	Progressed	Level	• Estado actual del código
Finalización de un nivel de juego (victoria con info de las 3 estrellas o derrota)	Completed	Level	• Estrellas obtenidas (score -> raw) • Pasos mínimos (minimun_steps) • Pistas usadas (no_hints) • Nivel pasado a la primera (first_execution) • Movimientos del laser (steps)
Inicio del primer nivel de una categoría	Initialized	Completable	

Finalización de un nivel de una categoría (no el último)	Progressed	Completable	
Finalización del último nivel de una categoría	Completed	Completable	
Pulsado botón de reinicio de nivel	Interacted	GameObject	
Reintento de un nivel de juego	Initialized	Level	
Salida al menú principal (desde un nivel de juego)	Completed (fail)	Level	• Número de pasos • Nivel • Estado actual del código
Creación de una variable	Interacted	GameObject	• Nombre de la variable • Tipo de bloque • Action • Nivel
Borrado de una variable	Interacted	GameObject	• Nombre de la variable • Tipo de bloque • Action • Nivel
Generación de una tarjeta	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque

Eliminación de una tarjeta	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque
Encaje de una tarjeta con otra	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque • Info de conexión
Desencaje de una tarjeta con otra	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque • Info de conexión
Movimiento de una tarjeta	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque • Posición
Cambio del valor de una variable (textos, números, etc.)	Interacted	GameObject	• Tipo de bloque • Action • ID del bloque • Valor antiguo • Valor nuevo
Duplicación de una tarjeta	Interacted	GameObject	• Tipo de bloque • Action • ID de la nueva tarjeta
Uso de una pista	Interacted	GameObject	• Número de pistas restantes • Nivel

Acceso/Cierre del panel de tutoriales (desde un nivel del juego)	Accessed	Screen	
Categoría de Tutorial elegida (en un nivel)	Accessed	Accesible	• Categoría • Desde: nivel
Inicio del modo creación	Initialized	Completable	
Pulsado botón de reinicio de nivel de creación	Interacted	GameObject	
Reinicio del modo creación	Initialized	Completable	
Salida al menú principal (desde el modo creación)	Completed (fail)	Completable	
Validación completada	Completed	Completable	• Número de pasos
Generación del tablero (con el tamaño elegido)	Interacted	GameObject	• Número de filas • Número de columnas
Elemento de tablero cogido desde el tablero	Interacted	GameObject	• ID del elemento • Acción • Posición antigua • Posición nueva

Elemento de tablero cogido desde el creador	Interacted	GameObject	• ID del elemento • Acción • Posición
Elemento de tablero colocado	Interacted	GameObject	• ID del elemento • Acción • Posición • Rotación
Elemento de tablero eliminado	Interacted	GameObject	• ID del elemento • Acción • Posición • Rotación
Elemento de tablero girado	Interacted	GameObject	• ID del elemento • Acción • Posición • Rotación
Celda cambiada	Interacted	GameObject	• Posición • Nuevo estado
Estado del objeto cambiado	Interacted	GameObject	• ID del elemento • Nombre del valor cambiado • Valor antiguo • Valor nuevo
Pulsado botón de cambio de modo (en el creador)	Interacted	GameObject	
Cambio de modo (con el estado del tablero si se cambia	Accessed	Screen	• Modo • Estado del tablero

al modo de validación)			
Botón de reseteo de vista (en nivel o creación)	Interacted	GameObject	
Minimización de panel de Victoria	Interacted	GameObject	
Minimización de panel de Derrota	Interacted	GameObject	
Aparición de un pop-up de tutorial	Initialized	Dialog fragment	• Contenido
Cierre de un pop-up de tutorial	Completed	Dialog fragment	• Contenido
Menú de configuración interactuado	Interacted	GameObject	• Abierto o cerrado
Cambio de lenguaje	Interacted	GameObject	• Nuevo lenguaje
Cambio de resolución	Interacted	GameObject	• Nueva resolución
Cambio a modo ventana/pantalla completa	Interacted	GameObject	• Modo
Botón de créditos pulsado	Interacted	GameObject	
Botón de salir pulsado	Interacted	GameObject	

Tabla 1: Listado de los diferentes tipos de trazas en Articoding

Apéndice B - Formulario con posibles respuestas utilizado en las evaluaciones con usuarios

- Primera Sección
    - ¿Cuál es tu rango de edad?
        - Menor de 18 años.
        - 18 a 24 años.
        - 25 a 34 años.
        - 35 a 44 años.
        - 45 a 54 años.
        - Más de 54 años.
    - ¿Cuál es tu nivel de estudios?
        - Estudios primarios.
        - Estudios secundarios.
        - Bachillerato.
        - Formación profesional.
        - Estudios universitarios.
    - Si es posible especificarlo, ¿en qué área de conocimiento se han centrado tus estudios?
        - [Texto de respuesta corta].
    - ¿Has jugado a Articoding?
        - Sí.
        - No.
        - Tal vez.
        - Otra… [Texto de respuesta corta].
    - ¿Lo has usado con estudiantes?
        - Sí.
        - No.
        - Tal vez.
        - Otra… [Texto de respuesta corta].

- Segunda Sección
    - Pestaña de Resultados Generales
        - Conocer el tiempo medio empleado por los alumnos para superar cada categoría es útil.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Conocer los intentos medios empleados por los alumnos para superar cada categoría es útil.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
    - Pestaña de Alumnos
        - La vista general de los alumnos me permite evaluar individualmente a cada uno.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Tengo claro cómo se puede cambiar el nombre de los alumnos.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - El número de alumnos que se ven en el listado por página me parece el adecuado.

            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
    - Pestaña de Categorías
        - Conocer el tiempo, intentos y estrellas medios de los alumnos por cada nivel me ayuda a conocer qué problemas pueden tener.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Los gráficos de vela que representan la distribución de tiempo empleado por los alumnos son útiles.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Los puntos que están más apartados en los gráficos me ayudan a la hora de detectar qué alumnos han tenido más dificultades en niveles concretos.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Tengo claro cómo filtrar por categorías de niveles.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.

        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
    - Pestaña de Comparativa
        - Soy capaz de ver qué diferencias hay entre mis alumnos y los datos globales.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Entiendo los datos que me proporcionan las gráficas.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Tengo claro cómo filtrar por grupos de niveles.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Me parece útil poder comparar mis participantes con los datos globales.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
    - Pestaña de Errores
        - Sé cómo identificar qué niveles son los que tienen más errores.
            - [1] Muy en desacuerdo.

        - [2] Algo en desacuerdo.
        - [3] Ni de acuerdo ni en desacuerdo.
        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
    - Me parece útil conocer qué porcentaje de errores por nivel tienen mis alumnos.
        - [1] Muy en desacuerdo.
        - [2] Algo en desacuerdo.
        - [3] Ni de acuerdo ni en desacuerdo.
        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
    - Tengo claro cómo filtrar por tipos de errores.
        - [1] Muy en desacuerdo.
        - [2] Algo en desacuerdo.
        - [3] Ni de acuerdo ni en desacuerdo.
        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
    - Soy capaz de identificar qué alumnos necesitan más ayuda con este tipo de error.
        - [1] Muy en desacuerdo.
        - [2] Algo en desacuerdo.
        - [3] Ni de acuerdo ni en desacuerdo.
        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
    - Me parece útil poder ver en la tabla de todos los alumnos los errores concretos que ha tenido cada uno.
        - [1] Muy en desacuerdo.
        - [2] Algo en desacuerdo.
        - [3] Ni de acuerdo ni en desacuerdo.
        - [4] Algo de acuerdo.
        - [5] Muy de acuerdo.
- Preguntas Globales

- El tiempo de carga a la hora de hacer el login es razonable.
    - [1] Muy en desacuerdo.
    - [2] Algo en desacuerdo.
    - [3] Ni de acuerdo ni en desacuerdo.
    - [4] Algo de acuerdo.
    - [5] Muy de acuerdo.
- La navegación por la página web es intuitiva.
    - [1] Muy en desacuerdo.
    - [2] Algo en desacuerdo.
    - [3] Ni de acuerdo ni en desacuerdo.
    - [4] Algo de acuerdo.
    - [5] Muy de acuerdo.
- En caso negativo en la anterior pregunta, ¿qué mejorarías de la página?
    - [Texto de respuesta corta].
- La herramienta me ayuda a identificar los problemas con los que los alumnos tienen más problemas.
    - [1] Muy en desacuerdo.
    - [2] Algo en desacuerdo.
    - [3] Ni de acuerdo ni en desacuerdo.
    - [4] Algo de acuerdo.
    - [5] Muy de acuerdo.
- Me gustaría poder usar esta herramienta de análisis como ayuda en mis clases.
    - [1] Muy en desacuerdo.
    - [2] Algo en desacuerdo.
    - [3] Ni de acuerdo ni en desacuerdo.
    - [4] Algo de acuerdo.
    - [5] Muy de acuerdo.
- La herramienta me desborda con demasiada información.
    - [1] Muy en desacuerdo.
    - [2] Algo en desacuerdo.

            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - La herramienta me permite conocer qué errores cometen los alumnos para poder ayudarlos.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - Echo de menos algún tipo de análisis en la página.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.
        - En caso afirmativo en la anterior pregunta, ¿qué análisis echas de menos?
            - [Texto de respuesta corta].
        - Recomiendo esta página web a otros profesores que enseñan programación.
            - [1] Muy en desacuerdo.
            - [2] Algo en desacuerdo.
            - [3] Ni de acuerdo ni en desacuerdo.
            - [4] Algo de acuerdo.
            - [5] Muy de acuerdo.

- Tercera Sección
    - ¿Cuántos jugadores han conseguido superar la categoría de variables?
        - [Texto de respuesta corta].
    - ¿Cuál es el tiempo medio de la clase en la categoría de Types?
        - [Texto de respuesta corta].

- ¿Hasta qué nivel ha llegado el quinto participante de la tabla de la primera página?
  - [Texto de respuesta corta].
- ¿Cuántas estrellas de media han obtenido los alumnos en el nivel Variables 3?
  - [Texto de respuesta corta].
- ¿Cuál es la media de intentos de los alumnos en el nivel de Variables 2?
  - [Texto de respuesta corta].
- ¿Y el valor máximo?
  - [Texto de respuesta corta].
- ¿Quién ha superado más niveles de variables (la clase o el global)?
  - [Texto de respuesta corta].
- ¿Cuál es el nivel con más errores de variables?
  - [Texto de respuesta corta].
- ¿Y el que menos?
  - [Texto de respuesta corta].
- ¿Cuál es el último nivel alcanzado por este alumno?
  - [Texto de respuesta corta].