Aplicativo multiplataforma para el aprendizaje lúdico de conceptos de inteligencia artificial enfocado en el entorno académico

David Alejandro Babativa Díaz
Juan Esteban Hernández Álvarez

Universidad Santo Tomás
Facultad de Ingenieria Electrónica
Bogotá, Colombia
2023

Aplicativo multiplataforma para el aprendizaje lúdico de conceptos de inteligencia artificial enfocado en el entorno académico

David Alejandro Babativa Díaz
Juan Esteban Hernández Álvarez

Proyecto de grado presentado como requisito para optar al título de INGENIERO ELECTRÓNICO

Director: MSc. Sindy Paola Amaya
Co-Director: MSc. Armando Mateus Rojas
Co-Director: PhD. Juan Manuel Calderón Chávez

Grupo de Estudio y Desarrollo en Robótica - GED
Universidad Santo Tomás
Facultad de Ingeniería Electrónica
Bogotá, Colombia
2023

Autoridades de la Universidad

RECTOR GENERAL
R.P. FRAY ÁLVARO JOSÉ ARANGO RESTREPO, O.P.

VICERRECTOR ADMINISTRATIVO Y FINANCIERO GENERAL
R.P. FRAY LUIS FRANCISCO SASTOQUE POVEDA, O.P.

VICERRECTOR ACADÉMICO GENERAL
R.P. FRAY EDUARDO GONZALEZ GIL, O.P.

SECRETARIO GENERAL
Dra. INGRID LORENA CAMPOS VARGAS

DECANO DIVISIÓN DE INGENIERÍAS
R.P. FRAY ERICO JUAN MACCI CÉSPEDES, O.P.

SECRETARIA DE DIVISIÓN
E.C. LUZ PATRICIA ROCHA CAICEDO

DECANO FACULTAD DE INGENIERÍA ELECTRÓNICA
Dr. CARLOS ANDRES TORRES PINZÓN

Nota de aceptación

_____
_____
_____
_____
_____
_____

_____

Firma del autor

_____

Firma del jurado

_____

Firma del jurado

BOGOTÁ D.C. ——— DE 2023

Advertencia

La Universidad Santo Tomás no se hace responsable de las opiniones y conceptos expresados en el trabajo de grado, solo velará por qué no se publique nada contrario al dogma ni a la moral católica y porque el trabajo no tenga ataques personales y únicamente se vea el anhelo de buscar la verdad científica.

Capítulo III –Art. 46 del Reglamento de la Universidad Santo Tomás.

Dedicatoria

Dedico este proyecto a mis padres, Cristina y Arley, por su constante apoyo durante todo mi desarrollo como profesional. También a toda mi familia por siempre darme su amor incondicional y su motivación para llegar hasta este punto.

Juan Esteban Hérnandez Álvarez

Dedico este proyecto a mis padres y a mi hermano porque por su amor, paciencia y apoyo he podido llegar hasta este punto, todo el trabajo que ellos han puesto en mi y el valor que siempre me recordaban que tengo. También a toda mi familia porque siempre están allí para mí.

David Alejandro Babativa Díaz

Agradecimientos

Agradezco principalmente a mis padres por brindarme todo lo necesario en mi desarrollo como ingeniero, a mi directora Sindy Paola Amaya y mis Co-directores Armando Mateus Rojas y Juan Manuel Calderón Chávez por su guía y conocimiento durante todo el desarrollo del proyecto y a mi familia en general por siempre brindarme su apoyo incondicional en mi camino de desarrollarme como profesional.

Juan Esteban Hernández Álvarez

Le agradezco a mis padres Flor Marina y Ricardo por su apoyo incondicional para cumplir todos mis objetivos, a todos mis docentes les quiero agradecer por ser una guía constante en mi camino, a mis compañeros, en particular a Juan Esteban por su apoyo y constancia en este viaje. Finalmente les quiero agradecer a todas aquellas personas que de una manera u otra han influido en mí para mejorar personalmente y como valor para la sociedad.

David Alejandro Babativa Díaz

Índice

Resumen

Contexto

El presente trabajo de grado se centra en el desarrollo de un aplicativo multiplataforma utilizando el *Framework Flutter*, enfocado a ser una herramienta para enseñar conceptos básicos de inteligencia artificial (*IA*) que utiliza técnicas de gamificación. La *IA* es un campo en constante evolución y comprender sus fundamentos es esencial en diversas disciplinas, como la informática, la ingeniería y la toma de decisiones empresariales. Sin embargo, aprender estos conceptos puede resultar desafiante debido a su complejidad y a la falta de recursos educativos interactivos.

Método

Se implementó la metodología *SCRUM* debido a que ofrece una forma estructurada y adaptable de gestionar proyectos, permitiendo una mayor colaboración, adaptabilidad y entrega temprana de valor. Estas características hacen de *SCRUM* una elección popular para muchas organizaciones que buscan mejorar la forma en que entregan productos comúnmente como desarrollo de software y servicios.

Resultados

Los resultados muestran por medio de los casos de prueba el desarrollo de todas las pantallas, procesos y material audiovisual necesario para el funcionamiento total de un aplicativo multiplataforma que usa elementos de gamificación como herramienta para enseñar conceptos básicos de inteligencia artificial enfocado en un ambiente educativo para medir el progreso y tener la posibilidad de tener acompañamiento por parte de un evaluador.

Conclusiones

Existen varias estrategias de gamificación como se investigaron en la fase uno del desarrollo conceptual sin embargo para hacer un aplicativo apropiado para su propósito y para el usuario es necesario escoger adecuadamente un grupo de estas estrategias, porque deben ser compatibles entre sí para que el usuario se sienta inmerso en una experiencia diseñada específicamente para él. Es decir, para este proyecto una forma adecuada que se encontró para combinar estas estrategias fue utilizando "Puntos y niveles", "Desafíos y misiones", "Recompensas virtuales" y "Personalización".

La gamificación como herramienta tiene un método para su utilización, este método sigue diversos pasos para poder definir que un aplicativo efectivamente esta gamificado como muestra la documentación investigada [1] y [2] en el capitulo marco teórico de este documento, pero en la practica se notó que en el proceso de desarrollo pueden intervenir sujetos externos y es necesario aplicar una nueva estrategia de gamificación para solventar este inconveniente sin necesidad de seguir los pasos del método, como sucedió con la planeación en la fase uno del desarrollo conceptual para integrar juegos en el aplicativo al no tener un objetivo especifico referente al tema que trata de enseñar, sin embargo es una estrategia para motivar al usuario mediante una actividad lúdica en el proceso de aprendizaje, mejor conocido como pausa activa.

El desarrollo del aplicativo que inicialmente fue desarrollado sobre la plataforma *Android* fue fácilmente portable a las diferentes plataformas como *iOS GNU/linux* y *Windows*, lo anterior se deduce de la fase de verificación dentro del documento.

La integración de los diferentes módulos fue realizado mediante la creación de arboles de *Widgets* dentro de la carpeta *lib* del proyecto *Flutter* diseñando las relaciones entre *Widgets* y la limitación de servicios de la arquitectura de micro-servicios, lo anterior se deduce de la fase de implementación e integración dentro del aplicativo.

La elección de los contenidos de las secciones de los libros *Artificial Intelligence: A Modern Approach. 4ed* y *Automate the Boring Stuff with Python* de los autores *Stuart Russell, Peter Norvig y Al Sweigart*, fue establecida mediante la relación del plan de estudios de los estudiantes de básica-media en Colombia y los requerimientos de conocimientos para el aprendizaje de dichas secciones, lo anterior se deduce de la fase de investigación dentro del documento.

El uso de la metodología *SCRUM* y la implementación de la arquitectura basada en micro-componentes en el desarrollo del aplicativo permiten la agilidad de desarrollo y despliegue de componentes dentro de la arquitectura de manera individual, lo anterior se deduce de las fases de implementación y verificación dentro del documento.

Palabras clave: Aplicativo, *AWS*, Gamificación, *Flutter*, Datos, Modelos, *Python*, Pantallas.

Abstract

Context

This project focuses on the development of a cross-platform application using the Flutter framework. The application is designed to be a tool that can teach basic concepts of artificial intelligence (AI) using gamification techniques. AI is an ever-evolving field, and understanding its fundamentals is essential across disciplines, including computer science, engineering, and business decision-making. However, learning these concepts can be challenging due to their complexity and the lack of interactive educational resources.

Method

SCRUM methodology is implemented because it offers a structured and adaptable way of managing projects, enabling greater collaboration, adaptability, and early delivery of value. These features make SCRUM a popular choice for many organizations seeking to improve their product delivery processes, such as software development and services [3].

Results

The results show that all the screens, processes, and audiovisual material necessary for the full operation of a multiplatform application that uses gamification as a tool to teach basic concepts of artificial intelligence in an educational environment are delivered through the test cases. These test cases also measure the progress and provide follow-up, conducted by an evaluator.

Finally, it is possible to learn artificial intelligence concepts through a multiplatform application that utilizes audiovisual material. This application allows users to answer questions related to these concepts.

Conclusions

There are several gamification strategies investigated in phase one of conceptual development. However, to make an application suitable for this purpose and the user, it is necessary to properly choose a group of these strategies because they must be compatible with each other. This compatibility ensures that the user feels immersed in an experience designed specifically for them. For this project, an adequate way that was found to combine these strategies was by using "Points and levels," "Challenges and missions," "Virtual rewards," and "Customization".

Gamification, as a tool, follows a method that entails several steps to effectively gamify an application, as evidenced by the researched documentation [1] and [2] in the theoretical framework chapter of this document. However, in practice, it has been observed that external factors can intervene during the development process, necessitating the application of new gamification strategies to address such inconveniences. This was evident in the planning phase of the conceptual development, where integrating games into the application lacked a specific objective pertaining to the teaching topic. Nevertheless, it served as a strategy to motivate users through playful activities during the learning process, commonly referred to as active pause.

The application, initially developed on the Android platform, was easily portable to different platforms, including iOS, GNU/Linux, and Windows. This conclusion is drawn from the verification phase within the document.

The utilization of the SCRUM methodology and the implementation of the architecture based on micro-components in the development of the application enable the agility of development and deployment of individual components within the architecture. This deduction is based on the implementation and verification phases outlined in the document.

Keywords: Application, *AWS*, *Gamification*, *Flutter*, *Data*, *Models*, *Python*, *Screens*.

Índice de Figuras

Índice de tablas

Glosario

API: Es un conjunto de reglas y protocolos que permite a diferentes aplicaciones comunicarse entre sí. Es una interfaz que define cómo los distintos componentes de software deben interactuar y qué operaciones pueden realizar [4].

API RESTful: Es un estilo arquitectónico para el diseño de servicios web que se basa en el protocolo *HTTP*. Se considera una de las formas más comunes y populares de implementar *APIs* [4].

Aplicación Multiplataforma: Se refiere a un software o aplicación que puede ejecutarse en múltiples plataformas o sistemas operativos sin necesidad de ser desarrollado de forma independiente para cada una de ellas [5][6][7].

Aplicación Nativa: Se refiere a un tipo de software diseñado y desarrollado específicamente para ser ejecutado en un sistema operativo en particular, aprovechando al máximo las capacidades y características de ese sistema. Las aplicaciones nativas están escritas en un lenguaje de programación compatible con el sistema operativo objetivo y se compilan directamente en código máquina [8][9].

Arbol de *Widgets*: Es una estructura jerárquica que representa la composición y la organización de los componentes visuales en una interfaz de usuario. Es una forma común de representar la estructura de elementos de una interfaz gráfica en muchos *frameworks* y bibliotecas de desarrollo de aplicaciones [5][7].

Framework: Es una estructura o grupo de herramientas y bibliotecas de software que proporciona una base para el desarrollo de aplicaciones. Es una abstracción que ofrece una serie de funcionalidades predefinidas y reglas de diseño que permiten a los desarrolladores crear aplicaciones de manera más eficiente y consistente [5].

HTTP: Es un protocolo de comunicación utilizado para el intercambio de información en servicios web [4].

NoSQL: Es un tipo de base de datos diseñada para el almacenamiento y la gestión de datos que difiere del enfoque tradicional de las bases de datos relacionales (SQL) [10].

Serverless: Es un modelo de computación en la nube que permite a los desarrolladores crear y ejecutar aplicaciones sin tener que preocuparse por la infraestructura subyacente [11].

Sprint: Es el tiempo establecido dentro de la metodología *SCRUM* para desarrollar y entregar un incremento de producto [3].

SQL: Es un lenguaje de programación utilizado para administrar bases de datos relacionales [12].

Stateful Widget: Es un tipo de *Widget* de *Flutter* que tiene un estado mutable interno, lo que significa que puede cambiar su apariencia o comportamiento a lo largo del tiempo en respuesta a eventos o interacciones del usuario [5][7].

Stateless Widget: Es un tipo de *Widget* que representa un elemento dentro una pantalla dentro de un aplicativo *Flutter*, sin tener un estado interno mutable. Se construye a partir de un conjunto de propiedades proporcionadas en el momento de su creación y no puede cambiar su apariencia o comportamiento una vez que se ha renderizado en la pantalla [5][7].

Widget: es un componente visual o interactivo que forma parte de una interfaz de usuario en una aplicación o sistema. Los *widgets* se utilizan para mostrar información, permitir la interacción del usuario y proporcionar funcionalidades específicas de manera visualmente atractiva y accesible [5][7].

Capítulo 1

Introducción

1.1 Planteamiento del Problema

La tecnología hace parte de los pilares de la sociedad brindando avances en el impacto social y cultural del país. Los países del primer mundo enfocan su capital de inversión social a término del futuro enfocándose en la educación y la inversión en capital intelectual; la robótica y la automatización en el sector industrial hacen que las necesidades en empleo sean diferentes, por ende la educación y la propiedad intelectual son conceptos que son considerados importantes para el futuro, dejando de lado en menos proporción la producción de materias primas [13]. Latinoamerica, en términos de progreso tecnológico, está atrasada en comparación con Europa, América del Norte y el sudeste asiático, haciendo necesario enfocarse en el cumplimiento de objetivos definididos por el Ministerio de Educación Nacional. De lo anterior es pertinente preguntarse: ¿Qué tipo de educación es necesaria para las exigencias en el futuro? Responder esta pregunta conlleva esfuerzos continuos por parte de organizaciones gubernamentales y no gubernamentales y por ende una solución debe establecerse en el contexto de las problemas sociales de inequidad y problemas ambientales y son las próximas generaciones que se enfrentarán a estos retos [14].

Los estudiantes actualmente tienen que desarrollar habilidades tecnológicas y desenvolverse de forma competitiva en el mercado laboral, y ser capaces de solucionar problemas sociales mediante su conocimiento científico; por ende los Ministerios de Educación Nacional y de Tecnologías de la Información desarrollaron en 2019 el proyecto *NOVACAMP* donde apuntan a la solución de problemas vinculados a la vida real. Con el objetivo de que los estudiantes reten sus conocimientos y desarrollen diversas habilidades *STEM* en la escuela dado a que la gran mayoría de las instituciones educativas nacionales se enfocan en la enseñanza de las ciencias(S) y matemáticas(M); sin embargo se deja de lado el concepto de tecnología(T) que es de suma importancia en el futuro del país y menos a la ingeniería(E) que refleja en la educación los conceptos de diseño e innovación. Otro problema a solucionar en el futuro de la educación en Colombia es la incursión en un enfoque educativo que promueva el cierre de brechas de género y llegar a los diferentes grupos étnicos del país [15].

Según el Ministerio de Educación de Colombia mediante la implementación del proyecto *NOVACAMP* en el 2022 lograron la asistencia de 368 grupos de más de 1.000 estudiantes y 300 docentes y tuvieron una participación activa de diferentes grupos étnicos y géneros representados en la siguiente estadística [16].

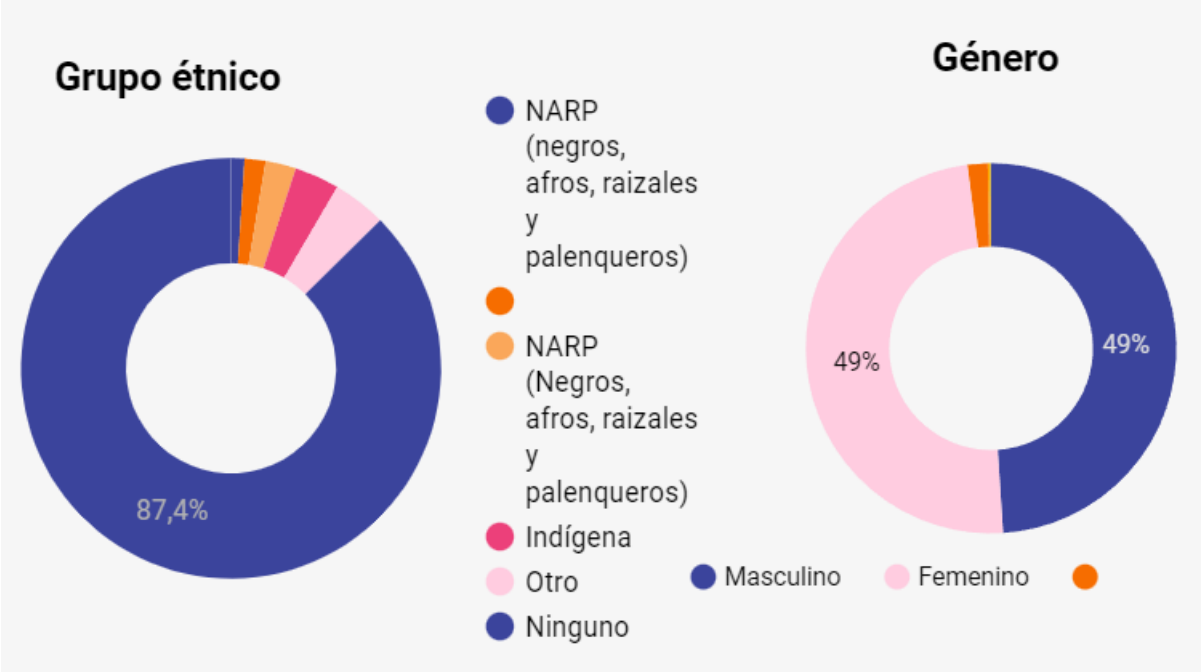

Figure 1.1: Estadísticas de participación de grupos étnicos y identidades de géneros en el proyecto *NOVACAMP* 2022. Fuente: Colombia Aprende Ministerio de Educación de Colombia [15].

Según la Universidad de Oxford en 2020, Colombia se posicionó entre los principales países con un alto potencial de crecimiento en el uso de inteligencia artificial en instituciones gubernamentales. Esto se logra a través de la transformación digital tanto del sector público como privado, con el objetivo de generar valor social y económico y prepararse para los desafíos de la Cuarta Revolución Industrial [17]. En cuanto a la inteligencia artificial en Latinoamérica, países como Uruguay, Chile, Brasil, México y Perú han avanzado en el desarrollo de estrategias acordes a la inteligencia artificial, gestionando de manera independiente en cada país y aplicando estas estrategias principalmente en el sector público [18].

Colombia se prepara para el uso y desarrollo de la inteligencia artificial mediante el cumplimiento de la Ley 3975 del 2019 del Consejo Nacional de Política Económica y Social, titulada "POLÍTICA NACIONAL PARA LA TRANSFORMACIÓN DIGITAL E INTELIGENCIA ARTIFICIAL". Esta ley proporciona la información necesaria sobre la política de búsqueda y generación de valor social y económico en Colombia, a través del fortalecimiento del capital humano y el desarrollo de condiciones de investigación, además, se cumple con los catorce principios de desarrollo de inteligencia artificial en el país [19].

Además del impacto de la inteligencia artificial en el gobierno colombiano, sus aplicaciones abarcan desde usos recreacionales hasta implementaciones científicas. Estas incluyen la personalización de publicidad, la individualización de perfiles en redes sociales, la identificación automática de correos electrónicos no deseados, algoritmos de reconstrucción de información perdida o dañada, la prevención de ataques cibernéticos, el diseño de controladores industriales, y muchas otras aplicaciones que surgen constantemente [19] [20].

Teniendo en cuenta los proyectos actuales liderados por el gobierno colombiano para posibilitar la enseñanza de *STEM* en la educación media, se busca desarrollar una herramienta de software que permita enseñar un área específica de *STEM*, que es la inteligencia artificial. En este contexto, se plantea la siguiente pregunta de investigación:

¿Qué elementos debe tener un aplicativo multiplataforma como herramienta de gamificación para los procesos de enseñanza y aprendizaje de conceptos básicos de inteligencia artificial en un curso de básica media en Colombia?

1.2 Objetivos

Considerando la pregunta problema y las temáticas a abordar en el desarrollo del presente proyecto, se plantea el objetivo general y los objetivos específicos a cumplir.

1.2.1 Objetivo General

Desarrollar un aplicativo para la enseñanza aprendizaje de conceptos básicos de inteligencia artificial mediante estrategias de gamificación bajo un entorno de programación multiplataforma.

1.2.2 Objetivos Específicos

- Investigar las diferentes estrategias didácticas de la gamificación para diseñar un videojuego que facilite la enseñanza de conceptos de inteligencia artificial mediante la consulta de investigación de proyectos similares en bases de datos científicas obtenidas en el sistema *CRAI* de la Universidad Santo Tomás.
- Diseñar la interfaz de usuario de un videojuego y su modelo de datos para el desarrollo de software que cumpla las estrategias de gamificación seleccionadas, así como las políticas de acceso de usuario usando técnicas de diseño de interfaces de usuarios para gamificación.
- Implementar un módulo de inteligencia artificial basado en procesamiento de lenguaje natural que sirva como un asistente virtual del usuario dentro del aplicativo.
- Integrar el módulo y la interfaz de usuario en un aplicativo multiplataforma (GNU/Linux, Windows, Android y iOS) incluyendo estrategias de gamificación para la enseñanza-aprendizaje de conceptos de inteligencia artificial.
- Verificar el funcionamiento del aplicativo multiplataforma a través del planteamiento de casos de prueba para diferentes roles de usuario.

1.3 Justificación

Acorde al planteamiento del problema, existe la necesidad en Colombia de reforzar el aprendizaje en *STEM* para ayudar a mitigar las brechas de educación presentes en el país [21]. Por lo tanto, la implementación de un aplicativo software como posible herramienta de apoyo mediante el uso de métodos de gamificación modernos es una opción viable para atacar esta problemática. Además, a través de la implementación de técnicas de inteligencia artificial como instrumento de soporte para facilitar el proceso de enseñanza de tópicos académicos y el uso de videojuegos como herramienta de aprendizaje, se puede incrementar la motivación y la dedicación de los estudiantes con respecto al uso de técnicas tradicionales de enseñanza, convirtiéndose en facilitadores de los procesos de enseñanza-aprendizaje [22].

Uno de los programas de éxito de la implementación de gamificación y temas de *STEM*, es el programa de educación denominada *EDGAME* desarrollado por la embajada estadounidense en la República de Filipinas que tiene como fin unir esfuerzos de profesores, desarrolladores y entusiastas en *STEM*, para motivar y enseñar a estudiantes filipinos diferentes conceptos tecnológicos, todo esto con el objetivo de ayudar a mitigar las brechas de educación en el país [23].

La inteligencia artificial en el contexto educativo nacional es un concepto que se ha trabajado en los últimos años y se ve como un tema importante para el futuro cercano, debido al aumento de las necesidades laborales en este sector por lo tanto hacen que la educación se enfoque a preparar profesionales expertos en esta área [24]. Por otra parte, la gamificación es una de las estrategias de enseñanza con mayor proyección de uso en el futuro ya que permite personificar la enseñanza y adaptarse a las necesidades de cada estudiante [25].

Uno de los mayores casos de éxito de la implementación de técnicas de gamificación como método de enseñanza es el caso de *Duolingo*, un aplicativo multiplataforma enfocado en la enseñanza aprendizaje de diferentes lenguajes mediante el uso de juegos y actividades rápidas que permiten tener la atención de los usuarios y enseñar un concepto concreto acerca de un idioma, siendo así una compañía que tiene un valor en el mercado de 3.68 Billones de dólares en 2022 [26] [27].

En relación al impacto local que tendría el proyecto en la Universidad Santo Tomás este ayudaría en el desarrollo de competencias de inteligencia artificial en cursos especializados en este tema, convirtiéndose en una herramienta que podría ayudar a los estudiantes a entender conceptos relacionados con esta área del conocimiento incorporándose como una herramienta transversal de la educación, posibilitando la enseñanza también a jóvenes de educación básica media con el objetivo de motivar el estudio de carreras enfocadas en *STEM* en Colombia.

1.4 Impacto Social

Por medio del desarrollo de este proyecto se busca obtener un impacto social desde el Grupo de Estudio y Desarrollo en Robótica - GED de la facultad de Ingeniería Electrónica de la Universidad Santo Tomás a través de la línea de investigación en inteligencia computacional, para ayudar al desarrollo de competencias tecnológicas de la comunidad. En este caso centrándose en estudiantes de educación básica media, para que mediante los conocimientos adquiridos en sus espacios académicos enfocadas en *STEM* y la implementación de un módulo de asistencia virtual que usa la inteligencia artificial puedan relacionar y fortalecer conceptos sobre la lógica, la aritmética, estadística, la programación y la inteligencia artificial, mediante la definición de recursos de aprendizaje (retos, desafíos, etc) propuestos dentro del videojuego, de esta manera buscar que aprendan conceptos de inteligencia artificial [28].

Este proyecto también está enfocado en motivar a estudiantes de educación básica media, principalmente grados décimo y once, para estudiar programas académicos afines a la ingeniería, la matemática, la programación, la ciencia y la tecnología; y promover el ingreso de mujeres jóvenes a áreas disciplinares *STEM*, (dado que históricamente el porcentaje de ingreso ha sido menor) [29].

Con respecto al programa de Ingeniería Electrónica de la Universidad Santo Tomás, se plantea enfocar el proyecto en dos lineas estratégicas de proyección social a las que apunta la Universidad, las cuales son Desarrollo Comunitario y Educación Continua. Desde este punto de vista el proyecto se presenta como herramienta que permite a cualquier persona que tenga conocimientos básicos de *STEM*, entender y aprender nuevos conceptos de interés enfocados en inteligencia artificial; esta aplicación sería desplegada en repositorios como *GitHub* de manera abierta a cualquier persona de la comunidad que cuente con un dispositivo móvil generando de esta manera una educación progresiva asociada al desarrollo tecnológico [30].

Por otra parte la inteligencia artificial aplicada a los objetivos de desarrollo sostenible es empleada ampliamente en la educación de calidad mediante la utilización de estrategias de gamificación proporcionando sistemas de aprendizaje personalizado para propósitos específicos, un ejemplo es la utilización de aplicaciones que evalúan la capacidad de entendimiento de un nuevo idioma de un usuario para presentarle una ruta de aprendizaje específicamente diseñada para este [31].

Capítulo 2

Estado del Arte

Mediante el continuo desarrollo del sistema educativo, la integración del modelo interdisciplinario *STEM* se ha adaptado gradualmente y ha desempeñado un papel fundamental en la promoción de la educación [32]. Al mismo tiempo, con el respaldo de la Inteligencia Artificial, el campo educativo ha adoptado un enfoque centrado en la resolución de problemas reales utilizando diversas áreas interdisciplinarias de *STEM*. Esto prepara a los nuevos estudiantes para explorar las diferentes áreas de la tecnología y, al mismo tiempo, disfrutar aprendiendo a través de métodos de enseñanza modernos basados en la gamificación, que se explicarán posteriormente.

Para hablar de la integración del modelo interdisciplinario *STEM* en la educación, es necesario remontarse a una de sus áreas fundamentales: la inteligencia artificial.

Ya habiendo definido la inteligencia artificial como una ciencia, se busca incorporarla en el contexto de la enseñanza de *STEM* utilizando métodos de gamificación. Por lo tanto, es fundamental profundizar en la inteligencia artificial y su relación con la gamificación. En el constante desarrollo de los sistemas de administración del aprendizaje, los conceptos de gamificación e inteligencia artificial han adquirido mayor relevancia debido a su flexibilidad, alcance, métodos y otros atributos en el ámbito educativo. Por lo tanto, a través del desarrollo tecnológico, se busca maximizar la competencia de aprendizaje y brindar una experiencia altamente personalizada en el proceso de aprendizaje [33].

2.1 Sistemas de Administración de Aprendizaje y su relación con la inteligencia artificial

Desde la integración del internet en la educación, los sistemas de administración de aprendizaje o (*LMSs*) por sus siglas en ingles han dominado el campo de la educación principalmente atacando los problemas de la enseñanza tradicional y mas específicamente en las restricciones que esta tiene. El objetivo principal de los *LMSs* es administrar el contenido de la educación teniendo en cuenta la colaboración y el control de la misma, por esto conceptos como inteligencia artificial y gamificación, han ido rápidamente evolucionando en este tipo de sistemas, brindando una mejora en el proceso receptivo y así sobrepasando el problema de variación de tipos de estudiantes, para que de esta manera se pueda personalizar la educación independientemente de las diferentes capacidades de aprendizaje, antecedentes educacionales o niveles de educación (Primario, secundario o terciario) [34].

2.2 Predicción automática de estilos de aprendizaje

La predicción automática es una clara aplicación de la inteligencia artificial en el campo de la educación más específicamente en el escenario de los sistemas de administración de aprendizaje actuales, de acuerdo a [35] existen más de 70 estilos de aprendizaje donde sobresaltan los estilos propuestos en el modelo de *Felder* y *Silvermann* y el modelo *VARK* (Visual, Auditivo, Lectura/Escritura y Kinestésico) mediante el uso de la inteligencia artificial se puede identificar y monitorizar los estilos de aprendizajes óptimos de cada uno de los diferentes tipos de estudiantes con el objetivo de ayudar a mejorar los sistemas de administración de aprendizajes [34].

2.3 Uso de la Inteligencia Artificial en el Procesamiento del Lenguaje Natural y su relación con la gamificación

Una de las aplicaciones de la inteligencia artificial es su uso en el procesamiento del lenguaje natural donde busca que los sistemas de cómputo sean capaces de entender texto y palabras habladas como lo son capaces los humanos [36]. Principalmente este tipo de inteligencia artificial es usada en la gamificación como una herramienta de aprendizaje de lenguajes, ya que es un área con un alto crecimiento en el sector del aprendizaje.

2.4 Gamificación

Una herramienta actualmente empleada en los servicios de preparación y capacitación es la gamificación, que se puede describir como la integración de elementos propios de los juegos en un entorno que no los tiene, con el fin de brindar una experiencia lúdica [37]. El propósito de esta integración es aumentar el compromiso y la disposición del cliente, mejorando la calidad del entorno de aprendizaje para generar una sensación placentera [38]. La gamificación también se refiere a un proceso de mejora de un servicio al incorporar elementos de juego para fortalecer la creación de valor para el usuario [39]. Además, se utiliza como una herramienta para facilitar la comprensión de manera más sencilla de los conceptos básicos de STEM.

Actualmente los videojuegos se están convirtiendo en la forma mas popular de entretenimiento a nivel mundial para cualquier género y edad. Esto implica que muchos de los videojuegos requieran que los jugadores desarrollen habilidades de nivel superior las cuales promueven procesos cognitivos. La motivación y el compromiso son aptitudes que pueden ser integradas mediante la incorporación de mecánicas o elementos de juegos en la educación y al mismo tiempo se potencia el aprendizaje de competencias dentro de la asignatura requerida.

Para este proyecto de grado, se plantea utilizar la gamificación como una herramienta de aprendizaje para el concepto de inteligencia, tomando como referencia a España y Estados Unidos debido a que son líderes mundiales en esta área de desarrollo de la gamificación como estrategia de aprendizaje. Según la investigación realizada en el año 2020, el presidente de *ANAGAM* (Asociación Nacional de Gamificación), Juan Carlos Lozano, reconoce que hay una gran cantidad de empresas de todo el mundo que utilizan *ANAGAM* para resolver sus consultas sobre la implementación de la gamificación. Juan Carlos Lozano explica que algunas de las compañías mas recurrentes que hacen uso del capítulo de gamificación se encuentran *Gestazio*, *Cookie Box*, *Medialabs*, *Madison* o *Axis Corporate*. Al aplicar la gamificación son varias compañías de distintas índoles que apuestan por este sistema. Como se pueden encontrar entre otras *Aviva*, *BBVA* o *Accenture* España, que lo han incorporado con mucho éxito [28].

En la actualidad la *UNESCO* es participante fundamental que promueve que todos los docentes usen técnicas específicas con la finalidad de motivar al estudiante. En el caso de los juegos de aprendizaje: el 62% de los docentes en USA aseguran que la gamificación es efectiva, debido a que las instituciones educativas tienen certeza que en las simulaciones y los juegos de computador son herramientas que impulsan los nuevos enfoques en la educación científica. A su vez, motiva a los estudiantes con menos facilidades de aprendizaje, es por esta de otras múltiples razones que el 95% de los docentes usan juegos digitales específicamente diseñados para uso educativo, añadiendo, el 70% de los profesores concuerdan en que el uso de juegos digitales aumentan la motivación y compromiso de los estudiantes, como también, un 64% de maestros opinan que es mas efectivo el aprendizaje en los diferentes niveles educativos [28].

Como un ejemplo de la importancia y la capacidad de mejora en el proceso de aprendizaje se toma de referente la experiencia resultante de introducir un *Escape Room* a nivel educacional en la asignatura de introducción a la estadística que es cursado por diferentes grados de Ciencias Sociales en la Universidad de Castilla-La Mancha, España. El objetivo del estudio era evaluar la posible existencia de diferencias entre estudiantes que fueron parte en actividades de gamificación en entornos universitarios y otros que no, para realizar esta evaluación se aplica con un test para muestras independientes, las cuales son caracterizadas por el género de los participantes y el grado universitario. Esta muestra fue tomada a 77 participantes, y los datos fueron recogidos desde Mayo hasta Agosto del curso 2020/2021 a través de un cuestionario diseñado por los docentes de la respectiva asignatura. Este cuestionario contiene 30 preguntas en escala de *Likert* (1-10) agrupadas en seis bloques [40].

Este estudio deja como resultado valiosa información respecto a la implementación de métodos de gamificación, donde los resultados muestran un grado de satisfacción moderado por los estudiantes encuestados, es decir, mas

que satisfactorios considerando que es la primera implementación de un método de este tipo. Se demuestra principalmente que la influencia del género de los muestreados no es significativa, pero por otro lado, la naturaleza de los grados universitarios de los estudiantes, demuestra que influye significativamente en bloques como lo son: Satisfacción general, la Aplicabilidad, la Organización, pero también, los ítems peor valorados son entre los estudiantes de Trabajo social, haciendo énfasis a la dificultad del juego y al desinterés de la asignatura [40].

2.5 Gamificación en STEM

En el ámbito de *STEM (Science, Technology, Engineering, Maths)*, cuyo propósito es formar integralmente a cualquier ciudadano para que adquiera habilidades adecuadas para enfrentar el avance tecnológico del siglo XXI, la gamificación se ha vuelto cada vez más común en todos los niveles de educación y es una herramienta importante para mejorar el proceso de enseñanza-aprendizaje. A través del análisis de la literatura especializada en el tema, se evidencia que la gamificación contribuye al estudio de las asignaturas relacionadas con STEM en la educación básica y media. Además, se confirma el potencial de esta herramienta en el fomento del aprendizaje significativo, el aumento de la motivación y el desarrollo de competencias transversales [39].

Normalmente los centros educativos de educación básica media imparten de manera compartimentada materias del mismo ámbito pero se organizan de manera aislada de acuerdo con los currículos establecidos, esto quiere decir que fracciona los conocimientos, ocurriendo especialmente en las materias relacionadas con el ámbito científico y tecnológico, las cuales en etapas de secundaria corresponde a las materias de matemáticas, física, química, biología, geología y tecnología, las cuales tienen sus contenidos muy relacionados de manera inherente pero se organizadas de forma independiente. Esto trae como consecuencia que los estudiantes entiendan que no están conectados los conocimientos entre las materias y aun peor que no tiene relación directa con su entorno inmediato, esto provoca que el estudiante desarrolle actitudes negativas como el aburrimiento, el desinterés o incluso el rechazo completo a la asignatura [38].

Según [41], la enseñanza de STEM en educación básica media se ve altamente beneficiada con la implementación de herramientas metodológicas como la gamificación. Teniendo cómo propósito demostrar esta hipótesis mediante la revisión de la literatura en la que se documentan experiencias de docentes que les ha resultado favorable integrar este método en el ámbito de STEM buscando mejorar siempre el proceso de enseñanza-aprendizaje.

Introducir la gamificación en *STEM* tiene el propósito principal de promover la motivación de adquirir conocimientos en un ámbito especifico en el estudiante, por lo tanto la conclusión del estudio realizado en el articulo presenta que mejora las metodologías de enseñanza por medio de experiencias reales posiblemente vivenciales o simuladas que desarrollen el pensamiento critico. Por ende se puede asegurar que efectivamente la gamificación en el alumnado estimula el aprendizaje de *STEM*. Es decir, “aplicar una metodología gamificada con componente tecnológico en *STEM* proporciona la experiencia vivencial que permitirá a los estudiantes comprender y asimilar los contenidos de las materias del ámbito por medio de un aprendizaje significativo y de la participación activa” [41].

2.6 Inteligencia artificial en juegos móviles

Los juegos en plataformas móviles son aplicaciones que atraen a millones de usuarios a nivel mundial, representan un valor importante en el mercado de servicios móviles, en algunos estudios de mercado reportan que los ingresos en la industria de videojuegos inalámbricos esta en el orden de los billones de dolares a nivel global y con un crecimiento del 40% cada año [42] [43], algunos de los aspectos cruciales a su éxito comercial es la aplicación de algoritmos de inteligencia artificial para desafiar competitivamente a los jugadores. Sin embargo, crear este tipo de componente en plataformas móviles es complejo ya que los algoritmos de búsqueda de inteligencia artificial clásicos se ven limitados en dispositivos móviles por su falta de potencia de procesamiento de información, una de las soluciones más sencilla a este problema es la adaptación de algoritmos de *machine learning* a dispositivos móviles para así complementar los algoritmos de inteligencia artificial y que la experiencia de usuario no se vea afectada por la alta demanda de recursos computacionales [44].

Desarrollar un juego móvil completamente desde cero requiere de altas habilidades de programación y entendimiento de gráficos y física. Recientemente, existen bastantes variaciones de juegos de inteligencia artificial desde juegos de mesa virtuales hasta juegos nuevos como lo son los juegos de carreras, disparos en primera persona, juegos de

estrategia en tiempo real, y demás diferentes tipos de juegos. La competición comercial del mercado hace que cada compañía desarrolle sus propios módulos de inteligencia artificial para que funcionen en juegos públicos/comerciales. Dado a que estos módulos son de código abierto, son muy utilizados en adaptaciones de los mismos a diferentes juegos, un ejemplo claro de esto es la inteligencia artificial del juego *Angry Birds* donde se busca desafiar al jugador con modificaciones a los niveles oficiales para hacer mas difícil los niveles el juego en si [45].

Además de mejorar el videojuego, la inteligencia artificial facilita la mejora del contenido del juego en términos de variedad, calidad y cantidad. Esto se logra al permitir la optimización de los recursos computacionales para crear adaptaciones superiores del juego. Además, dada la alta demanda de videojuegos en dispositivos móviles, la experiencia del jugador se convierte en un factor fundamental para el éxito del juego. Mediante el uso del análisis de datos, es posible observar el comportamiento del juego y obtener retroalimentación sobre la experiencia de los jugadores. Esto es de suma importancia para los videojuegos modernos, ya que proporciona la posibilidad de optimizar el juego en todos sus aspectos [46].

2.7 Estadísticas

Según la Asociación Colombiana de Ingenieros de Sistemas (*ACIS*) en el articulo titulado En Colombia sólo el 30% de las mujeres se desempeñan en el sector tecnológico. Indica que *Crack the Code*, una compañía que se especializa en educación tecnológica para niños y jóvenes ha identificado muy poca participación del género femenino en las disciplinas de ciencias, tecnología ingeniería y matemáticas en Latinoaméria, de hecho, en el mundo tecnológico las mujeres en el mejor de los casos ocupan aproximadamente el 20% de los cargos relacionados con la industria [29].

De acuerdo con la UNESCO, "*Descifrar las claves: la educación de las mujeres y niñas en disciplinas de ciencias, tecnología, ingeniería y matemáticas*", solo el 3% de las mujeres estudiantes de educación superior eligen estudios en el campo de las tecnologías de la comunicación y la información (TIC) [29]. Estas cifras siguen evidenciando que en las áreas disciplinarias de ciencias, tecnología, ingeniería y matemáticas, solo el 35% de las mujeres se matriculan en alguna de estas carreras y solo el 30% trabaja en ellas. En Colombia, según datos del Ministerio de Educación, en 2018, el 56% de los graduados de educación superior eran mujeres y el 43% eran hombres. Sin embargo, la inscripción en carreras relacionadas con STEM es del 34,9% para mujeres y del 65% para hombres.
En relación al artículo "*Por áreas, las cifras de formación en disciplinas de tecnología e innovación varían según el género. Según datos del informe de emprendedoras STEM en América Latina, en el área de tecnología, los hombres representan el 74% de la participación, mientras que las mujeres solo un 25%. En el área de ingeniería, el género masculino ocupa el 66% y el género femenino solo un 33%. En cuanto a las mujeres que efectivamente se gradúan en profesiones del área STEM, los datos de Senecyt revelan que en ingeniería industrial y construcción solo se gradúa un 22%, en tecnologías de la información y la comunicación un 27%, y en ciencias naturales, matemáticas y estadística solo un 29%.*" [29].

Según el Ministerio de Tecnologías de la Información y las Comunicaciones en el articulo actualizado el 26 de Noviembre de 2020: las habilidades digitales inteligencia artificial, presenta que la inversión total del proyecto 18.985 millones de pesos con una meta de 4.413 de beneficiarios, con una cantidad de aspirantes de 251.745 en la convocatoria, donde el porcentaje de mujeres fueron de 26,73% es decir, 67.294 y el porcentaje de aspirantes masculinos fue del 68,99% es decir, 173.679 . Fueron admitidos el 3,7% que equivale a 9.309 de los postulados de 34 departamentos y de 889 municipios. Los departamentos con más beneficiarios fueron: Bogotá (3.401), Antioquia (2.163), Valle del Cauca (652), Cundinamarca (422) [47].

2.8 Programación Multiplataforma

La variación de sistemas operativos en dispositivos móviles hace que los estándares, lenguajes de programación, y mercados sean separados. Esto hace más difícil la tarea a los ingenieros de desarrollo de software ya que se necesita saber que plataforma desarrollar primero y adaptar los módulos y librerías del proyecto a varios lenguajes de programación, con este problema en mente, el desarrollo multiplataforma da la opción de construir un aplicativo en un mismo lenguaje de programación para diferentes plataformas (*Apple SDK*, *Android SDK*, Sistemas embebidos, Aplicativos basados en servicios web y aplicaciones de escritorio *Windows* y *GNU/Linux*). La primera opción disponible es las herramientas de desarrollo multiplataforma basados en Web, principalmente facilitando el desarrollo de software mediante la simulación del aplicativo como un servicio web así mediante esta virtualización desplegarse en varios dispositivos, este tipo de desarrollo multiplataforma tiene la desventaja de ser lento con respecto a la interfaz del usuario así brindando una baja experiencia de usuario [48]. Otra solución multiplataforma son los que tienen motores de renderización como el caso de *Skia*, el motor 2D de *Google* el cual impulsa la herramienta multiplataforma *Flutter* brindando así una solución al problema que presentan las herramientas multiplataforma basadas en web ya que al usar un motor de renderización los problemas de respuesta y velocidades del aplicativo se ven resueltos [49]. Existen varias herramientas de programación multiplataforma como los son:

- *React Native* es uno de los *frameworks* mas populares en el momento. Desarrollado internamente en *Facebook*, al principio la compañia la usaba para cambiar su estrategia de desarrollo móvil en una más eficiente. Los componentes de *React* están en el código nativo existente e interactúan con las *API* nativas a través de la interfaz de usuario y el paradigma del lenguaje de programación de JavaScript. [9]

- *Flutter* es el kit de herramientas de desarrollo de software de interfaz de usuario de *Google* basado en el lenguaje de programación *Dart*. La plataforma incluye dos conjuntos de *widgets* propios y totalmente personalizables que consideran todas las diferencias entre las plataformas. Tiene todo empaquetado en su interior y utiliza el motor de renderizado 2D altamente eficiente, *Skia*. [6]

- *Xamarin* es un *SDK*, de código abierto y multiplataforma para desarrollar aplicaciones *Android* e *iOS* con *.NET* y *C#*. *Xamarin* se extiende mediante las capacidades de las herramientas de *.NET* y bibliotecas diseñadas específicamente para crear aplicaciones para *Android*, *iOS*, *Windows*, *tvOS*, *watchOS* y *macOS* [50] [8].

Capítulo 3

Marco teórico

Para el desarrollo del proyecto es necesario definir conceptos utilizados en la ejecución de este, con la finalidad de dar un entendimiento básico de la solución a realizar.

3.1 Gamificación

La gamificación como ya se definió en el estado del arte de este documento. Es una herramienta de educación utilizada en la actualidad que consiste en el uso de mecánicas elementos y técnicas propias de los juegos en ambientes lúdicos en ambientes no lúdicos [38] [39].

3.1.1 Características y elementos de la gamificación:

No es necesario que todos los juegos tengan todos los elementos a continuación, sin embargo, es recomendable incluir la mayo cantidad de elementos [1].

- **Puntos:** Los puntos son una herramienta con la cual se puede llevar la cuenta del progreso de cada estudiante, y por consecuencia conocer en que ámbitos tienen fortalezas y en que otros tienen debilidades [1].
- **Niveles:** Esto aumenta la motivación del usuario en al presentarle diferentes retos con diferentes grados de dificultad [1].
- **Clasificación:** La evaluación típicamente se realiza mediante tablas y generalmente se determina por la cantidad de puntos o insignias obtenidas.[1].
- **Retos:** Son misiones para mostrarle a los estudiantes objetivos para que al alcanzar el cumplimiento de estos evocar el sentimiento de superación [1].
- **Insignias:** Son una herramienta encargada de mostrar los alcances obtenidos de los objetivos o retos con mayor dificultad para definir la culminación de una etapa [2].
- **Premios:** Los premios en conjunto con los retos tienen la finalidad de que al terminar una etapa conseguir una recompensa y de esta manera mejorar la experiencia y motivar al estudiante para continuar [2].

3.1.2 ¿Cómo hacer gamificación?

- **Definir un objetivo:** Los ámbitos en los que se desean desempeñar los estudiantes o las habilidades o conocimientos que se desean enseñar[1] [2] [51].
- **Transformar la educación convencional:** Mediante enfoques entretenidos con el objetivo de potenciar el proceso de enseñanza-aprendizaje. [1] [2] [51].
- **Proponer un reto:** Dejar en claro a los estudiantes cual es el objetivo lúdico. Definir: ¿Por qué es divertido? Y ¿Qué se debe conseguir?[1] [2] [51].

- **Establecer las normas del juego:** Crear reglas o normas sencillas pero concretas de fácil entendimiento para evitar confusiones y poder llegar al objetivo específico deseado. Se recomienda comprobar que quedaron claras las normas [1] [2] [51].

- **Crear recompensas:** Los premios en conjunto con los retos tienen la finalidad de que al terminar una etapa conseguir una recompensa y de esta manera mejorar la experiencia y motivar al estudiante para continuar [1] [2] [51].

3.1.3 Elementos de los juegos:

Los juegos se basan en tres elementos propios. Herranz (2013) señala que existen los siguientes elementos fundamentales: Las dinámicas, las mecánicas y los componentes del juego[1].

- **Dinámicas:** Son los aspectos globales a los que un sistema un sistema gamificado debe orientarse. Se relaciona con los efectos, la motivación y deseos que se buscan generar en el estudiante[1].
  - Restricciones del juego, la posibilidad de resolver una situación problemática en un entorno limitado.
  - Emociones derivadas de la curiosidad y la competitividad que surgen al enfrentar un reto.
  - Narrativa o guión del juego, el cual permite dar una idea general del reto al usuario.
  - Progresión del juego, debido a la importancia de una evolución y que el jugador tenga la sensación de avanzar en el juego.
  - Estatus, es necesario que las personas sean reconocidas.
  - Relaciones entre participantes.

- **Mecánicas:** Son una serie de normas que generan los juegas que se puedan disfrutar, las cuales generar actitudes de compromiso por parte de los usuarios, al aportarles retos y un camino por el que transitar [1].
  - Retos.
  - Oportunidades, competición y colaboración.
  - Dentro de las oportunidades el juego tendría turnos.
  - Formación de equipos.
  - Puntos a los participantes.
  - Clasificación.
  - Realimentación o *feedback*.
  - Reconocimientos o recompensas.

- **Componentes:** Elementos específicos que pueden variar su tipo y cantidad [1].
  - Logros y avances.
  - Avatares.
  - Insignias.
  - *Boss*, Finalización de niveles.
  - Combates.
  - Desbloqueos, permitir desbloquear contenido.
  - Niveles.
  - Formación de equipos.
  - Puntos.
  - Tablas de clasificación.
  - Pruebas.

3.1.4 Elementos de diseño de juegos:

- Patrones de diseño de interfaz, diseño de la interacción, componentes y soluciones para un problema determinado.
- Mecánicas y diseño de patrones del juego.
- Principios y diseño de patrones del juego.
- Modelos de juegos.
- Métodos.

[1] [2] [51].

3.1.5 Categorías de gamificación:

- **Interna:** En el contexto de una organización, el enfoque se centra en mejorar la motivación interna como medio para aumentar la motivación y el compromiso de los empleados.
- **Externa:** En el contexto empresarial, se implementa un enfoque externo con el objetivo de fortalecer las relaciones entre la empresa y sus clientes, buscando involucrarlos de manera activa y mejorar su satisfacción y fidelidad.
- **Cambio de comportamiento:** El enfoque de cambio de comportamiento tiene como objetivo promover la adopción de nuevos hábitos en la población, ya sea fomentando la elección de opciones más saludables o rediseñando la estructura de las clases para fomentar un mayor aprendizaje.

[1] [2] [51].

3.1.6 Mecanismos de gamificación:

- **Sistemas basados en PIC (Puntos Insignias Clasificación):**
  - Puntos, se otorgan cuando se realiza una tarea proporcionando sensación de progreso en el usuario y ofrece información valiosa a los desarrolladores.
  - Insignias, es una representación visual de un logro dentro del sistema gamificado, representan destrezas especificas alcanzadas o desarrolladas.
  - Tabla de clasificación, permite relacionar el progreso obtenido de un usuario respecto a otro, esto sirve como instrumento motivador.

  [1].
- **Sistema basado en puntos:**
  - Puntos de experiencia: Tener mejores bonificaciones.
  - Puntos reembolsables: Comprar mejoras con los puntos.
  - Puntos de habilidad: Mejora el personaje a medida de desarrollo.
  - Puntos de karma: Se crea un camino dependiendo la conducta.

  [1].
- **Sistemas basados en niveles:** Indicativo de progreso.
  - Estado, proporciona información acerca del estado del usuario y el dominio de este en el sistema.
  - Progreso, se refiere a la posición en la que se encuentra un usuario en el sistema.

  [1].
- **Sistemas basados en competición:** Confrontar usuarios para determinar cual tiene un mejor desarrollo de habilidades y conocimientos para alcanzar un objetivo [1].
- **Sistemas basados en cooperación:** Aliar usuarios para que en conjunto se enfrenten a un problema en común [1].

3.2 Desarrollo Multiplataforma

De forma general, las aplicaciones móvil o de cualquier otro tipo de plataforma deben crearse utilizando diferentes marcos y lenguajes, sin embargo, el desarrollo multiplataforma, propone que se puede usar un marco y un lenguaje para crear aplicaciones para cualquier plataforma. "El desarrollo de software multiplatafoma se refiere a el diseño y creación de aplicaciones de software las cuales tienen la capacidad de ejecutarse en múltiples sistemas operativos o plataforma" [52].

3.2.1 Diferencias y ventajas de la programación multiplataforma en *Flutter* en comparación a la programación nativa

Se establecen las ventajas y diferencias de la programación multiplataforma y la programación nativa.

	Programación Multiplataforma en Flutter	**Programación en una sola plataforma**
Soporte de plataformas	Flutter es compatible con múltiples plataformas, como iOS, Android, web y escritorio.	La programación en una sola plataforma generalmente se limita a una plataforma específica, como iOS o Android.
Desarrollo ágil	Flutter permite desarrollar aplicaciones para varias plataformas simultáneamente, lo que acelera el tiempo de desarrollo.	La programación en una sola plataforma puede requerir el desarrollo separado de aplicaciones para cada plataforma, lo que puede llevar más tiempo.
Consistencia de UI	Flutter ofrece una interfaz de usuario consistente en todas las plataformas, lo que facilita el mantenimiento y la coherencia visual.	La programación en una sola plataforma puede requerir la adaptación de la interfaz de usuario para que se ajuste a las pautas de diseño de cada plataforma, lo que puede generar diferencias visuales.
Rendimiento	Flutter utiliza el motor de renderizado Skia y permite un rendimiento nativo de alta velocidad en todas las plataformas.	La programación en una sola plataforma puede aprovechar las características y optimizaciones específicas de cada plataforma, lo que puede resultar en un rendimiento más ajustado a la plataforma objetivo.
Comunidad y recursos	Flutter tiene una comunidad activa y en crecimiento, con una amplia variedad de recursos, documentación y paquetes disponibles.	La programación en una sola plataforma puede tener una comunidad más específica y recursos limitados en comparación con una plataforma más popular como Flutter.
Mantenimiento	Flutter simplifica el mantenimiento ya que los cambios y actualizaciones se aplican a todas las plataformas simultáneamente.	La programación en una sola plataforma puede requerir el mantenimiento individual de cada aplicación, lo que puede llevar más tiempo y recursos.

Tabla 3.1: Diferencias y ventajas de programación multiplataforma en *Flutter* vs programación en una sola plataforma. Fuente: Autoría propia.

3.2.2 *Flutter*

Flutter es un *fremwork* de desarrollo de aplicaciones móviles de código abierto creado por *Google*. El cual permite a los desarrolladores crear aplicaciones nativas de calidad para *iOS* y *Android* de una sola base de código. En lugar de utilizar lenguajes de programación nativos como *Swift* o Java, *Flutter* utiliza el lenguaje de programación *Dart*, que también fue desarrollado por *Google*. *Flutter* proporciona un conjunto de *widgets* personalizables y flexibles que permiten crear interfaces de usuarios atractivas y fluidas [5].

Una de las principales ventajas de *Flutter* es su capacidad para realizar un "*hot reload*" o "*hot restart*", lo que significa que los cambios realizados en el código se reflejan rápidamente en la aplicación en ejecución, lo que acelera significativamente el ciclo de de desarrollo y permite una interacción más rápida. Además, *Flutter* cuenta con una amplia comunidad de desarrolladores y una gran cantidad de recursos y paquetes disponibles, lo que facilita la adopción y la resolución de problemas [6].

En resumen, *Flutter* es un *framework* versátil y potente que permite a los desarrolladores construir aplicaciones móviles nativas de alto rendimiento con una productividad mejorada y una experiencia de desarrollo más fluida.

3.2.3 *Dart*

Dart es un lenguaje de programación desarrollado por *Google*. Fue diseñado con el objetivo de ser un lenguaje moderno, eficiente y fácil de aprender, adecuado tanto para el desarrollo de aplicaciones móviles como para la creación de aplicaciones web y de escritorio [53].

Dart es un lenguaje orientado a objetos con características de tipado estático, lo que significa que las variables y expresiones deben ser declaradas con un tipo de especifico. Sin embargo, *Dart* también admite la inferencia de tipos, lo que permite omitir la declaración explícita de tipos en algunas ocasiones. Una de las principales características de *Dart* es su capacidad para compilar tanto a código nativo como a *JavaScript*. Esto significa que se puede utilizar *Dart* para crear aplicaciones que se ejecutan de forma nativa en dispositivos móviles (a través del kit de desarrollo de software de *Flutter*) o que se ejecuten en navegadores web [53].

Dart también ofrece un conjunto amplio de bibliotecas estándar que abarcan una amplia gama de funcionalidades, desde manipulación de cadenas y colecciones hasta entrada y salida de archivos. Además, cuenta con un sistema de gestión de paquetes llamado *Pub*, que facilita la incorporación de bibliotecas de terceros los proyectos [53].

Uno de los *frameworks* de desarrollo más populares basados en *Dart* es *Flutter*. Utilizando *Dart* como su lenguaje principal se ha vuelto muy popular en el desarrollo de aplicaciones multiplataforma.

3.3 Inteligencia Artificial

La inteligencia artificial se refiere a la capacidad de las máquinas para emplear algoritmos, adquirir conocimiento a través de datos y utilizar dicho conocimiento para tomar decisiones de manera similar a los seres humanos. Sin embargo, los dispositivos impulsados por IA no requieren descanso y son capaces de procesar grandes cantidades de datos simultáneamente [45]. Además, el desarrollo de errores es significativamente menor en las máquinas que realizan las mismas tareas que sus contrapartes humanas. En resumen, la inteligencia artificial se refiere a la capacidad de las máquinas para simular o imitar el comportamiento humano al realizar tareas como reconocimiento de voz, toma de decisiones, procesamiento del lenguaje natural y visión por computadora [44].

La IA se basa en algoritmos y modelos matemáticos que permiten a las maquinas aprender de los datos y mejorar en su rendimiento mediante la experiencia. Por consecuente existen diferentes enfoques dentro de la inteligencia artificial, como el aprendizaje automático (*machine learning*), el aprendizaje profundo (*deep learning*), la lógica difusa (*fuzzy logic*) y la visión computacional [46].

Uno de los enfoques clave que tiene la inteligencia artificial es el aprendizaje automático basado en la construcción de algoritmos y modelos estadísticos que permiten a las máquinas reconocer patrones y tomar decisiones basadas en los datos. El aprendizaje profundo, a su vez, es una técnica de aprendizaje automático que utiliza redes neuronales artificiales para simular el funcionamiento del cerebro humano y obtener un mayor nivel de precisión en el reconocimiento de patrones y en la toma de decisiones [46].

La inteligencia artificial tiene una amplia gama de aplicaciones en diversos campos, como la medicina, la industria, la atención al cliente, la conducción autónoma, la traducción automática, los asistentes virtuales, los sistemas de recomendación, entre otros. A medida que avanza la tecnología la inteligencia artificial continúa evolucionando y se espera que tenga un impacto significativo en nuestra sociedad y en la forma en que se interactúa con las máquinas y los sistemas automatizados [46].

La inteligencia artificial (IA) es un campo amplio que abarca diferentes enfoques y técnicas para desarrollar sistemas capaces de realizar tareas que requieren de inteligencia humana. A continuación se presenta algunos de los tipos mas comunes de la inteligencia artificial.

3.3.1 IA basada en reglas

Esta se trata de un enfoque clásico de inteligencia artificial en que se utilizan reglas lógicas para representar el conocimiento y tomar decisiones. Estas reglas son definidas por expertos humanos y se utilizan para resolver problemas específicos [54].

3.3.2 Aprendizaje automático (*Machine Learning*, ML)

El aprendizaje automático es una rama de la IA que se enfoca en desarrollar algoritmos y modelos capaces de aprender a partir de datos. Estos algoritmos pueden identificar patrones y realizar predicciones o tomar decisiones basadas en esos patrones. El aprendizaje automático incluye técnicas como las redes neuronales, los árboles de decisiones, las máquinas de vectores de soporte, entre otros [54].

3.3.3 Aprendizaje profundo (*Deep Learning*)

El aprendizaje profundo es una rama del aprendizaje automático que se enfoca en el uso de redes neuronales artificiales con múltiples capas. Estas redes tienen la capacidad de aprender de manera automática representaciones de datos complejos y llevar a cabo diversas tareas, como reconocimiento de imágenes, procesamiento del lenguaje natural y aplicaciones de visión por computadora[54].

3.3.4 IA basada en conocimiento

Este enfoque utiliza una base de conocimientos estructurada para resolver problemas específicos. La base de conocimientos contiene información y reglas que representan el conocimiento de expertos en un dominio particular. La IA basada en conocimiento se utiliza en sistemas expertos, asistentes visuales y sistemas de reconocimiento [54].

3.3.5 IA evolutiva

Esta rama de la inteligencia artificial se basa en algoritmos inspirados en la teoría de la evolución biológica. Utiliza algoritmos genéticos, programación genética y estrategias de búsqueda evolutiva para resolver problemas complejos y encontrar soluciones óptimas [54].

3.3.6 IA basada en agentes

En este enfoque, se considera a la IA como un agente que percibe su entorno y toma acciones para maximizar ciertos objetivos. Los agentes inteligentes pueden interactuar con su entorno, aprender de la retroalimentación y adaptarse a nuevas situaciones [54].

3.4 Bases de datos

Una base de datos es un conjunto de datos relacionados entres sí, que se almacenan de forma estructurada y accesible para su posterior consulta, uso y manipulación. Está diseñada para permitir la gestión eficiente de grandes cantidades de información, brindando un método para almacenar, recuperar, modificar y eliminar datos de manera sistemática [12].

Las bases de datos se utilizan ampliamente en diversos campos, como negocios, educación, ciencia, gobierno y muchos otros, para almacenar y administrar información relevante. Pueden contener valores de diferentes tipos, como de tipo texto, de tipo entero o números, imágenes, audios, vídeo, entre muchos otros [12].

Las bases de datos están compuestas por tablas o colecciones de datos estructurados, que a su vez están formadas por filas (registros) y columnas (campos). Cada registro representa una entidad o elemento específico, y cada campo contiene información particular sobre ese elemento. Existen diferentes tipos de bases de datos, como las bases de datos relacionales, donde se establecen relaciones entre las tablas mediante llaves primarias y foráneas; las bases de datos no relacionales, permiten una mayor flexibilidad en la estructura y el almacenamiento de los datos; y las bases de datos en memoria, mantienen los datos en la memoria principal del sistema para un acceso rápido [12].

En resumen, las bases de datos son herramientas fundamentales para organizar y gestionar grandes volúmenes de información de manera eficiente, lo que facilita la manipulación y análisis de datos para obtener información valiosa y respaldar la toma de decisiones.

3.4.1 *Firebase*

Firebase es una plataforma de desarrollo de aplicaciones móviles y *web* creada por *Google*. Proporciona un conjunto de servicios y herramientas que facilitan la creación y el desarrollo de aplicaciones web y móviles, permitiendo a los desarrolladores centrarse en la lógica de la aplicación sin preocuparse por la infraestructura subyacente [10].

Firebase ofrece una amplia gama de servicios, entre los cuales se destacan los siguientes:

- **Base de datos en tiempo real:** Se ofrece una base de datos en tiempo real en la nube que permite sincronizar y almacenar datos en tiempo real en todos los clientes conectados. Es una base de datos no relacional que proporciona actualizaciones instantáneas a los datos sin necesidad de recargar la página [55].
- **Autenticación de usuarios:** *Firebase* proporciona una plataforma de autenticación de usuarios que posibilita a los desarrolladores administrar de manera segura y fácil el proceso de inicio de sesión, registro y gestión de usuarios. Ofrece métodos de autenticación que incluyen correo electrónico y contraseña, así como la posibilidad de utilizar redes sociales como *Google*, *Facebook* y *Twitter*. [10].
- **Almacenamiento en la nube:** Proporciona un servicio de almacenamiento en la nube que permite a los desarrolladores almacenar y servir archivos estáticos, como imágenes, vídeos, documentos, entre otros. Proporciona una forma de carga y descarga de archivos desde la aplicación [55].
- **Mensajería en la nube:** *Firebase Cloud Messaging* (FCM) permite el envío de notificaciones *push* a dispositivos móviles y navegadores web. Los desarrolladores pueden enviar mensajes personalizados a grupos específicos de usuarios o a todos los usuarios de la aplicación [10].
- ***Hosting web*:** Firebase ofrece alojamiento web estático para aplicaciones web. Los desarrolladores pueden alojar y servir fácilmente su sitio web estático, *HTML*, *CSS* y archivos *JavaScript* directamente desde *Firebase* [10].
- **Funciones en la nube:** *Firebase Functions* permite implementar código de *backend* sin servidor que se ejecuta en respuesta a eventos específicos, como cambios en la base de datos o llamadas *HTTP*. Es útil para ejecutar lógica personalizada del lado del servidor [10].

3.5 *Python*

Python es un lenguaje de programación interpretado y de alto nivel que fue desarrollado por *Guido van Rossum* y presentado por primera vez en 1991. Se caracteriza por su sintaxis clara y fácil de leer, lo que lo ha convertido en una opción muy popular entre los programadores.[56]. Es un lenguaje versátil que se utiliza en una amplia variedad de aplicaciones, desde desarrollo web y científico hasta automatización de tareas y *scripting*. Algunas de las características de *Python* incluyen:

- **Sintaxis clara y legible:** *Python* utiliza sintaxis simple y fácil de entender, lo que facilita la lectura y escritura de código [56].
- **Interpretado** *Python* no necesita ser compilado antes de ejecutarse. Un interprete ejecuta directamente el código fuente , lo que permite un desarrollo rápido y una depuración más sencilla [56].
- **Orientado a objetos:** *Python* admite la programación orientada a objetos, lo que significa que permite la definición y creación de clases y objetos [56].
- **Amplia biblioteca estándar:** *Python* incluye una biblioteca estándar completa que ofrece una amplia selección de módulos y funciones predefinidos, lo que simplifica el proceso de desarrollo de diversas aplicaciones. [56].
- **Multiplataforma:** *Python* es compatible con varios sistemas operativos, como *Widows*, *macOS* y Linux, lo que permite que el código sea portátil y se ejecute en diferentes entornos[56].

3.6 *Serverless cloud functions*

Las funciones *serverless* en la nube (*Serverles cloud functions*) son una forma de computación en la nube donde los desarrolladores pueden ejecutar código sin preocuparse por la infraestructura subyacente. En lugar de tener que aprovisionar servidores o administrar la escalabilidad, las funciones *serverless* permiten a los desarrolladores centrarse en escribir el código de su aplicación [57].

Las funciones *serverless* son pequeñas piezas de código que se ejecutan en respuesta a eventos específicos, como una solicitud *HTTP*, un cambio en una base de datos o una carga de archivos. Estas funciones se ejecutan en un entorno aislado y se escalan automáticamente según la demanda, lo que significa que los desarrolladores solo pagan por el tiempo de ejecución real de sus funciones en lugar de mantener servidores en funcionamiento todo el tiempo [57].

3.6.1 *AWS lambda*

AWS Lambda es un servicio de computación sin servidor proporcionado por *Amazon* Web *Services (AWS)*. Permite ejecutar el código en la nube sin tener que aprovisionar ni administrar servidores, Con *AWS Lambda*, se puede ejecutar fragmentos de código en respuesta a eventos, como cambios en datos almacenados en *Amazon S3*, actualizaciones de tablas de bases de datos de *Amazon DynmoDB* o llamadas a *API de Amazon Gateway* [11].

Además, *AWS Lambda* admite varios lenguajes de programación, incluyendo *Phyton*, *JavaScript*, *Java*, *C#*, *Ruby* y *Go*, lo que brinda flexibilidad para desarrollar aplicaciones en el lenguaje de preferencia.

3.6.2 *Gateway AWS*

Gateway AWS se refiere a un conjunto de de servicios que actúan como puerta de enlace entre diferentes recursos y servicios en la nube de *AWS* y el mundo exterior. Estos *gateways* permiten la comunicación bidireccional segura entre entornos locales y los servicios en la nube de *AWS* [58].

Existen varios tipos de *gateway* en *AWS*, cada uno diseñado para un propósito especifico:

- ***AWS API Gateway:*** Proporciona una forma de crear, publicar, mantener, monitorear y proteger *APIs* para permitir el acceso a los servicios y aplicaciones de *AWS*, así como a otros sistemas [58].
- **Interpretado** *Python* no necesita ser compilado antes de ejecutarse. Un interprete ejecuta directamente el código fuente , lo que permite un desarrollo rápido y una depuración más sencilla [58].
- AWS Transit Gateway: Es un servicio que permite conectar redes locales (*on-premises*) con la nube de *AWS*, facilitando la comunicación entre diferentes VPC, cuentas *AWS* y redes locales [58].
- ***AWS Direct Connect Gateway:*** Permite conectar de manera segura y privada redes locales a los servicios de *AWS* utilizando conexiones *Direct Connect*. Estas conexiones ofrecen un ancho de banda alto y una latencia baja, lo que resulta útil para aplicaciones que requieren una conectividad confiable y de alto rendimiento [58].
- ***AWS VPN Gateway:*** Proporciona una forma de establecer conexiones *VPN (Virtual Private Nerwork)* entre redes locales y la nube de *AWS*. Esto permite la extensión segura de la red local a la nube y el acceso remoto a recursos en *AWS* [58].

3.7 *ChatGPT API*

API de *ChatGPT* es una interfaz de programación de aplicaciones proporcionada por *OpenAI* que permite a los desarrolladores interactuar con el modelo de lenguaje ChatGPT. El cual es una versión especifica de *GPT (Generative Pre-trained Transformer)* diseñada para generar respuestas coherentes y contextualmente relevantes en conversaciones [59].

La *API* de *ChatGPT* permite a los desarrolladores enviar solicitudes *HTTP* al modelo de *ChatGPT* y recibir respuestas generadas por el modelo. Se puede enviar mensajes de texto o instrucciones al modelo y obtener una respuesta generada por el modelo en función de la entrada proporcionada. Para utilizar la *API*, se necesita una clave

de *API* proporcionada por *OpenAI*. Se puede enviar solicitudes a la *API* utilizando bibliotecas de programación como *Python* o *URL*, esta *API* es útil para una variedad de casos de uso, como la construcción de *chatbots*, sistemas de respuesta automática, asistentes virtuales y más [59].

Capítulo 4

Diseño Metodológico

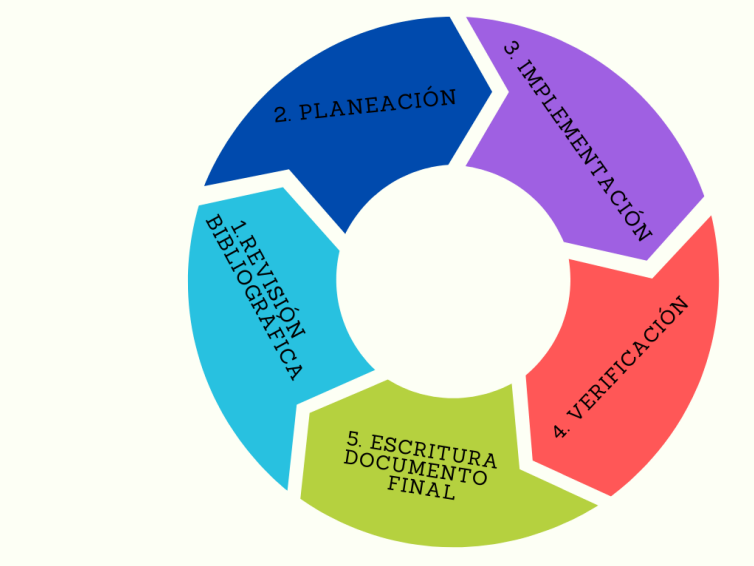

Figure 4.1: Metodología *SCRUM*. Fuente: Autoría propia.

Con la finalidad de cumplir el objetivo general del proyecto, que daba respuesta a la pregunta problema planteada, fue importante utilizar un método de investigación y verificación que se ajustara a este. Teniendo en cuenta lo anterior, se propuso una metodología *SCRUM*. Debido a que esta metodología es ampliamente utilizada en el desarrollo de software, se enfoca en la entrega temprana y continua de incrementos de trabajo completados [3].

Esto significa que, en lugar de esperar hasta el final del proyecto para entregar resultados, esta metodología se basa en la entrega de incrementos de funcionalidad utilizables en intervalos regulares. Lo que permite obtener retroalimentación temprana y garantizar que se entregue valor de manera más rápida. Teniendo en cuenta esto, *SCRUM* fomenta una mentalidad de mejora continua el cual al final de cada sprint, el equipo reflexiona sobre lo que funcionó bien y lo que podría mejorarse en términos de procesos, colaboración y resultados. Esto permite que el equipo aprenda y adapte con el tiempo, lo que puede conducir a una mayor eficiencia y calidad en el trabajo realizado[3].

4.1 Fases y Metodología

Ya que se definió el método de investigación y de verificación como se muestra en la figura 4.1, el enfoque que se planeó dar al desarrollo del proyecto es en esencia los siguientes pasos:

1. Se crea una lista de elementos o características que se deben desarrollar, ordenándolos por prioridad.
2. El equipo selecciona una cantidad de elementos del producto que se compromete a completar durante el próximo Sprint por medio del desarrollo de tareas más pequeñas.
3. Se fija un periodo de tiempo fijo llamado Sprint durante el cual el equipo trabaja en la implementación de las tareas seleccionadas.
4. A medida que el equipo trabaja en las tareas, se va generando un incremento de producto funcional que se revisa y valida al final del Sprint.
5. El equipo realiza una reunión de retrospectiva del Sprint donde analizan su desempeño y hacer las correcciones.
6. Repetir el proceso con cada elemento seleccionado del producto y mejorando continuamente el proceso.

Por las condiciones planteadas de entrega fue necesario acoplar esta metodología en cuatro fases las cuales, en las fases uno y dos se agrupan los tres primeros pasos para la utilización de la metodología *SCRUM*, en la tercera fase se implementó el cuarto paso y para la curta fase se agruparon el quito y sexto paso.

Capítulo 5

Desarrollo Conceptual

5.1 Fase I: Investigación del método y técnicas para gamificar y definir los temas del área de inteligencia artificial a tratar.

Como se evidencia en el capítulo del marco teórico primero se hace una investigación acerca de la definición especifica y en que consiste el termino gamificar, con el propósito de encontrar una herramienta innovadora que permita enseñar conceptos básicos de inteligencia artificial a personas que deseen tener una introducción en esta área, así que se llegó a la deducción que la gamificación es precisamente eso, una herramienta que permite mejorar el proceso enseñanza aprendizaje debido a que incluye mecánicas de juego en entornos de enseñanza, permitiendo de esta forma un resultado preceptivamente más agradable y con mejores resultado por parte del sujeto a prueba a la captación, retención, capacitación y entendimiento de nuevo conocimiento debido a que se generan situaciones de recompensas creando un habiente competitivo que permite al usuario involucrarse más, fomentando la sensación de satisfacción por el cumplimiento de retos y recompensándolo para crear aptitudes de compromiso. Esta conclusión se ve soportada por los ejemplos presentados en la sección de Gamificación en el capitulo del estado del arte del presente documento [39].

Las técnicas de gamificación se refieren a enfoques y tácticas empleadas para incorporar elementos y mecánicas de los juegos en contextos que no son necesariamente de entretenimiento. Su objetivo principal es motivar, comprometer y aumentar la participación de las personas. En el ámbito educativo, estas técnicas se emplean para hacer el proceso de aprendizaje más atractivo, interactivo y efectivo [60]. Algunas de las técnicas de gamificación más utilizadas incluyen:

- **Puntos y niveles:** Los estudiantes pueden ganar puntos por completar tareas, responder preguntas correctamente o alcanzar ciertos hitos. Estos puntos se acumulan y permiten desbloquear niveles o desafíos más avanzados, lo que brinda una sensación de progreso y logro [60] [1].
- **Tablas de clasificación:** Se muestra un *ranking* o tabla de clasificación donde los estudiantes pueden comparar su desempeño con el de sus compañeros. Esto fomenta la competencia amistosa y motiva a mejorar para subir de posición en la tabla [60].
- **Desafíos y misiones:** Se plantean retos y misiones que los estudiantes deben superar. Estos desafíos pueden requerir la resolución de problemas, la realización de actividades específicas o el logro de ciertos objetivos. Al completarlos, se obtienen recompensas y se avanza en el juego [2].
- **Recompensas virtuales:** Se otorgan premios virtuales, como medallas, insignias o trofeos, por alcanzar metas o realizar acciones destacadas. Estas recompensas no solo brindan reconocimiento, sino que también pueden desbloquear contenido adicional o ventajas en el juego [2].
- **Historia y narrativa:** Se crea una historia o narrativa que envuelve al estudiante en el juego. A medida que avanza, se van revelando nuevos capítulos o elementos de la historia, lo que genera curiosidad y motivación para continuar aprendiendo [60].
- **Personalización:** Se permite que los estudiantes personalicen su avatar, personaje o entorno dentro del juego. Esto les da un sentido de propiedad y les permite expresarse de manera individual, lo que aumenta su compromiso con la experiencia [60] [2].

- **Cooperación y trabajo en equipo:** Se promueve la colaboración entre estudiantes, ya sea a través de desafíos grupales o actividades en las que deben trabajar juntos para lograr un objetivo común. Esto fomenta el compañerismo, el intercambio de conocimientos y el desarrollo de habilidades sociales [60].

En consecuencia, se toma el modelo y varias de las técnicas de gamificación entre las cuales se escogieron "Puntos y Niveles", "Desafios y Misiones", "Recompensas virtuales" y "Personalización" debido al enfoque que se plantea introducirlas en el aplicativo multiplataforma que permita enseñar conceptos básicos de inteligencia artificial. Por ende, primero se debía definir los elementos que eran necesarios para que se considerara a un aplicativo de enseñanza como gamificado.

Mediante la investigación bibliográfica se identificaron las características y elementos de la gamificación los cuales determinan que los elementos esenciales para utilizar esta herramienta son los puntos, niveles, clasificación, retos, insignias y premios. El procedimiento para gamificar sería entonces primero definiendo un objetivo, después transformar el aprendizaje en juego para que posteriormente se pueda proponer un reto en conjunto con el establecimiento de las reglas y normas del juego para concluir creando las recompensas.

Si se tienen juegos dentro del aplicativo es necesario que tengan los elementos llamados dinámicas, mecánicas y componentes como se evidencia en el marco teórico del presente documento en la sección de gamificación en el apartado de elementos de los juegos, los cuales definen las características que debe tener un juego, que a su vez es una de las formas más relevantes y aceptadas por los usuarios para identificarlo como un reto. La categoría a la que se quiere enfocar la gamificación para el aplicativo es la Interna puesto que el principal objetivo de esta categoría es motivar al usuario sin necesidad de una interacción con otro que no sea el evaluador de su proceso.

Y por último el mecanismo de gamificación que se planteo es el basado en niveles, debido a que de esta forma se puede llevar el seguimiento del progreso de un usuario con mayor entendimiento para personas que no están involucradas o acostumbradas a mecanismos de gamificación.

Para la segunda parte de esta fase se centra en definir los temas de inteligencia artificial que son mas relevantes para generar una contextualización, una introducción y un primer acercamiento a la IA es por esto que mediante el libro *"Artificial Intelligence: A Modern Approach 4e by Stuart Russell"* se plantean los conceptos base para entender la inteligencia artificial entre los cuales se agruparon en siete conceptos globales. Estos siete conceptos son los que se identificaron como los mas importantes debido a las implicaciones que conlleva el entendimiento y dominio de los mismos, permitiendo a una persona aplicar estos conocimientos a un área específica de la inteligencia artificial y facilitar el entendimiento o aprendizaje de conceptos específicos de una aplicación determinada de la IA.

5.1.1 Introducción a la Inteligencia Artificial

Para enseñar sobre inteligencia artificial es necesario hacer la introducción a esta, en consecuencia se definieron dos conceptos básicos importantes que se deben conocer para entender que es la IA y sus implicaciones, a continuación se presentará su definición y la forma de interactuar con el ser humano.

5.1.1.1 Introducción

Como ya se ha definido en el presente documento en el capitulo de Marco teórico, IA es la capacidad que tienen las maquinas para simular la inteligencia humana para realizar tareas, utilizando algoritmos y modelos matemáticos avanzados para procesar grandes cantidades de datos y tomar decisiones o realizar predicciones.

La inteligencia artificial se ha vuelto omnipresente en nuestra vida cotidiana, desde los asistentes virtuales en nuestros teléfonos inteligentes hasta los sistemas de recomendación en plataformas de transmisión de contenido. También ha tenido un impacto significativo en sectores como la medicina, la automoción, la agricultura y la fabricación, mejorando la eficiencia y generando nuevas oportunidades [31].

Video de la subsección utilizado en el aplicativo `https://youtu.be/Lae9BefikUk`.

5.1.1.2 Agentes Inteligentes

Los agentes inteligentes son entidades capaces de percibir su entorno, procesar información y tomar decisiones para lograr objetivos específicos. Estos agentes utilizan técnicas de inteligencia artificial.

El objetivo de un agente inteligente es maximizar una función de rendimiento o alcanzar una meta específica. Puede adaptarse y aprender de la experiencia para mejorar su rendimiento y lograr mejores resultados [61].

Los agentes inteligentes se utilizan en una amplia variedad de aplicaciones, como sistemas de recomendación, sistemas de control automatizado, asistentes virtuales, robots autónomos, vehículos autónomos, entre otros. Estos agentes están diseñados para interactuar con su entorno de manera inteligente y autónoma, brindando soluciones eficientes y adaptativas a problemas complejos [62].

Video de la subsección utilizado en el aplicativo `https://youtu.be/1wKkFY_JJFg`.

5.1.2 Lógica Proposicional y de Primer Orden

La lógica proposicional y la lógica de primer orden son herramientas fundamentales en la inteligencia artificial para el razonamiento lógico y la representación del conocimiento. Cada una tiene sus propias aplicaciones y utilidades en este campo.

La lógica proposicional es utilizada en la inteligencia artificial para representar y razonar sobre situaciones donde las proposiciones pueden ser verdaderas o falsas, pero no tienen una estructura interna más compleja. Se utiliza para modelar problemas en los que las incertidumbres no son relevantes y se busca evaluar la validez de los argumentos lógicos [62]. Además, la lógica proposicional se emplea en la representación de reglas de producción en sistemas expertos, donde se establecen relaciones causales entre diferentes proposiciones.

5.1.2.1 Introducción a la Lógica Proposicional

La lógica proposicional, también conocida como lógica de enunciados o lógica sentencial, es un sistema formal utilizado en la lógica y la matemática para analizar y razonar sobre proposiciones y sus relaciones lógicas. Se centra en el estudio de las operaciones lógicas básicas y las reglas que gobiernan la combinación de proposiciones [63].

Video de la subsección utilizado en el aplicativo `https://youtu.be/cL6c9SZwnMI`.

5.1.2.2 Inferencia en Lógica Proposicional

- Negación ($\neg$): Niega el valor de verdad de una proposición. Por ejemplo, si "p" representa la proposición "llueve", entonces $\neg$p representaría la negación de esa afirmación, es decir, "no llueve" [62].

- Conjunción ($\wedge$): Representa la operación lógica de "y". Une dos proposiciones y es verdadera solo si ambas proposiciones son verdaderas. Por ejemplo, si "p" representa "hace sol" y "q" representa "hace calor", entonces p $\wedge$ q representa "hace sol y hace calor" [64].

- Disyunción ($\vee$): Representa la operación lógica de "o". Une dos proposiciones y es verdadera si al menos una de las proposiciones es verdadera. Por ejemplo, si "p" representa "es lunes" y "q" representa "es martes", entonces p $\vee$ q representa "es lunes o es martes" [65].

- Implicación ($\rightarrow$): Representa la relación de implicación o condicional. Si "p" representa una proposición y "q" representa otra, entonces p $\rightarrow$ q significa "si p, entonces q". Es falsa solo cuando p es verdadera y q es falsa [62].

- Bicondicional ($\leftrightarrow$): Representa la relación de equivalencia lógica. Si "p" y "q" son proposiciones, entonces p $\leftrightarrow$ q significa "p si y solo si q", es decir, p es verdadera si y solo si q es verdadera [66].

Video de la subsección utilizado en el aplicativo `https://youtu.be/IYI15yf71dk`.

5.1.2.3 Representación del Conocimiento en la Lógica de primer Orden

La representación del conocimiento en la lógica de primer orden se basa en la utilización de símbolos y reglas formales para describir y organizar la información. En la lógica de primer orden, el conocimiento se representa mediante la definición de un lenguaje formal compuesto por símbolos y expresiones bien definidas [62]. Los elementos fundamentales en la representación del conocimiento en lógica de primer orden son:

- Símbolos constantes: Representan objetos o entidades específicas. Por ejemplo, "Juan", "Mesa", "Perro" [67].
- Símbolos variables: Representan objetos genéricos. Se suelen denotar por letras minúsculas. Por ejemplo, "x", "y", "z" [68].
- Símbolos de función: Representan relaciones entre objetos. Se utilizan para expresar propiedades o características de los objetos. Por ejemplo, "PadreDe(x, y)", que representa la relación "x es el padre de y" [62].
- Símbolos de predicado: Representan afirmaciones o proposiciones que pueden ser verdaderas o falsas. Se utilizan para expresar propiedades o relaciones entre objetos. Por ejemplo, "Hombre(x)", que representa la afirmación "x es un hombre" [69].
- Símbolos de cuantificación: Representan la cuantificación universal ($\forall$) y la cuantificación existencial ($\exists$). Se utilizan para expresar la extensión de una afirmación sobre un conjunto de objetos [61].

la representación del conocimiento en la lógica de primer orden se basa en la utilización de símbolos y expresiones formales para representar objetos, relaciones, propiedades y afirmaciones, permitiendo realizar inferencias y razonamientos lógicos sobre el conocimiento representado.

Video de la subsección utilizado en el aplicativo `https://youtu.be/6CTrlR8EeqE`.

5.1.2.4 Razonamiento en Lógica de Primer Orden

El razonamiento en lógica de primer orden se basa en aplicar reglas y principios lógicos para llegar a conclusiones válidas a partir de un conjunto de premisas [62]. A continuación, describiré algunas de las principales formas de razonamiento en lógica de primer orden:

- *Modus Ponens*: Es una regla de inferencia que permite deducir una conclusión a partir de una premisa condicional y la afirmación del antecedente. Si se tiene una premisa de la forma "Si A, entonces B" y la afirmación de A, entonces se puede concluir que B es verdadera [62] [64].
- Generalización Universal ($\forall E$): Permite generalizar una afirmación para todos los elementos de un conjunto. Si se tiene una afirmación de la forma "Para todo x, P(x)" (donde P(x) es una propiedad que depende de x), se puede concluir que P(x) es verdadera para cualquier elemento del conjunto [65].
- Especialización ($\exists I$): Permite afirmar la existencia de un elemento que cumple cierta propiedad. Si se tiene una afirmación de la forma "Existe x tal que P(x)" (donde P(x) es una propiedad que depende de x), se puede concluir que existe al menos un elemento del conjunto para el cual P(x) es verdadera [66].
- Silogismo Hipotético: Es una forma de razonamiento que se basa en premisas condicionales y permite inferir una nueva conclusión. Si se tienen dos premisas condicionales de la forma "Si A, entonces B" y "Si B, entonces C", se puede concluir que "Si A, entonces C" [62].

Video de la subsección utilizado en el aplicativo `https://youtu.be/PYh9-CNvmHE`.

5.1.2.5 Lógica de Descripción

La lógica de descripción es un formalismo lógico utilizado para representar y razonar sobre conocimiento en dominios específicos. Es una extensión de la lógica de primer orden que incorpora constructos adicionales para expresar restricciones y propiedades más complejas [62].

En la lógica de descripción, se utilizan axiomas y reglas para definir conceptos, propiedades y relaciones entre ellos. Algunos de los elementos clave en la lógica de descripción son:

- Conceptos: Representan clases o conjuntos de objetos y se definen mediante restricciones. Por ejemplo, se puede definir el concepto "Mamífero" como la clase de los seres vivos que tienen glándulas mamarias y son de sangre caliente [69].
- Propiedades: Describen características o atributos de los objetos. Pueden ser simples o compuestas. Por ejemplo, se puede definir la propiedad "TieneAlas" para describir si un objeto tiene alas [68].
- Restricciones: Son condiciones que se imponen sobre los conceptos y propiedades para limitar el conjunto de objetos que los satisfacen. Por ejemplo, se puede definir una restricción que diga que un "Ave" es un "Animal" que "TieneAlas" y "PoneHuevos" [62].
- Relaciones: Expresan conexiones entre objetos. Pueden ser binarias, ternarias, etc. Por ejemplo, se puede definir la relación "EsPadreDe" para describir la relación entre dos individuos como padre e hijo [67].

La lógica de descripción permite realizar razonamientos sobre conceptos y relaciones entre ellos. Se pueden realizar consultas y deducciones utilizando inferencias lógicas para extraer conclusiones a partir de las definiciones y restricciones establecidas [62].
Video de la subsección utilizado en el aplicativo `https://youtu.be/gnqmN8qnIcY`.

5.1.3 Búsqueda e Introducción a la Planificación

La búsqueda y la planificación son herramientas clave en la inteligencia artificial que permiten resolver problemas complejos y tomar decisiones en entornos dinámicos. La búsqueda se enfoca en explorar un espacio de estados en busca de una solución, mientras que la planificación se centra en generar secuencias de acciones para lograr un objetivo o cumplir con restricciones específicas [66]. Ambos conceptos son fundamentales en el desarrollo de sistemas inteligentes capaces de resolver problemas en diversos dominios.

5.1.3.1 Introducción a la búsqueda en Inteligencia Artificial

La búsqueda se refiere a la exploración sistemática de un espacio de estados en busca de una solución o un objetivo. En inteligencia artificial, la búsqueda se utiliza para encontrar secuencias de acciones o movimientos que permitan alcanzar una meta o resolver un problema. Esta técnica se aplica en diversas áreas, como los juegos, la resolución de problemas de optimización, la planificación de rutas, entre otros. La búsqueda puede basarse en algoritmos como la búsqueda en anchura, búsqueda en profundidad, búsqueda heurística (como A*), entre otros, que determinan cómo explorar el espacio de estados de manera eficiente [65].
Video de la subsección utilizado en el aplicativo `https://youtu.be/awsxi7odfTQ`.

5.1.3.2 Búsqueda Informada (heurística) y Búsqueda No Informada (ciega)

La búsqueda informada (heurística) y la búsqueda no informada (ciega) son dos enfoques diferentes utilizados en la resolución de problemas mediante algoritmos de búsqueda. A continuación, se describen las características y diferencias de cada uno:

Búsqueda no informada (ciega): En la búsqueda no informada, también conocida como búsqueda ciega, no se utiliza ninguna información adicional sobre el problema o el espacio de búsqueda más allá de la definición básica de las reglas y las posibles acciones disponibles. Los algoritmos de búsqueda no informada exploran el espacio de búsqueda de manera sistemática sin tener en cuenta la estructura o la naturaleza del problema. Ejemplos de algoritmos de búsqueda no informada incluyen la búsqueda en anchura (BFS), la búsqueda en profundidad (DFS), la búsqueda en profundidad limitada (DLS) y la búsqueda de costo uniforme. Estos algoritmos son simples y fáciles de implementar, pero pueden ser ineficientes en espacios de búsqueda grandes o cuando no hay información adicional para guiar la búsqueda [62].

Búsqueda informada (heurística): En contraste con la búsqueda no informada, la búsqueda informada utiliza información adicional, conocida como heurística, para guiar la búsqueda hacia soluciones potencialmente mejores. La heurística es una función que estima cuánto falta para alcanzar el objetivo o qué tan prometedor es un nodo en términos de ser parte de la solución óptima. Los algoritmos de búsqueda informada más comunes son la búsqueda de mejoría iterativa (*Hill Climbing*), el algoritmo A* y el algoritmo de búsqueda voraz (*Greedy Search*). Estos algoritmos utilizan la información heurística para tomar decisiones más inteligentes sobre qué nodos explorar y en qué orden, lo que puede llevar a una búsqueda más eficiente y encontrar soluciones óptimas o cercanas a óptimas

en menos tiempo [64].

Video de la subsección utilizado en el aplicativo `https://youtu.be/hrWc9K3TPdA`.

5.1.3.3 Algoritmos de búsqueda

Los algoritmos de búsqueda son métodos y procedimientos utilizados en ciencias de la computación e inteligencia artificial para explorar y encontrar soluciones en un espacio de búsqueda. Estos algoritmos se aplican cuando se enfrenta un problema en el que es necesario encontrar una secuencia de acciones, un estado objetivo o una configuración deseada a partir de un estado inicial [69].

Los algoritmos de búsqueda exploran sistemáticamente el espacio de búsqueda, tomando decisiones sobre qué estados explorar y en qué orden, utilizando ciertas estrategias y criterios. Estas estrategias pueden ser informadas (heurísticas) o no informadas (ciegas). Algunos ejemplos comunes de algoritmos de búsqueda incluyen:

- Búsqueda en anchura (*BFS*): Explora los nodos en el espacio de búsqueda en orden de nivel, es decir, explora todos los nodos a una distancia "d" antes de explorar los nodos a una distancia "*d+1*" [69].
- Búsqueda en profundidad (*DFS*): Explora los nodos en el espacio de búsqueda en orden de profundidad, es decir, se adentra en un camino específico tanto como sea posible antes de retroceder [67].
- Búsqueda de costo uniforme: Explora los nodos en el espacio de búsqueda en función de los costos acumulativos asociados con cada camino y elige el camino con el costo más bajo [61].
- Algoritmo $A*$: Combina la búsqueda de costo uniforme con una función heurística que estima el costo restante para alcanzar el objetivo. Utiliza la suma del costo acumulado y la estimación heurística para elegir los nodos a explorar [62].

Video de la subsección utilizado en el aplicativo `https://youtu.be/JUSquXxuu3Y`.

5.1.3.4 Introducción a la Planificación

La planificación se ocupa de generar secuencias de acciones que permitan alcanzar un objetivo o cumplir con una serie de restricciones en un entorno determinado. La planificación se utiliza para tomar decisiones en situaciones complejas y dinámicas, donde es necesario determinar qué acciones deben llevarse a cabo en qué momento y en qué orden para alcanzar un objetivo deseado. Los sistemas de planificación pueden utilizar diferentes enfoques, como la planificación basada en metas, la planificación basada en reglas, la planificación basada en restricciones y la planificación basada en modelos [62]. Estos enfoques permiten generar planes o secuencias de acciones que resuelvan problemas complejos en una amplia gama de dominios, como la logística, la robótica, la gestión de proyectos, entre otros.

Video de la subsección utilizado en el aplicativo `https://youtu.be/DKp8m2XGHQU`.

5.1.4 Introducción al Aprendizaje Automático (*Machine Learning*)

El aprendizaje automático es un campo de estudio dentro de la inteligencia artificial que se centra en la creación de algoritmos y modelos que permiten a las computadoras adquirir conocimiento y tomar decisiones basadas en datos, sin necesidad de ser programadas de manera explícita [45]. En lugar de seguir instrucciones específicas, las máquinas aprenden a partir de ejemplos y experiencias [46].

En esencia, el aprendizaje automático se basa en la idea de que las computadoras pueden analizar grandes cantidades de datos, identificar patrones y construir modelos predictivos o descriptivos a partir de ellos. Estos modelos pueden utilizarse para hacer predicciones o tomar decisiones en nuevas situaciones, sin requerir una programación detallada [62]. Existen diferentes enfoques dentro del aprendizaje automático. Algunos de los más comunes son:

- Aprendizaje supervisado: En este enfoque, se proporcionan datos de entrada junto con las respuestas deseadas, y el algoritmo aprende a mapear las entradas a las salidas correspondientes. Por ejemplo, se pueden proporcionar imágenes de gatos y perros junto con etiquetas que indican si son gatos o perros, y el algoritmo aprende a reconocer nuevos animales [54] [62].

- Aprendizaje no supervisado: En este caso, el algoritmo trabaja con datos sin etiquetar y busca patrones o estructuras ocultas en los mismos. No se proporcionan respuestas deseadas. Es útil cuando se desea descubrir agrupamientos o relaciones entre los datos [54] [62].
- Aprendizaje por refuerzo: Aquí, un agente de aprendizaje interactúa con un entorno y recibe retroalimentación en forma de recompensas o castigos según sus acciones. El objetivo es que el agente aprenda a tomar decisiones que maximicen las recompensas a largo plazo [54] [62].

El proceso típico de desarrollo de un modelo de aprendizaje automático implica las siguientes etapas:

1. Recopilación y preparación de datos: Se obtienen los datos relevantes y se preparan para su procesamiento. Esto puede incluir la limpieza de datos, la normalización y la selección de características relevantes [68].
2. Selección del algoritmo: Se elige el algoritmo o modelo que se utilizará para entrenar el sistema. Esto dependerá del tipo de problema y de los datos disponibles [66].
3. Entrenamiento: Se utiliza un conjunto de datos de entrenamiento para ajustar los parámetros del modelo y hacer que se adapte a los datos [65].
4. Evaluación y ajuste: Se evalúa el rendimiento del modelo utilizando datos de prueba o validación. Si es necesario, se realizan ajustes en el modelo para mejorar su precisión [70].
5. Predicción o toma de decisiones: Una vez que el modelo ha sido entrenado y evaluado, se puede utilizar para hacer predicciones o tomar decisiones en nuevos datos [62].

Video de la subsección utilizado en el aplicativo `https://youtu.be/LBB1vg8Objc`.

5.1.5 Introducción al Procesamiento del Lenguaje Natural (NPL)

El procesamiento de lenguaje natural (PLN) es una disciplina de la inteligencia artificial que se concentra en la comunicación y relación entre las computadoras y el lenguaje humano. Su objetivo fundamental radica en capacitar a las máquinas para comprender, interpretar y generar lenguaje natural de manera similar a como lo hacen los seres humanos [46].

El PLN se basa en diversas disciplinas, como la lingüística computacional, la inteligencia artificial y la estadística. Implica el desarrollo de algoritmos y modelos que permiten a las computadoras procesar y analizar texto y discurso en forma de palabras, frases y oraciones [46] [62]. El proceso de PLN consta de varias etapas, que incluyen:

1. Tokenización: en esta etapa, el texto se divide en unidades más pequeñas, como palabras o caracteres, conocidos como "tokens". Esto facilita el análisis posterior [71].
2. Análisis morfológico: se realiza el análisis de las formas y estructuras de las palabras para comprender su raíz, género, número, tiempo verbal, etc [62].
3. Análisis sintáctico: se busca comprender la estructura gramatical y las relaciones entre las palabras en una oración, como el sujeto, el verbo y el objeto [71].
4. Análisis semántico: se intenta comprender el significado de las palabras y las oraciones, considerando el contexto y la interpretación [71].
5. Desambiguación: se resuelven las ambigüedades léxicas y gramaticales que puedan surgir durante el análisis del texto [62].
6. Reconocimiento de entidades: se identifican y clasifican las entidades mencionadas en el texto, como nombres propios, fechas, lugares, organizaciones, etc [46].
7. Extracción de información: se extrae información relevante y se identifican relaciones entre entidades y conceptos en el texto [62].
8. Generación de lenguaje natural: se crea texto en lenguaje natural como respuesta o como generación de texto automática [71].

El PLN se aplica en una amplia gama de aplicaciones, como asistentes virtuales, motores de búsqueda, sistemas de traducción automática, análisis de sentimientos, resúmenes automáticos, análisis de opiniones, *chatbots*, entre otros.

Video de la subsección utilizado en el aplicativo `https://youtu.be/Cn_D45heaWM`.

5.1.6 Introducción a la Visión Computacional

la visión computacional es un campo de la inteligencia artificial que se centra en enseñar a las computadoras a ver y comprender imágenes y vídeos. Utiliza técnicas de procesamiento de imágenes, aprendizaje automático y redes neuronales convolucionales para realizar tareas como detección de objetos, reconocimiento facial y clasificación de imágenes [54]. La visión computacional tiene numerosas aplicaciones prácticas en diferentes industrias y continúa avanzando con nuevos avances tecnológicos [62].

La visión computacional se basa en el procesamiento de imágenes y el análisis de patrones visuales para extraer información útil y tomar decisiones [72]. Algunas de las tareas comunes en visión computacional incluyen:

- Detección de objetos: Identificar y localizar objetos específicos dentro de una imagen o vídeo.
- Seguimiento de objetos: Rastrear y seguir el movimiento de un objeto a lo largo del tiempo en un vídeo.
- Reconocimiento facial: Identificar y reconocer caras en imágenes o vídeos.
- Clasificación de imágenes: Categorizar imágenes en diferentes clases o etiquetas, como perros, gatos, automóviles, etc.
- Segmentación de imágenes: Dividir una imagen en regiones o segmentos más pequeños y asignar etiquetas a cada región.

Para realizar estas funciones, la visión por computadora emplea una combinación de métodos de procesamiento de imágenes, aprendizaje automático y redes neuronales convolucionales (CNN, por sus siglas en inglés). Las redes neuronales convolucionales resultan particularmente eficaces en el procesamiento de imágenes debido a su habilidad para aprender y extraer características visuales relevantes de manera automática [72].

Video de la subsección utilizado en el aplicativo `https://youtu.be/Fo_b8599B78`.

5.1.7 Introducción a *Python*

Es un lenguaje de programación de alto nivel, interpretado y de propósito general.Una de las características distintivas de *Python* es su sintaxis clara y legible, que hace que el código escrito sea fácil de comprender y mantener [73]. Además, se enfoca en la legibilidad del código, fomentando el uso de indentación para estructurar los bloques de código en lugar de utilizar llaves o palabras clave.

Una de las fortalezas de *Python* es su amplia biblioteca estándar, que incluye módulos y funciones para tareas comunes, como manipulación de archivos, *networking*, procesamiento de texto, desarrollo web, entre otros. Esto permite a los programadores ahorrar tiempo y esfuerzo al no tener que escribir código desde cero para muchas tareas básicas [74].

Python es utilizado en una variedad de campos, desde el desarrollo de aplicaciones y sitios web hasta la automatización de tareas, el análisis de datos y la inteligencia artificial. Su popularidad se debe a su facilidad de uso, su enfoque en la legibilidad del código y su versatilidad, lo que lo convierte en una elección común tanto para principiantes como para desarrolladores experimentados [75].

Python es importante para la inteligencia artificial debido a su sintaxis legible, su amplia disponibilidad de bibliotecas especializadas, su comunidad activa, su flexibilidad y su capacidad para facilitar el desarrollo rápido de prototipos. Estas características han hecho de Python el lenguaje preferido para muchos profesionales y entusiastas de la IA.

Video de la subsección utilizado en el aplicativo `https://youtu.be/fsvP9sMSgm0`.

5.1.7.1 Fundamentos de *Python*

Videos de la subsección utilizados en el aplicativo `https://youtu.be/yvX_F1Y45hs`
`https://youtu.be/0ulv1Vno3Aw`.

1. Sintaxis y Variables: [76]

- *Python* utiliza una sintaxis clara y legible con indentación significativa. Los bloques de código se definen mediante sangría en lugar de llaves.
- Puedes asignar valores a variables utilizando el signo igual (=). *Python* es un lenguaje de tipado dinámico, lo que significa que no necesitas declarar el tipo de variable.

2. Tipos de datos básicos: [73]
   - Números: enteros (*int*) y de punto flotante (*float*).
   - Cadenas de texto (*string*): se definen entre comillas simples o dobles.
   - Booleanos (*bool*): verdadero (T*rue*) o falso (*False*).
   - Listas: colecciones ordenadas y modificables de elementos.
   - Tuplas: colecciones ordenadas e inmutables de elementos.
   - Diccionarios: colecciones de pares clave-valor.
3. Operadores aritméticos y de comparación: [75]
   - Operadores aritméticos: suma (+), resta (-), multiplicación (*), división (/), módulo (%), y potencia (**).
   - Operadores de comparación: igualdad (==), desigualdad (!=), mayor (>), menor (<), mayor o igual (>=), menor o igual (<=).
4. Estructuras de control: [76]
   - Condicionales: *if*, *elsif*, *else*.
   - Bucles: *for* y *while*.
5. Funciones: [74]
   - Puedes definir tus propias funciones en *Python*.
   - Una función puede tener parámetros de entrada y retornar un valor.

Videos de la subsección utilizados en el aplicativo `https://youtu.be/xI2_OeIetG0` `https://youtu.be/dUHfN3s6k6M`.

En el aplicativo, las características de gamificación que se planearon desarrollar fueron los Puntos, Niveles, Restos, Insignias y Premios. De esta manera, se presentó un aplicativo gamificado y parametrizado según los lineamientos establecidos. Se empezó con la definición de un objetivo, que en este caso es enseñar conceptos básicos de inteligencia artificial. Luego, se transformó el aprendizaje en un juego mediante retos, los cuales se presentan como preguntas sobre el tema visto a través de contenido audiovisual con múltiples respuestas, pero solo una correcta. Al responder correctamente, los participantes reciben puntos. Esto llevaba al siguiente lineamiento, que consistía en establecer las normas del juego, las cuales requerían responder a la pregunta planteada seleccionando una respuesta. Solo se puede pasar a la siguiente etapa si se selecciona la respuesta correcta. El último lineamiento consistía en las recompensas, que se definieron como las insignias otorgadas por pasar un reto y el poder desbloquear el siguiente nivel.

Además, se diseñó el aplicativo mediante el mecanismo de gamificación definido como sistema basado en niveles, centrado en la categoría interna que consiste en la motivación dentro de una organización. Para precisar las características, se empezó con los Niveles, en los cuales se determinó que para cada nivel sería apropiado contemplar un solo concepto global de Inteligencia Artificial con sus respectivas subsecciones que complementaban el entendimiento de cada concepto. Es por esto que se presentan siete mundos, proporcionando al final de cada subsección un reto (pregunta), entregando puntos cada vez que se pasaba un reto. También se otorgan insignias por cada mundo o nivel pasado.

Por último se implementaron dos juegos interactivos dentro del aplicativo un *"Retro Pacman"* y un "Juego de Memoria" siguiendo los parámetros que deben tener los juegos dentro de una aplicación según la teoría de juegos en la gamificación [1] que en esta misma fase ya se definieron en el tercer párrafo después de las técnicas de gamificación.

5.2 Fase II: Planificación y arquitectura del aplicativo.

Para el correcto desarrollo del aplicativo multiplataforma es importante entrar en la fase de planificación y arquitectura del aplicativo en términos de su diseño e implementación de componentes esenciales dentro del mismo como lo son el modelo de datos, interfaz de usuario y políticas de acceso de usuario.

5.2.1 Arquitectura del aplicativo.

Se establece la arquitectura del dispositivo mediante la implementación de la arquitectura de *software*. La elección de una arquitectura de *software* adecuada es fundamental para garantizar el correcto desarrollo del aplicativo multiplataforma. En este sentido se adopto una arquitectura basada en componentes que permite una separación clara de responsabilidades y una fácil integración de nuevas funcionalidades. Más específicamente se establece una arquitectura de micro-servicios, en la cual el aplicativo se divide en una serie de servicios autónomos y cohesivos . Esta arquitectura facilita la gestión de la complejidad, ya que cada micro-servicio se enfocará en una funcionalidad específica, la escalabilidad es más sencilla de implementar, ya que se puede escalar los servicios individualmente en función de la demanda y optimizar los recursos disponibles.

5.2.1.1 Comunicación entre componentes

Para la comunicación entre los diferentes componentes del sistema se establece el uso de *API RESTful* [4]. Esto permite que los servicios se comuniquen de forma eficiente y desacoplada. Además, permite tener flexibilidad para adaptarse a futuros cambios en los requisitos y facilita la integración con otras aplicaciones o servicios externos.

Dichas *API RESTful* son utilizadas del paquete de *APIs* brindado por *Google* denominados *firebase_auth* para el componente de autenticación, registro e inicio de sesión y *firebase_core* que nos permite conectarnos a diferentes herramientas del kit de *cloud Firebase* como lo es *Firestore* donde está almacenado la base de datos con los datos del aplicativo. Estos paquetes de *APIs* solo soportan plataformas móviles y web, para el desarrollo de *APIs* en entornos de escritorio se utilizan los paquetes *Open Source* de *Fidart_auth* y *Firedart* para poder realizar los componentes de autenticación y enlazamientos con la base de datos respectivamente [4][77][55][7].

5.2.1.2 Componentes de los micro-servicios.

Los componentes individuales de los micro-servicios son definidos dentro del aplicativo con el fin de cumplir con tareas especificas dentro del mismo. Por ende se establecen las tareas del componente, las *APIs* que consume y los procesos que realiza .

Componentes de la Arquitectura		
Componente	APIs Consumidas	Procesos que realiza
Grupos y Usuarios Estudiantes	1. Firedart 2. Firedart_Auth 3. Firebase_core 4. Firebase_auth	1. Añadir, Modificar, Visualizar las colecciones y campos "groups"dentro de los documentos "user_Id" con rol de "Profesor". 2. Añadir y Visualizar los documentos de "user_Id" con rol de "Estudiante"dentro de los documentos individuales de la colección "groups".
Servicios de Usuarios	1. Firedart 2. Firedart_Auth 3. Firebase_core 4. Firebase_auth	1. Modificar, Eliminar y Visualizar los documentos de "user_Id" para poder realizar procesos de cambio de contraseña, re-establecimiento de contraseña y eliminación de cuenta.
Registro y Inicio de de Sesión	1. Firedart 2. Firedart_Auth 3. Firebase_core 4. Firebase_auth	1. Añadir, Modificar y Visualizar los documentos de "user_Id" para poder registrar y iniciar sesión dentro del aplicativo.

Progreso y Calificación	1. Firedart 2. Firedart_Auth 3. Firebase_core 4. Firebase_auth	1. Modificar y Visualizar los documentos de "user_Id" con rol de "Estudiante".
Asistente Virtual	1. API ChatGPT	1. Realizar una petición HTTP con la API de ChatGPT con el valor de entrada del usuario con rol "Estudiante".
Compilador de Python	1. AWS Lambda API	1. Ejecutar la función serverless con parámetro de entrada digitado por el usuario con rol "Estudiante" y esperar al valor de returno de la función.
Información de Mundos	1. Firedart 2. Firedart_Auth 3. Firebase_core 4. Firebase_auth	1. Visualizar todos los campos y colecciones dentro de la colección "worlds" con toda la información acerca de los mundos y sus aspectos de gamificación individuales.

Tabla 5.1: Componentes de la Arquitectura. Fuente: Autoría propia.

En la siguiente tabla se presenta las ventajas de la implementación de una arquitectura de micro-servicios en *Flutter*.

5.2.1.3 Ventajas de la arquitectura micro-servicios en *Flutter*

Ventaja	**Descripción**
Escalabilidad	La arquitectura de micro-servicios permite escalar cada servicio de forma independiente, lo que facilita el manejo de cargas de trabajo variables y el crecimiento de la aplicación.
Flexibilidad tecnológica	Los micro-servicios permiten utilizar diferentes tecnologías y lenguajes de programación para cada servicio, lo que brinda flexibilidad al equipo de desarrollo y la posibilidad de utilizar las herramientas más adecuadas para cada componente de la aplicación.
Desarrollo ágil	La arquitectura de micro-servicios facilita el desarrollo ágil al permitir que diferentes equipos trabajen de manera independiente en cada servicio, acelerando el ciclo de desarrollo y despliegue de nuevas funcionalidades.
Mantenibilidad y modularidad	Los micro-servicios son unidades independientes y desacopladas, lo que facilita la mantenibilidad y el reemplazo o actualización de servicios sin afectar a otros componentes de la aplicación. Además, la modularidad permite una mejor organización del código y facilita la reutilización de componentes.
Resistencia a fallos	En una arquitectura de micro-servicios, si un servicio falla, los demás pueden seguir funcionando de manera independiente, lo que brinda una mayor resistencia a fallos y permite una recuperación más rápida.
Evolución y adaptabilidad	Los micro-servicios facilitan la evolución y adaptabilidad de la aplicación, ya que se pueden agregar, modificar o eliminar servicios sin afectar al resto de la arquitectura. Esto permite mantener la aplicación actualizada y responder rápidamente a los cambios del negocio.

Tabla 5.2: Ventajas de utilizar una arquitectura de micro-servicios en *Flutter* [5][78]. Fuente: Autoría propia.

5.2.1.4 Capa de presentación

En cuanto a la capa de presentación del aplicativo multiplataforma se utiliza un enfoque basado en una arquitectura de cliente-servidor. Se implementó una aplicación cliente que se ejecute en diferentes plataformas, web, dispositivos móviles y escritorio, y se comunique con nuestros servicios a través de la API RESTful mencionada anteriormente. Este enfoque permite compartir la lógica de negocio y mantener una experiencia de usuario coherente en todas las plataformas, minimizando así el esfuerzo de desarrollo y mantenimiento [78][79].

5.2.1.5 Arquitectura de micro-servicios del aplicativo

Al adicionar y definir los diferentes componentes dentro de la arquitectura del aplicativo, *APIs* dentro del marco de arquitectura de micros-servicios. Se establece la arquitectura general del aplicativo en términos de *software* [4].

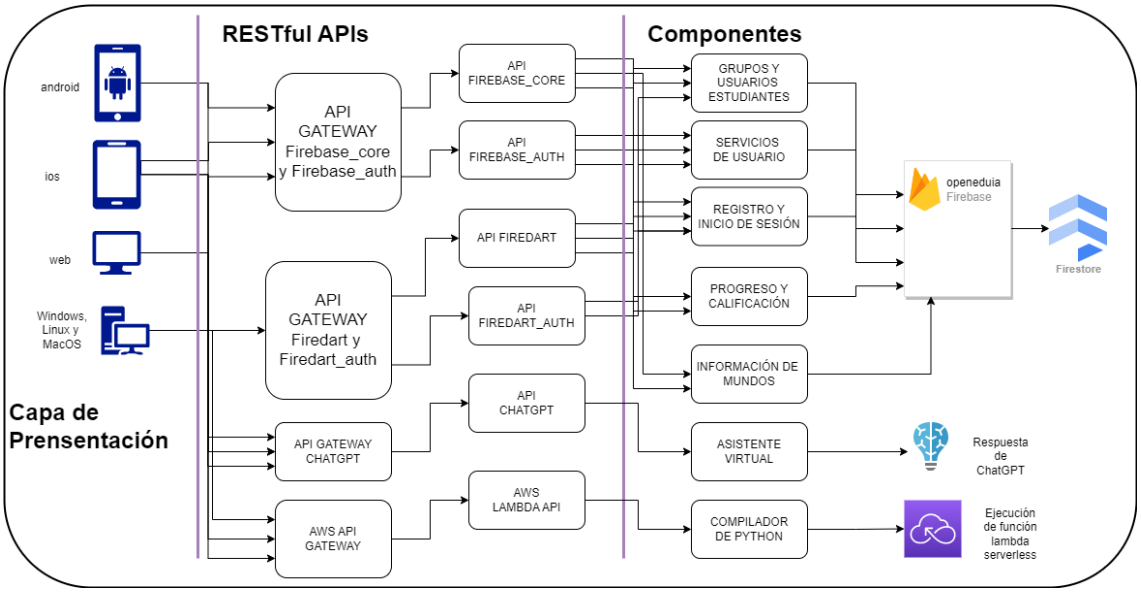

Figure 5.1: Arquitectura de micro-servicios del aplicativo. Fuente: Autoría propia.

5.2.2 Diseño del modelo de datos

Para el correcto diseño e implementación del modelo de datos se establece las limitaciones de la arquitectura del modelo de datos dentro de una base de datos de tipo no relacional. Para ello se establece la arquitectura interna del manejo de datos dentro de la base de datos no relacional utilizada para administrar los datos de usuario y del aplicativo en un servicio de base de datos en la nube denominada *Firestore Database* dentro de la colección de herramientas de desarrollo *Firebase* [55][10].

5.2.2.1 Modelo de datos de usuario

Dentro del servicio de base de datos no relacional el proceso de almacenar datos se desarrolla mediante el uso de colecciones y documentos dentro del espacio asignado dentro de *Firebase*, para ello se define la colección *users* donde se almacenarán los documentos individuales con la información del usuario. También se describe los campos dentro dentro de cada documento con la información del usuario.

Posteriormente al tratarse de un modelo de datos basado en una base de datos no relacional se añaden colecciones y diferentes campos dependiendo del campo *rool* que describe que tipo de rol va a tener el usuario dentro del aplicativo habiendo solo dos opciones disponibles “Profesor” y “Estudiante” que dentro del campo pueden poseer los valores *teacher* y *student* respectivamente . El documento se actualiza con nuevos campos dependiendo del tipo de rol escogido en el campo *rool*, si el campo tiene un valor igual a *teacher* el documento se actualizará con un nuevo campo y además creara una nueva colección denominada *students_groups* donde se encontra todos los grupos creados por el usuario dentro de documentos de manera individual (Ver Anexo A).

Campos dentro del documento	Tipo	Valor	Descripción
Id del Documento	String	userId	Número de identificación generado por FireAuth en el proceso de registro de usuario.
email	String	Ingresado por el usuario.	Dirección de correo electrónico digitado por el usuario en el proceso de registro de usuario.
first_name	String	Ingresado por el usuario.	Primer nombre del usuario digitado por el usuario en el proceso de registro de usuario.
last_name	String	Ingresado por el usuario	Apellido del usuario digitado por el usuario en el proceso de registro de usuario.
rool	String	teacher / student	Tipo de rool seleccionado por el usuario en el proceso de registro de usuario.
icon	String	Ingresado por el usuario.	Imagen de icono de usuario dentro del aplicativo.

Tabla 5.3: Campos del documento *user* dentro de la colección *users*. Fuente: Autoría propia.

5.2.2.2 Modelo de datos de usuario con rol de "Estudiante"

Dependiendo del valor de campo *rool* dentro del documento de cada usuario se establecen los diferentes campos. En el caso de que el campo *rool* este definido con el valor *student* dentro del documento del usuario este responderá a que es una cuenta tipo “Estudiante” y por ende serán agregados diferentes campos dentro del mismo documento (Ver Anexo B).

Campos dentro del documento	Tipo	Valor	Descripción
challenges_count	String	0	Contiene el valor de la cantidad de desafios dentro del aplicativo el estudiante ha realizado.
subsection_progress	String	0	Establece la sub-sección la cual el estudiante esta cursando.
world_progress	String	1	Establece el mundo actual donde el estudiante se encuentra dentro del aplicativo.

Tabla 5.4: Campos adicionales. Fuente: Autoría propia.

5.2.2.3 Modelo de datos de usuario con rol de "Profesor"

Si el campo de *rool* dentro del documento de usuario es establecido con el valor *teacher* se actualiza los campos dentro del documento agregando el siguiente campo dentro del documento.

Campos dentro del documento	Tipo	Valor	Descripción
groups_count	String	0	Contiene el valor de la cantidad de grupos creados por el profesor dentro del aplicativo.

Tabla 5.5: Campo adicional dentro del documento cuando el campo *rool* contiene un valor establecido de *teacher*.

Posteriormente en el proceso de creación de grupos dentro de la pantalla de *Profesor* el usuario puede crear documentos con la información acerca de ese grupo que es almacenado dentro de la colección *students_groups*, esta colección conteniendo los documentos con la información individual de cada uno de los grupos creados por el usuario.

Campos dentro del documento	Tipo	Valor	Descripción
name	String	Ingresado por el usuario.	El nombre del grupo ingresado por el profesor.
students_count	String	0	Cantidad de estudiantes activos dentro del grupo.

Tabla 5.6: Campo adicional dentro del documento cuando el campo *rool* contiene un valor establecido de *teacher*. Fuente: Autoría propia.

Dentro del documento creado en la colección *students_groups* adicional de encontrar los campos anteriores en cada uno de los documentos individuales, el usuario tiene la posibilidad de agregar estudiantes dentro del grupo, así creando una colección *students* dentro del documento individual *group_“número de grupo”*, esta nueva colección posee documentos individuales con la información acerca de los usuarios con campo *rool* con valor *student*.
Al adicionar los campos, colecciones y documentos dentro de esas mismas colecciones se da por terminada la arquitectura del modelo de datos de usuario de usuario con rol de "Profesor" (Ver Anexo C).

Campos dentro del documento	Tipo	Valor	Descripción
id	String	userId	Número de identificación del estudiante dentro del grupo.

Tabla 5.7: Campos dentro de los documentos *group_Id* de la colección *students_groups*. Fuente: Autoría propia.

5.2.2.4 Representación modelo entidad-relación del modelo de datos de los usuarios con sus adaptaciones dependiendo el tipo de rol

Se diseñó el modelo entidad-relación de la colección *users* y se hace su transformación de una base de datos tipo no relacional como la diseñada e implementada en el kit de servicios *cloud* en *Firebase* a un modelo convencional de base de datos de tipo relacional para así poder comprender las relaciones existentes entre las colecciones existentes y sus documentos individuales. Al tratarse de un modelo de base de datos no convencional nos permite adaptar de manera individual los documentos dentro de la colección de manera que puedan existir diferentes adaptaciones a los campos existentes y agregar nuevas colecciones dentro del mismo documento si es necesario, todo esto a partir del campo *rool* que nos permite establecer el tipo de usuario dentro del aplicativo (Ver Anexo D).

5.2.2.5 Modelo de datos de los mundos.

Para el desarrollo del modelo de datos de los mundos se establecieron la cantidad de mundos ha desarrollar dentro de la fase de investigación utilizando como guía de aprendizaje el libro *Artificial Intelligence: A Modern Approach, 4th Global ed.* de los autores *Stuart Rusell* y *Peter Norvig* y el plan de estudios planteado para los estudiantes de básica media en Colombia. Posteriormente de definir los 6 mundos se definió un mundo adicional sobre el lenguaje de programación *Python* basandose en el contenido del libro *Open Source* titulado *Automate the Boring Stuff with Python* de *Al Sweigart* [73][62].

Campos dentro del Documento	Tipo	Valor
world_0"número identificación"	worldId	Identificador del documento.
description	String	Descripción del Mundo.
duration	Time	Duración del Mundo.
name	String	Nombre del Mundo

Tabla 5.8: Campos dentro de los documentos *world_Id* dentro de la colección *worlds*. Fuente: Autoría propia.

A partir de la definición de la cantidad de mundos, sub-secciones dentro de cada mundo se establece los campos dentro del documento (Ver Anexo E). Al establecer los campos dentro del documento se definen las diferentes colecciones dentro de cada documento de manera individual.

Colecciones dentro del documento	Descripción
challenges	Colección con toda información acerca de los desafios del mundo.
comments	Colección con todos los comentarios de los usuarios acerca del mundo.
learning_objectives	Colección con la información del mundo acerca de los objetivos de aprendizaje del mismo.
subsections	Colección con la información acerca de las sub-secciones dentro del mundo.

Tabla 5.9: Colecciones dentro del documento *subsection_Id* dentro de la colección *subsections* de la colección *worlds*. Fuente: Autoría propia.

Se diseña el modelo entidad-relación de la colección *worlds* y se hace su transformación de una base de datos tipo no relacional como la diseñada e implementada en el kit de servicios *cloud* en *Firebase* a un modelo convencional de

base de datos de tipo relacional para así poder comprender las relaciones existentes entre las colecciones existentes y sus documentos individuales (Ver Anexo F).

5.2.2.6 Modelo de datos de los desafíos de los mundos.

Se describe los campos de los documentos de la colección *challenges* con el tipo de campo y descripción de cada uno de los campos. La colección de *challenges* contiene todos los documentos con la información de los desafíos dentro de cada mundo.

Campos del documento	Tipo	Descripción
world_Id	document_Id	Número de identificación del documento dentro de la colección challenges.
name	String	Nombre del desafio dentro del documento
description	String	Descripción del desafio.
difficulty	Int	Dificultad de completar el desafio (0-5).
reward	Int	Número de recompensas al completar el desafio.
tittle	String	Titulo del desafio dentro del documento.

Tabla 5.10: Campos del documento *world_Id* de la colección *worlds*. Fuente: Autoría propia.

Dentro de la colección de *challenges* se realiza el documento *challenge_01* en los mundos *world_01* y *world_02* respectivamente. Dichos documentos hacen referencia a los desafios *Retro Pacman* y *Juego de Memoria* con los valores de campo de descripción, dificultad, recompensas y titulo, dentro de los campos *description*, *difficulty*, *reward* y *tittle* respectivamente (Ver Anexo G).

5.2.2.7 Modelo de datos de los comentarios de los mundos.

Para la creación del modelo de datos de los comentarios dentro de los documentos individuales de los mundos se establecen los nombres de los campos, tipos de campos y descripción general de cada uno de los mismos.

Colección Comments			
Campos del documento	Tipo	Valor	Descripción
comment_Id	document_Id	comment_Id	Número de identificación del documento dentro de la colección "comments"
content	String	Ingresado por el usuario.	Nombre del desafio dentro del documento
icon	String	Icono seleccionado por el usuario.	Descripción del desafio.
id	String	Id del usuario que crea el documento.	Dificultad de completar el desafio (0-5).
user_name	String	Nombres del usuario	Número de recompensas al completar el desafio.
user_lastname	String	Apellidos del usuario.	Titulo del desafio dentro del documento.

Tabla 5.11: Campos del documento *comment_Id* de la colección *comments*. Fuente: Autoría propia.

Cada documento de manera individual es creado por el usuario en el proceso de creación de comentario acerca del mundo.

5.2.2.8 Modelo de datos de los objetivos de aprendizaje de los mundos.

Para el diseño del modelo de datos de los objetivos de aprendizaje dento de los mundos se establecen los campos de los documentos individuales dentro de la colección *learning_objectives*.

Colección learning_objectives			
Campos del documento	Tipo	Valor	Descripción
objective_Id	document_Id	comment_Id	Número de identificación del documento dentro de la colección "learning_objectives"
description	String	Descripción del objetivo	Descripción del objetivo individual dentro del mundo.

Tabla 5.12: Campos del documento *objective_Id* de la colección *learning objectives*. Fuente: Autoría propia.

Se definen los valores de los campos de la colección *learning_objetives* dentro de cada mundo así definiendo los objetivos de aprendizaje de los mismos (Ver Anexo H).

5.2.2.9 Modelo de datos de las sub-secciones de los mundos.

Para el diseño de la sub-secciones dentro de los mundos se establecen los campos dentro de los documentos de la colección *subsections*. Como lo son los campos de identificación de documento , contenido, dificultad, duración de la sub-sección, puntos al completar la sub-sección, titulo, cantidad de videos dentro de la sub-sección y la cantidad de preguntas y quices dentro de la sub-sección, exceptuando el mundo con identificador de documento *world_7* que no posee el proceso de calificación por ende no posee las colecciones relacionadas con ese proceso.

Colección subsections		
Campos del documento	Tipo	Descripción
subsection_Id	document_Id	Identificador del documentodentro de la colección.
content	String	Este campo hace referencia a la descripción de la sub-sección.
difficulty	Int	Dificultad de completar la sub-sección.
duration	Time	Duración en términos de tiempo para completar la sub-sección.
points	Int	Valor en términos de puntos de la sub-sección.
tittle	String	Titulo otorgado a la sub-sección.
videos_count	Int	Cantidad de videos dentro de la sub-sección.
questions_count	Int	Cantidad de preguntas dentro de la sub-sección.
quices_count	Int	Cantidad de quices dentro de la sub-sección.

Tabla 5.13: Campos del documento *subsection_Id* de la colección *subsections*. Fuente: Autoría propia.

Posteriormente se establece los valores de los campos de la colección *subsections* de todos los mundos. Teniendo en cuenta que el documento *world_07* dentro de la colección de *worlds* no contiene el sistema de calificación por ende los campos *questions_count*, *quices_count* y *points* contienen un valor nulo ya que no existen esos campos dentro de ese mundo (Ver Anexo I).

5.2.2.10 Modelo de datos de la colección *questions* dentro de la colección *subsections* al interior de los mundos.

Para el diseño de la colección *questions* dentro de los documentos individuales de la colección *quices* que se encuentran de forma individual en los documentos de los mundos de la colección se establecen los parametros de campo dentro

de los documentos como son nombre del campo, tipo de campo y descripción acerca del campo.

Campos del documento	Tipo	Descripción
question_Id	Document Id	Identificador del documento dentro de la colección "questions".
question	String	Campo con la formulación de la pregunta a responder.
correct_answer	String	Campo con la información de la respuesta correcta dentro del contexto de la pregunta
first_option	String	Campo con la información de la primera opción de respuesta.
second_option	String	Campo con la información de la segunda opción de respuesta.
three_option	String	Campo con la información de la tercera opción de respuesta.
four_option	String	Campo con la información de la cuarta opción de respuesta.

Tabla 5.14: Campos del documento *question_Id* de la colección *questions*. Fuente: Autoría propia.

Posteriormente se establecen los valores de campo dentro de los campos de documentos para cada uno de los mundos dentro de la colección *worlds* (Ver Anexo J).

5.2.2.11 Modelo de datos de la colección *quices* al interior de la colección *subsections* dentro de los mundos.

Para el diseño de la colección *quices* dentro de la colección *subsections* se establecen los parámetros de campo dentro de los documentos individuales como son nombre del campo, tipo de campo y descripción acerca del campo.

Colección "quices"		
Nombre del campo	Tipo	Descripción
quiz_Id	Document Id	Identificador del documento quiz dentro de la colección.
name	String	Nombre otorgado al quiz

Tabla 5.15: Campos del documento *quiz_Id* de la colección *quices*. Fuente: Autoría propia.

Posteriormente se establece los valores de campo para cada documento individual en todos los mundos exceptuando el mundo con identificador *world_07* que no posee el proceso de calificación (Ver Anexo K).

5.2.2.12 Modelo de datos de la colección *videos* al interior de la colección *subsections* dentro de los mundos.

Para el diseño de la colección *videos* dentro de la colección *subsections* se establecen los parámetros de campo dentro de los documentos individuales como son nombre del campo, tipo de campo y descripción acerca del campo.

colección "videos"		
Nombre del campo	Tipo	Descripción
video___Id	Document Id	Identificador del documento video dentro de la colección.
name	String	Nombre del video.
videoUrl	String	Url del video en la plataforma Youtube

Tabla 5.16: Campos del documento *video_Id* de la colección *videos*. Fuente: Autoría propia.

Posteriormente se establece los valores de campo para cada documento individual en todos los mundos (Ver Anexo L).

5.2.3 Despliegue del modelo de datos a *Firebase* y enlace a *Flutter*

En este apartado se describe los pasos necesarios para desplegar la base de datos en *Firebase* y conectarla con el proyecto *Flutter*. Se utiliza *Firebase Firestore*, una base de datos *NoSQL* en la nube [10][55].

5.2.3.1 Creación del Proyecto de Firebase

Inicialmente se crea el proyecto en *Firebase* con nombre *openeduia*. Para ello, se accede al sitio web de *Firebase* (`https://firebase.google.com/`), se inicia sesión con las credenciales y se selecciona "Crear un proyecto nuevo". Se establece el nombre para el proyecto y se desarrollan las instrucciones para completar el proceso de creación [10].

5.2.3.2 Configuración de la base de datos

Posteriormente de la creación del proyecto, se establece el proceso de configuración de la base de datos. Se accede al panel de control de *Firebase* y se selecciona *Firestore Database*. Posteriormente se crea la base de datos y se implementan todas las colecciones y documentos del modelo de datos diseñado anteriormente de forma manual [10].

5.2.3.3 Obtención de la configuración de Firebase

Antes de poder acceder a la base de datos desde el proyecto de *Flutter*, se debe de obtener la configuración de *Firebase*. Para ello se accede al panel de control de *Firebase*, se selecciona la opción de "Configuración del proyecto" y luego la opción "Añadir aplicación". Se establece la plataforma de destino de nuestra aplicación, en este caso *Flutter* y se sigue las instrucciones para obtener el archivo de configuración, que contiene las credenciales y la información necesaria para conectar con *Firebase* [10] [55].

5.2.3.4 Configuración del proyecto *Flutter*

Una vez que obtenida la configuración de *Firebase*, se configura el proyecto *Flutter* para conectarse con la base de datos. Agregamos la dependencia de *Firebase Firestore* en el archivo *pubspec.yaml* y se ejecuta el comando `flutter pub get` para descargar las dependencias (Ver Anexo M).

5.2.3.5 Inicialización *Firebase* en el proyecto *Flutter*

En el archivo principal de la aplicación Flutter, se inicializa Firebase con la configuración obtenida en el paso anterior. Esto permite establecer la conexión con la base de datos (Ver Anexo N).

5.2.4 Implementación Políticas de acceso

Para la implementación de políticas de acceso el usuario tiene que ser mayor de trece años de edad y obtener el consentimiento de los padres o tutores legales para registrarse en el caso de no contar con la mayoría de edad. Todo acorde a la Ley de Protección de la Privacidad en Línea de los Niños (COPPA) de Estados Unidos y la Ley de Protección de Datos Personales (1581 de 2012) de Colombia [80][81].

5.3 Fase III: Fase de implementación e integración.

5.3.1 Implementación de la interfaz de usuario en *Flutter*

Para la implementación de la interfaz de usuario en *Flutter* es necesario primero establecer el árbol de *Widgets* dentro del aplicativo para así poder establecer las pantallas principales del aplicativo y sus *Widgets* hijos (Ver Anexo O).

5.3.1.1 Implementación de pantalla de registro.

Posterior a la creación del árbol de *Widgets* dentro de *Flutter* se establece la pantalla de *LoginRegisterScreen* dentro del arbol de *Widgets*. Dentro de la clase *LoginRegisterScreen* se Permite a los usuarios ingresar información como nombre, correo electrónico, contraseña y rol (estudiante o profesor) a través de un formulario. El código también incluye validaciones para los campos del formulario y maneja el proceso de registro creando una cuenta de usuario utilizando *Firebase Authentication* y almacenando los datos del usuario en *Firestore*. El código importa las bibliotecas necesarias y define la clase *Register* como un *StatefulWidget*. Dentro de la clase *Register*, se definen variables y controladores para los campos del formulario y se inicializan.

El método *setLoginScreen* se utiliza para navegar hacia atrás a la pantalla de inicio de sesión.

En el método *build*, se muestra un *SingleChildScrollView* que contiene el formulario de registro. El formulario se construye utilizando un *Form widget* y se establece un *GlobalKey* para realizar validaciones. Se agregan varios *TextFormField widgets* para que los usuarios ingresen su nombre, correo electrónico, contraseña, etc. Estos *Widgets* incluyen la funcionalidad de validación y formato visual.

El formulario también incluye un interruptor de selección de rol (*ToggleSwitch*) para que el usuario elija entre estudiante y profesor. Cuando se envía el formulario, se realiza la validación y, si es válido, se ejecuta la lógica de registro. Esto implica crear una cuenta de usuario con *Firebase Authentication* y guardar los datos del usuario en *Firestore*. Durante el proceso de registro, se muestra un indicador de progreso (*showProgress*) para indicar que se está realizando una operación en segundo plano. Finalmente, hay un botón para volver a la pantalla de inicio de sesión utilizando el método *setLoginScreen*. (Ver Anexo Q).

El código del anexo anterior permite la creación y registro en la base de datos de *Firebase* del nuevo usuario, permitiendo así utilizar posteriormente los servicios de inicio de sesión, restablecimiento de contraseña, cambio de contraseña y eliminación de cuenta de usuario definidos en la arquitectura del aplicativo.

Figure 5.2: Implementación de pantalla de registro en *Flutter*. Fuente: Autoría propia.

5.3.1.2 Implementación de pantalla de inicio de sesión

Se define la clase *CustomLogInBody*, que extiende *StatefulWidget*. Esta clase representa el cuerpo de una pantalla de inicio de sesión personalizada del aplicativo. La clase *CustomLogInBody* tiene varios métodos y propiedades que se utilizan para implementar la funcionalidad de la pantalla de inicio de sesión. A continuación se muestra un resumen de las principales características del código:

El estado interno de la clase *CustomLogInBody* se gestiona a través de la clase *_CustomLogInBodyState*, que extiende *State <CustomLogInBody >*. Aquí se definen las variables booleanas *_isObscure3* y visible, y se crean los controladores *emailController* y *passwordController* para los campos de correo electrónico y contraseña respectivamente.

El método *setRegisterScreen* se utiliza para navegar a la pantalla de registro cuando el usuario selecciona la opción correspondiente. El método *setForgotPasswordScreen* se utiliza para navegar a la pantalla de recuperación de contraseña cuando el usuario selecciona la opción correspondiente. El método *build* se encarga de construir la interfaz de usuario de la pantalla de inicio de sesión. Aquí se definen varios *Widgets* como *TextFormField* para capturar el correo electrónico y la contraseña del usuario, y se utilizan varios botones y textos para proporcionar opciones adicionales como registro y recuperación de contraseña.

El método *signIn* se utiliza para manejar la lógica de inicio de sesión. Dependiendo de la plataforma (web o no web), se utiliza la clase *web_auth.FirebaseAuth* o *FirebaseAuth* para autenticar al usuario utilizando el correo electrónico y la contraseña proporcionados. Después de la autenticación exitosa, se redirige al usuario a la pantalla correspondiente según su rol (profesor o estudiante) .

Los métodos *wrongdBMessage*, *wrongEmailMessage* y *wrongPasswordMessage* se utilizan para mostrar mensajes de error en forma de cuadros de diálogo en caso de que ocurran ciertos errores durante el inicio de sesión (Ver Anexo R).

El código del anexo anterior permite establecer el servicio de inicio de sesión, permitiendo así acceder a la pantalla principal del estudiante o profesor, dependiendo del tipo de rol establecido en el servicio de registro de usuario.

Figure 5.3: Implementación de pantalla de inicio de sesión en *Flutter*. Fuente: Autoría propia.

5.3.1.3 Implementación de pantalla de restablecimiento de contraseña

El *Widget ForgotPasswordScreen* contiene una pantalla de restablecimiento de contraseña (ForgotPasswordScreen) y una función (resetPassword) para enviar un correo electrónico de restablecimiento de contraseña.

Se Importa varios paquetes necesarios para el desarrollo de la aplicación, como *flutter/material.dart* para los *Wid-*

gets de *Flutter*, *firebase_auth* para la autenticación de *Firebase*, *firedart* para interactuar con *Firebase Firestore* y *quickalert* para mostrar alertas rápidas en la aplicación.

Se define una función *resetPassword* que toma un parámetro *BuildContext* y una cadena *email*. Esta función se utiliza para restablecer la contraseña del usuario. Si la aplicación se está ejecutando en la web (*kIsWeb* es verdadero), utiliza *web_auth.FirebaseAuth* para enviar un correo electrónico de restablecimiento de contraseña. De lo contrario, utiliza *FirebaseAuth* de la biblioteca *firebase_auth* para realizar la operación.

Luego, se define una clase *ForgotPasswordScreen* que extiende *StatefulWidget*. Esta clase representa la pantalla de restablecimiento de contraseña de la aplicación.

La clase *ForgotPasswordScreen* tiene un estado asociado (*_ForgotPasswordScreenState*) que se encarga de construir la interfaz de usuario de la pantalla.

En el método *build* del estado, se crea la estructura de la pantalla utilizando varios *Widgets* de *Flutter*, como *Scaffold*, *AppBar*, *TextFormField*, *MaterialButton*, etc.

La pantalla muestra un encabezado con el título *OpenEduIA* y un botón de información. Luego, se muestra un campo de entrada de correo electrónico (*TextFormField*) y un botón de "Restablecer Contraseña". Si se ingresa un correo electrónico válido y se presiona el botón, se muestra una alerta de confirmación (*QuickAlert*) y luego se llama a la función *resetPassword* para enviar el correo electrónico de restablecimiento de contraseña.

La pantalla también muestra un enlace "¿Quieres volver?" que redirige al usuario a la pantalla de inicio de sesión cuando se toca (Ver Anexo S).

El código del anexo anterior permite establecer el servicio de restablecimiento de contraseña definido en la arquitectura del aplicativo. Este servicio permite enviar un correo de recuperación de contraseña al correo del usuario, para así establecer una nueva contraseña.

Figure 5.4: Implementación de pantalla de restablecimiento de contraseña en *Flutter*. Fuente: Autoría propia.

5.3.1.4 Implementación de pantalla principal de estudiante

La pantalla principal del estudiante se muestra a través de la clase HomeStudentScreen, que es un StatefulWidget.

- En las primeras líneas, se importan los paquetes necesarios de Flutter.
- La clase HomeStudentScreen es un StatefulWidget que representa la pantalla de inicio para estudiantes. Tiene un estado _HomeStudentScreenState asociado a él.

- La variable *_currentIndex* es un entero que mantiene el índice actual de la pestaña seleccionada en la barra de navegación inferior.
- La lista *_children* contiene una lista de *Widgets* que se muestran en función de la pestaña seleccionada.
- El método *build* crea la interfaz de usuario de la pantalla utilizando *Widgets*de *Flutter*.
- *SystemChrome.setPreferredOrientations* se utiliza para fijar la orientación de la pantalla en modo vertical.
- *WillPopScope* se utiliza para evitar que el usuario vuelva atrás mediante el botón físico o de software del dispositivo.
- El cuerpo principal de la pantalla está envuelto en un *Stack*, que permite apilar varios *Widgets* uno encima del otro.
- La clase *CustomStudentHomeBody* es un *StatelessWidget* que representa el contenido principal de la pestaña de inicio.
- El *Widget WorldsCardSwiper* muestra una lista de tarjetas deslizables que contienen imágenes de los "mundos" disponibles.
- Los métodos *getWorldsImage* y *getWorldsName* se utilizan para obtener las imágenes y nombres de los mundos respectivamente.
- La clase *WorldsCardSwiper* es un *StatefulWidget* que muestra las tarjetas deslizables de los mundos en función de los datos obtenidos.
- En el método *build* de *WorldsCardSwiper*, se utiliza *FutureBuilder* para manejar la carga asíncrona de los datos de los mundos y construir la interfaz de usuario en función del estado de carga.
- Según la plataforma en la que se esté ejecutando la aplicación, se muestra una imagen de fondo diferente para cada tarjeta deslizable.
- Finalmente, el código define la estructura general de la pantalla de inicio del estudiante, que incluye la barra de navegación inferior utilizando el *Widget GNav* de la biblioteca *google_nav_bar* (Ver Anexo T).

El código del anexo anterior permite el acceso a los mundos con las temáticas definidas de inteligencia artificial a tratar. Además, permite la navegación a las diferentes pantallas dentro del rol de estudiante definidas anteriormente.

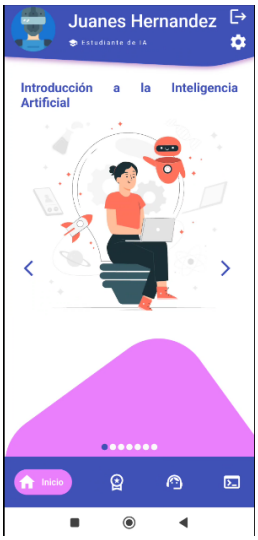

Figure 5.5: Implementación de pantalla principal del estudiante en *Flutter*. Fuente: Autoría propia.

5.3.1.5 Implementación de pantalla de logros del estudiante

El código proporcionado es una clase llamada *CustomStudentArchivementsBody*que extiende de *StatelessWidget*. A continuación, se muestra un resumen del código:

- El constructor CustomStudentArchivementsBody recibe un parámetro opcional Key y lo pasa al constructor de la clase padre StatelessWidget.
- La función *getChallengesInfo* es una función asíncrona que recibe un contexto de tipo *BuildContext* y devuelve un objeto Future. Esta función se encarga de obtener información sobre los desafíos, recompensas, progreso de mundos y progreso de subsecciones del estudiante. La implementación de esta función depende de si la aplicación se está ejecutando en la web o en otro entorno.
- El método *build* es anulado para construir la interfaz de usuario. En este método, se crea un *Scaffold* con un fondo blanco y un *SingleChildScrollView* como su *widget* de cuerpo.
- Dentro del *SingleChildScrollView*, se colocan varios *widgets* que representan diferentes secciones de información sobre los logros del estudiante.
- El *widget FutureBuilder* se utiliza para mostrar diferentes contenidos dependiendo del estado del futuro devuelto por la función *getChallengesInfo*. Si el futuro tiene datos, se muestra la información de los desafíos completados, las recompensas obtenidas, el progreso de mundos y el progreso de *subsecciones*. Si el futuro no tiene datos, se muestra un mensaje de carga.
- En general, la interfaz de usuario muestra información como desafíos completados, recompensas obtenidas, puntos totales, progreso de mundos y progreso de subsecciones. Estos datos se obtienen llamando a la función getChallengesInfo y se muestran en diferentes widgets, como Text, ListView.builder y StepProgressIndicator (Ver Anexo U).

El código del anexo anterior permite mostrar información sobre los desafíos completados, las recompensas obtenidas, los puntos totales, el progreso de los mundos y el progreso de las subsecciones del estudiante. Esto permite la integración de las técnicas de gamificación seleccionadas en la primera fase del proyecto.

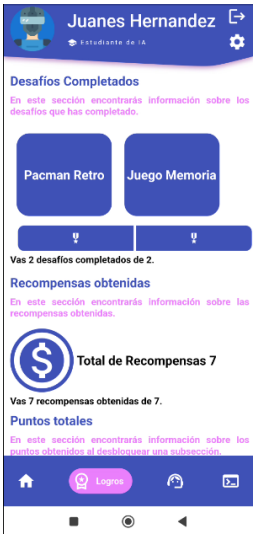

Figure 5.6: Implementación de pantalla de Logros del estudiante en *Flutter*. Fuente: Autoría propia.

5.3.1.6 Implementación de pantalla de compilador de Python

La pantalla principal del estudiante se muestra a través de la clase CustomStudentPythonShell dentro de las opciones de *Gnav* dentro de la pantalla principal *HomeStudentScreen*, que es un *StatefulWidget*.

- Importaciones de paquetes: El código comienza con la importación de varios paquetes necesarios para el funcionamiento del programa, como *flutterfoundation*, *flutter material*, *http*, *dart:convert*, y otros. Estas

importaciones proporcionan las funcionalidades y componentes necesarios para construir la interfaz de usuario y realizar solicitudes *HTTP*.

- Función *executeCommand*: Esta función es una función asíncrona que recibe un código de *Python* como entrada y devuelve una cadena de texto que representa la salida de la ejecución del código. Utiliza la biblioteca http para realizar una solicitud POST a *API Amazon* LAMBDA para conectarse con la función *serverless*. El cuerpo de la solicitud contiene el código de *Python* proporcionado. El resultado de la solicitud se asigna a la variable *result* y se realiza un procesamiento adicional en la cadena de texto antes de devolverla.
- Clase *CustomStudentPythonShell*: Esta clase es un *StatefulWidget*, lo que significa que es capaz de mantener un estado interno que puede cambiar a lo largo del tiempo. La clase define la interfaz de usuario de la aplicación y contiene varios elementos, como un campo de texto (*TextFormField*), un botón (*MaterialButton*) y un contenedor para mostrar la salida del código de *Python*. Además, hay métodos para inicializar y liberar recursos, y un método build para construir la estructura de la interfaz de usuario.
- _*CustomStudentPythonShellState*: Esta clase es la implementación del estado para la clase *CustomStudentPythonShell*. Define el estado interno de la clase, como los controladores de texto y de enfoque, y proporciona métodos para manipular el estado (Ver Anexo V).
- Realización del código en *Python* dentro de la función *serverless* que recibe el código a ejecutar del usuario(Ver Anexo W), por protección de datos personales del aplicativo no se agrega la dirección de la función *LAMBDA* dentro del anexo.

El código de los dos anteriores anexos permite la conexión y utilización de un servicio de función *serverless* en *AWS* con el objetivo de implementar un compilador del lenguaje *Python* dentro del aplicativo. Esto permite a los estudiantes probar los conocimientos adquiridos en *Python* dentro del mismo aplicativo.

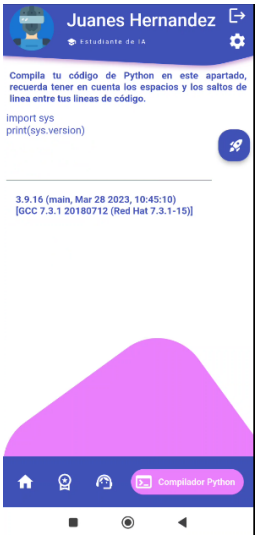

Figure 5.7: Implementación de pantalla de compilador de Python en *Flutter*. Fuente: Autoría Propia.

5.3.1.7 Implementación de pantalla de Asistente Virtual

Para la implementación de la pantalla de asistente virtual se define una clase llamada *CustomStudentAssistantBody* que extiende *StatefulWidget*, lo que indica que es un *Widget* con estado que puede cambiar a lo largo del tiempo. Aquí se detalla el funcionamiento del código:

- Se importan los paquetes necesarios, como*flutter-material.dart* y *chatgpt_completions*, así como otros archivos y clases locales necesarios.
- Se define la clase *CustomStudentAssistantBody*, que es el *Widget* principal de la pantalla del asistente virtual para estudiantes. Esta clase también define el estado interno del *Widget*.
- Se crea un controlador de texto llamado *messageController* para manejar la entrada de texto del usuario.

- Se implementa el método *build* para construir la interfaz de usuario del *Widget*. En esta función, se define la estructura visual de la pantalla utilizando otros *Widgets* de *Flutter*.
- Se utiliza el *Widget Scaffold* como el contenedor principal de la pantalla. *Scaffold* proporciona una estructura básica para la interfaz de usuario y se utiliza comúnmente en las aplicaciones de *Flutter*.
- El contenido principal de la pantalla se coloca dentro de un *SingleChildScrollView*, que permite desplazarse verticalmente si el contenido supera el tamaño de la pantalla.
- Se construye la interfaz de usuario utilizando una combinación de *Widgets* como *Column*, *Row*, *Container*, *TextFormField*, *Text*, *Icon*, etc. Estos *Widgets* se utilizan para crear la disposición visual de los elementos en la pantalla, como la barra de búsqueda, el botón de envío y la sección de respuestas.
- El método *textAnswer* se define para enviar una solicitud a una *API* de generación de texto (*chatgpt_completions*) y obtener una respuesta basada en la entrada del usuario. La respuesta se muestra en la sección de respuestas de la pantalla.
- Se utiliza un *FutureBuilder* para manejar la llamada asincrónica al método *textAnswer* y mostrar el resultado en la interfaz de usuario. Dependiendo del estado del *Future*, se *renderiza* un indicador de carga, la respuesta obtenida o un mensaje predeterminado (Ver Anexo X).

El código del anexo anterior permite la implementación de un asistente virtual que permite al estudiante hacer preguntas mediante el uso de un campo de texto a un modelo de *chat* de la empresa *OpenAI* llamado ChatGPT. Utiliza una *API RESTful* para obtener la respuesta a la pregunta realizada por el estudiante.

Figure 5.8: Implementación de pantalla de Asistente Virtual en *Flutter*. Fuente: Autoría propia.

5.3.1.8 Implementación desafío *Retro Pacman* dentro del primer mundo

Para la implementación de la pantalla del desafío se define una clase *PacManWorld1HomeScreen*. Aquí se detalla el funcionamiento del código:

- Se importan los paquetes necesarios, como *flutter-material.dart*, *dart:math* y *bonfire*, así como otros archivos y clases locales necesarios.
- Se define la clase *PacManWorld1HomeScreen*, que es un *StatelessWidget* y representa la pantalla de inicio del juego de *Retro Pacman*.
- Se define la constante heightMap, que representa la altura del mapa del juego, y la constante tileSize, que representa el tamaño de cada celda en el mapa.
- El método build se implementa para construir la interfaz de usuario del widget. En esta función, se define la estructura visual de la pantalla utilizando otros widgets de Flutter.

- Se utiliza el *Widget Container* como el contenedor principal de la pantalla y se establece su color de fondo en negro.
- El contenido principal se coloca dentro de un *Widget Center* para centrar el contenido en la pantalla.
- Se define un tamaño (size) basado en el ancho y alto de la pantalla del dispositivo.
- Se calcula un factor de zoom (zoom) en función del tamaño de la pantalla y la altura del mapa del juego. Esto se utiliza para ajustar la escala del juego al tamaño de la pantalla del dispositivo.
- Se utiliza un *Widget BonfireWidget* como el *Widget* principal del juego. Se proporciona un objeto *WorldMapByTiled* como el mapa del juego, que se carga desde un archivo de mapa llamado " *map.tmj*". Además, se configuran los diferentes objetos del juego, como sensores, puntos, fantasmas, etc., utilizando un diccionario de objetos y propiedades.
- Se agrega un joystick al juego utilizando el *Widget Joystick*. En este caso, se utiliza un joystick direccional con configuraciones específicas.
- Se define un mapa de superposiciones (overlayBuilderMap) que mapea un nombre de superposición a un constructor de *Widgets*. En este caso, se tiene una superposición llamada "*score*" que muestra una interfaz de juego.
- Se configura la cámara del juego utilizando el objeto *CameraConfig*, estableciendo el factor de zoom calculado anteriormente.
- Se crea un objeto *PacMan* como el jugador principal del juego, estableciendo su posición inicial.
- Finalmente, se retorna la estructura de *Widgets* anidados para construir la interfaz de usuario del juego (Ver Anexo Y).

El código del anexo anterior implementa un desafío basado en el videojuego *PacMan* para integrar el proceso de desafíos en la estrategia de gamificación establecida. Esto permite que los estudiantes se involucren con el mundo de *Introducción a la Inteligencia Artificial* dentro del aplicativo.

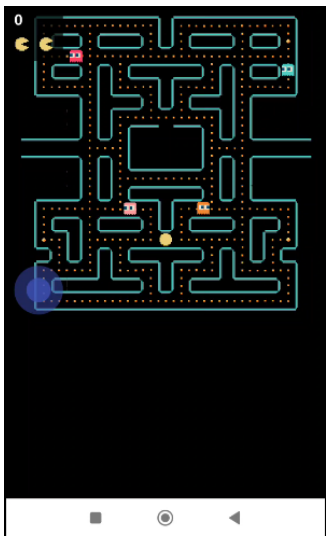

Figure 5.9: Implementación de pantalla de Desafío de *RetroPacman* en *Flutter*. Fuente: Autoría propia.

5.3.1.9 Implementación desafío *Juego de Memoria* dentro del segundo mundo

Para la implementación de la pantalla del desafío de Juego de Memoria se define una clase *Game*. Aquí se detalla el funcionamiento del código:

- Importar los paquetes necesarios para la aplicación, como *dart:async*, *dart:io*, *package:flutter-material.dart*, etc.

- Verifica si la aplicación se está ejecutando en la web utilizando la variable *kIsWeb*.
- Si se está ejecutando en la web, utiliza las bibliotecas *web_firestore* y *web_auth* para acceder a *Firestore* y la autenticación respectivamente.
- Si no se está ejecutando en la web, utiliza las bibliotecas c*loud_firestore y firebase_auth* para acceder a *Firestore* y la autenticación respectivamente.
- Define una clase *Game* que extiende de *StatefulWidget*.
- Implementa la función *createState* para crear el estado mutable de *Game* mediante la clase *_GameState*.
- Declara y inicializa varias variables de estado, como *levelForCardCount*, *tries*, *score*, *axisNumber*, *timer*, *startTime*, *level* y *complete*.
- Implementa los métodos *startTimer*, *initState*, *didChangeDependencies* y *build*.
- En el método *startTimer*, se inicializa un temporizador que decrementa el valor de *startTime* cada segundo. Cuando *startTime* alcanza cero, se muestra un diálogo de finalización del juego.
- En el método *initState*, se inicializa el juego y se llama a la función *initGame* de la clase *GameLogic*.
- En el método *didChangeDependencies*, se inicializa el juego y se llama a la función *initGame* de la clase *GameLogic*.
- En el método *build*, se construye la interfaz de usuario del juego utilizando *Widgets* de *Flutter*, como *Scaffold*, *AppBar*, *SingleChildScrollView*, *Column*, *Row*, *GridView*, etc. Los *Widgets* muestran información sobre el tiempo restante, puntaje y movimientos realizados. También muestra una cuadrícula de tarjetas que representan el juego en sí (Ver Anexo Z).

El código del anexo anterior implementa un desafío basado en el juego de memoria para integrar el proceso de desafíos en la estrategia de gamificación establecida. Esto permite que los estudiantes se involucren con el mundo de *Lógica proposicional y de primer orden* dentro del aplicativo.

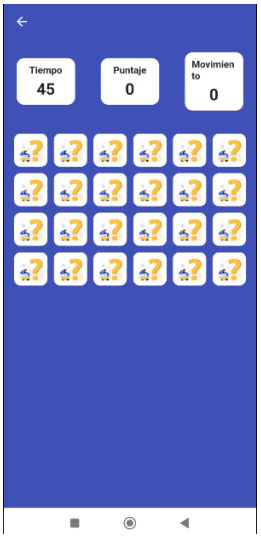

Figure 5.10: Implementación de pantalla de desafío *Juego de Memoria* en *Flutter*. Fuente: Autoría propia.

5.3.1.10 Implementación pantalla de quices y vídeos dentro de cada sub-sección de los mundos

Se define *CustomWorldSubsectionHomeScreen* que extiende de *StatefulWidget*. Esta clase representa la pantalla principal de una sección de un mundo personalizado en una aplicación *Flutter*.

- La clase *CustomWorldSubsectionHomeScreen* tiene varios campos requeridos, como *subsectionName*, *subsectionProgress*, *worldProgress*, *actualWorld* y *actualSubsection*, que se utilizan para mostrar información específica en la pantalla.

- La clase *CustomWorldSubsectionHomeScreen*implementa el método *build* para construir la interfaz de usuario de la pantalla. En esta función, se utiliza un *Scaffold* como el contenedor principal de la pantalla.
- La parte superior de la pantalla contiene una barra de aplicaciones personalizada (*AppBar*) con un título centrado y un fondo de color *indigo*. El título se obtiene del campo *subsectionName*.
- El cuerpo de la pantalla es un *Stack* que contiene un fondo personalizado (*CustomStudentBackground*) y dos *Widgets* condicionales (*CustomWorldSubsectionWidgetVideos* y *CustomWorldSubsectionWidgetQuiz*) dependiendo del valor de *_currentIndex*. El valor de *_currentIndex* se actualiza cuando se cambia la pestaña en la barra de navegación inferior.
- La barra de navegación inferior (*bottomNavigationBar*) utiliza el widget GNav de la biblioteca *google_nav_bar* para mostrar dos pestañas: Videos" y "Quices". La pestaña actual se resalta en color púrpura. El valor de *_currentIndex* se actualiza cuando se cambia la pestaña.
- El código también incluye una función asincrónica *getUserProgress* que se utiliza para obtener el progreso del usuario desde una base de datos. La implementación varía según si la aplicación se ejecuta en la web o en dispositivos móviles. El progreso del usuario se almacena en las variables *world_progress* y *subsections_progress* y se devuelve como una lista.
- La función *getCorrectAnswer* se utiliza para obtener el índice de la respuesta correcta en una lista de opciones de preguntas. Toma la lista de opciones y la respuesta correcta como argumentos y devuelve el índice + 1 de la respuesta correcta. Si no se encuentra la respuesta correcta, devuelve -1.
- La función asincrónica *completedSection* se utiliza para actualizar el progreso del usuario cuando se completa una sección. Al igual que *getUserProgress*, su implementación varía según si la aplicación se ejecuta en la web o en dispositivos móviles. Actualiza el progreso del usuario en la base de datos y navega a la pantalla *HomeStudentScreen*.
- La clase *CustomWorldSubsecionShowQuiz* representa la pantalla de visualización de un cuestionario. Tiene campos para el mundo y la subsección actual, así como para la pregunta del cuestionario, la respuesta correcta y las opciones de respuesta.
- La pantalla de visualización del cuestionario muestra la pregunta y las opciones de respuesta. Cuando se elige una opción, se verifica si es la respuesta correcta y se realiza la acción correspondiente (Ver Anexo AA).

El código del anexo anterior permite al estudiante acceder a los videos y cuestionarios dentro de cada subsección de los mundos, lo que permite evaluar y tener acceso a los videos explicativos de cada uno de los temas de inteligencia artificial establecidos.

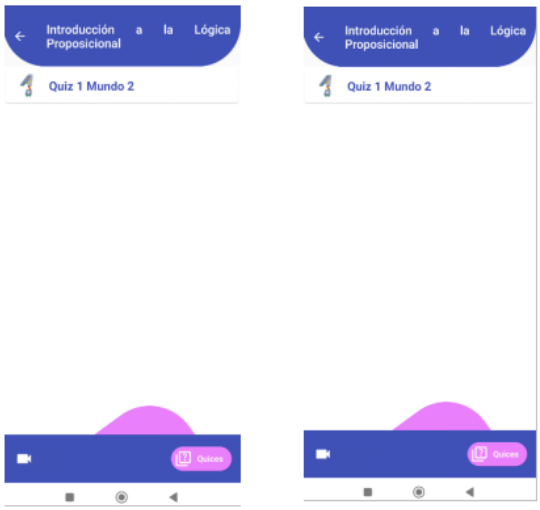

Figure 5.11: Implementación de pantallas de quices y videos en *Flutter*. Fuente: Autoría propia.

5.3.1.11 Implementación pantalla principal del profesor

La implementación de la pantalla principal del profesor posee tres clases principales *HomeProfessorScreen*, *CustomProfessorActiveGroupsScreen* y *CustomActiveGroupWidget*.

- *HomeProfessorScreen*: Esta clase representa la pantalla principal para el profesor. Es un *Widget* de estado mutable (*StatefulWidget*) que muestra un fondo personalizado (*ProfessorBackground*) y la pantalla de grupos activos personalizada (*CustomProfessorActiveGroupsScreen*) apilada encima de ella. Además, establece las orientaciones de pantalla preferidas utilizando *SystemChrome.setPreferredOrientations*.
- *CustomProfessorActiveGroupsScreen*: Esta clase es un *Widget* sin estado (*StatelessWidget*) que muestra la pantalla de grupos activos para el profesor. Contiene una barra de aplicación personalizada (*CustomStudentAppBarHome*) y un *Widget* de lista de grupos activos personalizado (*CustomActiveGroupWidget*).
- *CustomActiveGroupWidget*: Esta clase es un *Widget* de estado mutable (StatefulWidget) que muestra una lista de grupos activos para el profesor. Utiliza el método *getGroupsInfo* para obtener información sobre los grupos del profesor desde una base de datos (*Firebase Firestore*) y luego muestra los grupos en una lista deslizable (*ListView*). Cada elemento de la lista es representado por un *Card* con detalles del grupo (Ver Anexo AB).

En el código del anexo anterior, se permite a los usuarios registrados con el rol de profesor ingresar a la pantalla inicial del profesor. En esta pantalla, los profesores encontrarán los grupos de estudio creados y los estudiantes agregados a dichos grupos de estudio.

Figure 5.12: Implementación de pantalla principal del profesor en *Flutter*. Fuente: Autoría propia.

5.4 Fase IV: Verificación

En esta sección se describe el proceso de verificación del aplicativo multiplataforma realizado en *Flutter* utilizando la metodología *SCRUM* y casos de prueba. La verificación es una etapa crucial del desarrollo de software que garantiza que el aplicativo cumpla con los requisitos y funcionalidades establecidos. La metodología *SCRUM* proporciona un marco de trabajo ágil y colaborativo, mientras que los casos de prueba son herramientas efectivas para validar y evaluar el funcionamiento del aplicativo [3]. En el contexto de la verificación del aplicativo *Flutter*, se pueden seguir los siguientes pasos:

5.4.1 Definición del Producto

En esta etapa, se establecen los objetivos y requisitos del aplicativo *Flutter* en contexto a los objetivos y entregados planteados en el anteproyecto. Esto incluye definir las funcionalidades principales, las interfaces de usuario, las integraciones con otros sistemas y cualquier otro aspecto relevante [3] [7].

5.4.2 Planificación de *Sprint*

Se planificó un sprint de desarrollo y verificación, que es un período de tiempo fijo durante el cual se lleva a cabo la implementación y la verificación de una parte específica del aplicativo. Se definieron las tareas, se asignaron responsabilidades y se estableció un cronograma [3].

5.4.3 Desarrollo del Aplicativo

Se trabajó en la implementación del aplicativo *Flutter* de acuerdo con los requisitos y funcionalidades definidos. Durante este proceso, se siguen las mejores prácticas de desarrollo y se realizaron pruebas unitarias para garantizar la calidad del código [3] [5] [7].

5.4.4 Creación de Casos de Prueba

Se elaboran casos de prueba que cubran los diferentes escenarios y funcionalidades del aplicativo. Estos casos de prueba deben ser exhaustivos y cubrir la mayoría de situaciones que puedan surgir durante la interacción del usuario con el aplicativo.

5.4.5 Ejecución de Pruebas

Se ejecutan los casos de prueba en el aplicativo *Flutter* para comprobar su funcionamiento y detectar posibles errores o fallos. Es importante registrar los resultados de las pruebas, incluyendo cualquier defecto encontrado y su severidad [3] [5] [7].

5.4.6 Despliegue de los cambios en el repositorio *Open Source*

Al finalizar el sprint, se realiza un despliegue de los cambios en el código dentro del repositorio *Open Source* en la plataforma de control de versiones *GitHub* [3] [82] [83] (Ver en `https://github.com/Juanes-27/openeduia_public.git/`).

5.4.7 Despliegue del aplicativo en diferentes plataformas

Para el despliegue del aplicativo se utiliza comandos especializados de *Flutter* dentro del sistema de desarrollo *Visual Studio Code* para generar el ejecutable de cada una de dichas plataformas, mediante el uso de las extensiones *Dart*, *Flutter* y *Flutter Tree*.

Plataforma	Comando
iOS	`flutter build ios`
Android	`flutter build apk`
Linux	`flutter build linux`
Windows	`flutter build windows`

Tabla 5.17: Comandos de *Flutter* para generar ejecutables. Fuente: Autoría propia.

5.4.8 Problema de Concurrencia

Para manejar el problema de concurrencia dentro del aplicativo en *Flutter* se establecen los siguientes puntos a seguir:

- Sincronización de acceso a recursos compartidos: Se asegura de que los hilos o procesos que acceden a la base de datos o recursos compartidos lo hagan de manera sincronizada. Esto se logra mediante del uso de los *kits* de *APIs* brindado por *Google* que permite establecer los limites de petición, colas de petición y procesos en diferentes hilos para así poder balancear la carga de las peticiones al servidor [84].

- Transacciones y bloqueo de registros: Al utilizar *Firebase* como base de datos, aprovecha las transacciones para garantizar operaciones atómicas en la base de datos. Esto ayuda a evitar condiciones de carrera al realizar múltiples operaciones que afectan los mismos registros [85].

- Escalabilidad independiente: En una arquitectura de microservicios, cada microservicio se ejecuta de manera independiente y puede escalar horizontalmente de manera individual según sea necesario. Esto significa que se puede ajustar la capacidad de cada servicio específico para manejar la carga de concurrencia de manera más efectiva, en lugar de escalar todo el sistema de forma uniforme. Esto permite distribuir la carga y evitar cuellos de botella en puntos críticos [86].

Capítulo 6

Resultados y Discusión

6.1 Casos de prueba

Los casos de prueba son una técnica utilizada en el ámbito de la ingeniería de software para verificar el funcionamiento de un sistema. Consiste en una serie de pasos detallados que describen cómo probar una función o característica específica del sistema, con el objetivo de comprobar que cumple con los requisitos establecidos y que se comporta correctamente en diferentes situaciones.

Para este proyecto se diseñaron los casos de prueba para probar funcionalidades y escenarios de prueba de los sistemas que se desarrollaron los cuales interactúan directamente con el usuario. Estos sistemas fueron agrupados por su propósito y a continuación se muestran los casos de prueba.

6.1.1 Registro y Configuración

ID	Resumen	Prioridad	Precondiciones	Datos de Entrada	Pasos	Resultado Esperado	Resultado Obtenido	Plataforma	Fecha de Ejecución
1	Registrar al usuarios con correo electrónico y contraseña según su tipo de rol.	Alta	No estar registrado dentro del aplicativo.	Nombres, Apellidos, Correo electrónico, Contraseña, Confirmación de contraseña y Tipo de rol.	1. Abrir el aplicativo. 2. Ir a la pestaña de "¿No estás registrado? Regístrate ahora". 3. Ingresar Nombres, Apellidos, Correo electrónico, Contraseña, Confirmación de contraseña. 4. Escoger entre el rol de estudiante o profesor. 5. Presionar el botón "Regístrate".	1. Creación del usuario dentro de la base de datos. 2. Aparición de barra de notificación inferior con el texto "Cuenta creada" y se mantiene en la misma pantalla.	1. Creación del usuario dentro de la base de datos. 2. Aparición de barra de notificación inferior con el texto "Cuenta creada" y se mantiene en la misma pantalla.	Linux, Windows, Android y Web	13/02/2023
2	Ingresar a la pantalla principal dependiendo del tipo de rol.	Alta	Estar registrado.	Correo electrónico y Contraseña.	1. Abrir el aplicativo. 2. Ingresar el correo electrónico y contraseña registrada. 3. Presionar el botón de "Ingreso".	1. Se crea una instancia de autenticación con la base de datos. 2. Se muestra la pantalla de bienvenida según el rol del usuario.	1. Se crea una instancia de autenticación con la base de datos. 2. Se muestra la pantalla de bienvenida según el rol del usuario.	Linux, Windows, Android y Web	13/02/2023
3	Restablecimiento de contraseña mediante el uso del correo electrónico independiente del tipo de rol.	Alta	Estar registrado.	Correo electrónico.	1. Abrir el aplicativo. 2. Ir a la pestaña de "¿Olvidaste tu contraseña? Toca aquí". 3. Ingresar en el campo de texto del correo electrónico registrado de la cuenta la cual quiere restablecer la contraseña. 4. Presionar el botón "Estoy seguro" dentro del dialogo de advertencia.	1. Se envía un link al correo electrónico para el restablecimiento de contraseña mediante un dialogo de cambio de contraseña dentro de un servicio web dentro de Firebase y FireAuth. 2. Posteriormente una aparición de barra de notificación inferior con el texto "Revisa tu correo para restablecer tu contraseña". 3. Direccionamiento dentro de la pantalla inicial del inicio de sesión del aplicativo.	1. Se envía un link al correo electrónico para el restablecimiento de contraseña mediante un dialogo de cambio de contraseña dentro de un servicio web dentro de Firebase y FireAuth. 2. Posteriormente una aparición de barra de notificación inferior con el texto "Revisa tu correo para restablecer tu contraseña". 3. Direccionamiento dentro de la pantalla inicial del inicio de sesión del aplicativo.	Linux, Windows, Android y Web	13/02/2023
4	Mostrar la información sobre las librerías utilizadas dentro del aplicativo.	Baja	N/A	N/A	1. Abrir el aplicativo. 2. Ir al icono de información en la parte superior derecha. 3. Presionar el botón "VIEW LICENSES".	1. Mostrar la pantalla de información con el nombre del aplicativo, versión actual del mismo y lista con todas las librerías utilizadas y sus correspondientes licencias.	1. Mostrar la pantalla de información con el nombre del aplicativo, versión actual del mismo y lista con todas las librerías utilizadas y sus correspondientes licencias.	Linux, Windows, Android y Web	14/02/2023
5	Cambio de avatar del usuario independiente del rol.	Media	Estar registrado y logueado.	Correo electrónico y Contraseña	1. Abrir el aplicativo. 2. Iniciar sesión con el correo electrónico y contraseña. 3. Presionar el botón "Continua". 4. Ubicarse dentro de la barra superior del aplicativo y presionar dentro del avatar actual del usuario. 5. Seleccionar un nuevo avatar del usuario. 6. Presionar el botón "Aceptar" dentro del dialogo emergente.	1. Actualización del avatar del usuario dentro de la base de datos y direccionamiento de la pantalla actual con el nuevo avatar seleccionado.	1. Actualización del avatar del usuario dentro de la base de datos y redireccionamiento de la pantalla actual con el nuevo avatar seleccionado.	Linux, Windows, Android y Web	14/02/2023
6	Terminar instancia de inicio de sección del usuario independiente del rol.	Alta	Estar registrado y con sesión iniciada.	Correo electrónico y Contraseña	1. Abrir el aplicativo. 2. Iniciar sesión con correo electrónico y contraseña. 3. Presionar el botón "Continuar" para pasar la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra superior del aplicativo. 5. Presionar el botón "Material Logout". 6. Presionar sobre el botón "SI" dentro del dialogo de alerta de confirmación de salida.	1. Finalización de instancia de inicio de sesión. 2. Navegación dentro de la pantalla inicial de inicio de sesión del aplicativo.	1. Finalización de instancia de inicio de sesión. 2. Navegación dentro de la pantalla inicial de inicio de sesión del aplicativo.	Linux, Windows, Android y Web	14/02/2023
7	Cambio de contraseña de usuario independiente del rol.	Alta	Estar registrado y logueado.	Correo electrónico y Contraseña	1. Abrir el aplicativo. 2. Iniciar sesión con correo electrónico y contraseña. 3. Presionar el botón "Continuar" para pasar la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra superior del aplicativo. 5. Presionar el botón "Material Settings". 6. Navegar en la pantalla de "Configuración". 7. Presionar sobre el botón "Cambia tu contraseña". 8. Ingresar los datos sobre el campo del texto "Nueva Contraseña" y "Confirma tu contraseña". 9. Presionar el botón "Estoy seguro".	1. Finalización de instancia de inicio de sesión. 2. Cambio de contraseña dentro de la base datos. 3. Aparición en la parte inferior de una notificación con la información "Contraseña cambiada". 4. Redireccionamiento dentro de la pantalla de inicio de sesión del aplicativo.	1. Finalización de instancia de inicio de sesión. 2. Cambio de contraseña dentro de la base datos. 3. Aparición en la parte inferior de una notificación con la información "Contraseña cambiada". 4. Redireccionamiento dentro de la pantalla de inicio de sesión del aplicativo.	Linux, Windows, Android y Web	15/02/2023
8	Eliminación de cuente de usuario independiente del rol.	Alta	Estar registrado y con inicio de sesión.	Correo electrónico y Contraseña	1. Abrir el aplicativo. 2. Iniciar sesión con correo electrónico y contraseña. 3. Presionar el botón "Continuar" para pasar la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra superior del aplicativo. 5. Presionar el botón "Material Settings". 6. Navegar en la pantalla de "Configuración". 7. Presionar sobre el botón "Elimina mi cuenta". 8. Ingresar los datos sobre el campo del texto "Ingresa la Frase". 9. Presionar nuevamente el botón "Elimina mi cuenta". 10. Presionar el botón "Estoy seguro".	1. Finalización de instancia de inicio de sesión. 2. Eliminación del usuario dentro de la base de datos. 3. Aparición de notificación inferior con la información "Cuenta Eliminada". 4. Direccionamiento dentro de la pantalla inicial de inicio de sesión del aplicativo.	1. Finalización de instancia de inicio de sesión. 2. Eliminación del usuario dentro de la base de datos. 3. Aparición de notificación inferior con la información "Cuenta Eliminada". 4. Direccionamiento dentro de la pantalla inicial de inicio de sesión del aplicativo.	Linux, Windows, Android y Web	15/02/2023

Tabla 6.1: Tabla casos de uso de registro y configuración. Fuente: Autoría propia.

Se realiza el video correspondiente para demostrar en material visual los casos de prueba de registro y configuración (ver en `https://youtu.be/vGUbD7zHY2I`) denominado "Casos de Prueba Registro y Configuración".

6.1.2 Pantallas de rol Estudiante

ID	Resumen	Prioridad	Precondiciones	Datos de Entrada	Pasos	Resultado Esperados	Resultado Obtenido	Plataforma	Fecha de Ejecución
9	Tarjetas deslizables con la información de mundos disponible.	Alta	Estar registrado y logueado con una cuenta en el rol de Estudiante.	N/A	1. Abrir el aplicativo. 2. Iniciar sesión con el correo electrónico y contraseña dentro de la cuenta con rol de 'Estudiante'. 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Inicio". 5. Ubicarse en el centro de la pantalla deslizar utilizando las flechas o el gesto táctil dependiendo del dispositivo. 6. Navegar entre los diferentes mundos dentro del aplicativo 7. Para seleccionar el mundo deseado presionar sobre la imagen respectiva de cada tarjeta deslizable.	1. Desplegar una nueva pantalla que en la barra superior aparecerá con cuatro secciones tituladas "Descripción, Contenido, Desafíos y Comentarios". 2. Los títulos de "Desafíos y Comentarios" estarían dentro de una barra de navegación inferior. 3. Presionar sobre cada icono la pantalla cambia a cada sesión correspondiente si el mundo está habilitado. 4. Para habilitar cada mundo es necesario completar los mundos anteriores. 5. Aparecerá un dialogo de información con el texto "Tienes que completar el mundo anterior" y se deberá presionar el botón "Aceptar" para volver al menú principal de la cuenta con el rol 'Estudiante'.	1. Desplegar una nueva pantalla que en la barra superior aparecerá con cuatro secciones titulaladas "Descripción, Contenido, Desafíos y Comentarios". 2. Los títulos de "Desafíos y Comentarios" estarían dentro de una barra de navegación inferior. 3. Presionar sobre cada icono la pantalla cambia a cada sesión correspondiente si el mundo está habilitado. 4. Para habilitar cada mundo es necesario completar los mundos anteriores. 5. Aparecerá un dialogo de información con el texto "Tienes que completar el mundo anterior" y se deberá presionar el botón "Aceptar" para volver al menú principal de la cuenta con el rol 'Estudiante'.	Linux, Windows, Android y Web	20/02/2023
10	Pantalla con todos los logros obtenidos por el usuario con rol "Estudiante" dentro del aplicativo.	Alta	Estar registrado y logueado con una cuenta con el rol de Estudiante.	N/A	1. Abrir el aplicativo. 2. Iniciar sesión con el correo electrónico y contraseña dentro de la cuenta con rol de 'Estudiante'. 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Logros" y posteriormente ubicarse en el centro de la pantalla. 5. Deslizar hacia la parte de abajo para ver la informción de challenges completados, recompensas obtenidas, mundo actual y subsección actual.	1. Pantalla con cuatro sesiones "Challenges Completados, Recompensas Obtenidas, Mundo Actual, Subsección Actual". 2. Aparecerá la información y gráficas de la cantidad de challenges pasados, recompensas obtenidas y progreso del usuario con rol de 'Estudiante'.	1. Pantalla con cuatro sesiones "Challenges Completados, Recompensas Obtenidas, Mundo Actual, Subsección Actual". 2. Aparecerá la información y gráficas de la cantidad de challenges pasados, recompensas obtenidas y progreso del usuario con rol de 'Estudiante'.	Linux, Windows, Android y Web	20/02/2023
11	Pantalla de asistencia virtual para el usuario con rol "Estudiante" dentro del aplicativo.	Alta	Estar registrado y logueado con una cuenta con rol de Estudiante y acceso a internet.	Texto con la pregunta que se desea enviar al asistente virtual.	1. Abrir el aplicativo. 2. iniciar sesión con el correo electrónico y contraseña dentro de la cuenta con el rol de "Estudiante". 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Asistencia virtual". 5. Ingresar la pregunta en el campo de texto "Pregúntame lo que quieras...". 6. Presionar el botón con el icono "Material Send".	1. Actualización de la pantalla actual con el texto respuesta después de haber enviado la petición de respuesta a la pregunta mediante el uso de un Gateway y una REST API al servicio de ChatGPT por medio del uso de una API key única.	1. Actualización de la pantalla actual con el texto respuesta después de haber enviado la petición de respuesta a la pregunta mediante el uso de un Gateway y una REST API al servicio de ChatGPT por medio del uso de una API key única.	Linux, Windows, Android y Web	21/02/2023
12	Pantalla de Compilador de Python para el usuario con rol "Estudiante" dentro del aplicativo,	Alta	Estar registrado y logueado con una cuenta con rol de Estudiante y acceso a internet.	Código en Python en formato de texto para compilar dentro de la sección.	1. Abrir el aplicativo. 2. Iniciar sesión con el correo electrónico y contraseña dentro de la cuenta con rol de 'Estudiante'. 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Compilador de Python". 5.Ingresar el código en el campo de texto "Ejemplo". 6. Presionar el botón con el icono 'Material Rocket Launch'.	1. Actualizacón de la pantalla actual con el texto respuesta despues de enviar la petición a un servicio Serverlas dentro de AWS Lambda el cual compila el texto enviado. 2. Mediante el uso de un Gateway y una REST API al servicio se obtiene el resultado de la compilación.	1. Actualización de la pantalla actual con el texto respuesta después de enviar la petición a un servicio Serverlas dentro de AWS Lambda el cual compila el texto enviado. 2. Mediante el uso de un Gateway y una REST API al servicio se obtiene el resultado de la compilación.	Linux, Windows, Android y Web	21/02/2023
13	Pantalla de inicio de cuenta con rol "Estudiante".	Alta	Estar registrado y logueado con una cuenta con rol de Estudiante y acceso a internet.	N/A	1. Abrir el aplicativo. 2. Iniciar sesión con el correo electrónico y contraseña dentro de la cuenta con rol de 'Estudiante'. 3. Presionas el botón "Continuar" para pasar a la pantalla de bienvenido del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Inicio".	1. Se presenta la pantalla de inicio del usuario con rol de Estudiante, mostrando el primer mundo en el centro de la pantalla, las flechas para deslizarse a través de los mundos del aplicativo. 2. En la parte superior aparece el titulo con el nombre del usuario y debajo de este el rol al que pertenece "Estudiante de IA", con su respectivo avatar en la parte izquierda y por último el icono de salir se sesión y el de ajustes en la parte izquierda superior.	1. Se presenta la pantalla de inicio del usuario con rol de Estudiante, mostrando el primer mundo en el centro de la pantalla, las flechas para deslizarse a través de los mundos del aplicativo. 2. En la parte superior aparece el título con el nombre del usuario y debajo de este el rol al que pertenece "Estudiante de IA", con su respectivo avatar en la parte izquierda y por último el icono de salir se sesión y el de ajustes en la parte izquierda superior.	Linux, Windows, Android y Web	21/02/2023

Tabla 6.2: Tabla casos de uso de pantalla con rol de Estudiante. Fuente: Autoría propia.

Se realiza el video correspondiente para demostrar en material visual los casos de prueba de pantalla de rol estudiante (ver en `https://youtu.be/UHKXNqkwbWE`) denominado "Casos de Prueba Estudiante".

6.1.3 Pantallas de rol Profesor

ID	Resumen	Prioridad	Precondiciones	Datos de Entrada	Pasos	Resultado Esperado	Resultado Obtenido	Plataforma	Fecha de Ejecución
14	Agregar cuenta tipo "Estudiante" dentro de un grupo de estudio creado por un usuario tipo "Profesor".	Alta	Estar registrado y logueado con una cuenta con rol de "Profesor".	Código de identificación del grupo de estudio creado por una cuenta tipo "Profesor".	1. Abrir el aplicativo. 2. Iniciar sesión con correo electrónico y contraseña dentro de una cuenta con rol de "Profesor". 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra superior del aplicativo y tocar dentro del icono "Material Settings". 5. Navegar sobre la pantalla de "Configuración". 6. Presionar sobre la opción "Agregar tu Cuenta a un grupo de Estudio". 7. Ingresar los datos sobre el campo de texto "Código de Grupo de Estudio". 8. Presionar el botón "Agregar mi Cuenta". 9. Presionar el botón "Estoy seguro" dentro del dialogo de advertencia.	1. Agregar al usuario dentro del grupo de estudio dentro del grupo de estudio dentro de la base de datos. 2. Aparición de notificación inferior con la información "Cuenta Agregada a Cuenta de usuario".	1. Agregar al usuario dentro del grupo de estudio dentro del grupo de estudio dentro de la base de datos. 2. Aparición de notificación inferior con la información "Cuenta Agregada a Cuenta de usuario".	Linux, windows, Android y Web.	27/02/2023
15	Pantalla de inicio de cuenta con rol de "Profesor".	Alta	Estar registrado y logueado con una cuenta con rol de "Profesor".	Nombre del nuevo grupo de estudio.	1. Abrir el aplicativo. 2. Iniciar sesión con correo electrónico y contraseña dentro de una cuenta con rol de "Profesor". 3. Presionar el botón "Continuar" para pasar a la pantalla de bienvenida del usuario. 4. Ubicarse dentro de la barra de navegación inferior en la sección titulada "Crear Grupo". 5. Ingresar los datos de nombre de nuevo grupo de estudio y tocar el botón "Aceptar". 6. Presionar el botón "Estoy seguro" del dialogo de advertencia emergente en la pantalla.	1. Creación de grupo de estudio dentro de la base de datos del aplicativo. 3. Aparición de notificación de barra inferior con el mensaje 'Grupo de estudio creado'.	1. Creación de grupo de estudio dentro de la base de datos del aplicativo. 3. Aparición de notificación de barra inferior con el mensaje "Grupo de estudio creado".	Linux, windows, Android y Web.	27/02/2023

Tabla 6.3: Tabla casos de uso de pantalla con rol de Profesor. Fuente: Autoría propia.

Se realiza el video correspondiente para demostrar en material visual los casos de prueba de pantalla de rol profesor (ver en `https://youtu.be/xJTABYC6nCA`) denominado "Casos de Prueba Profesor".

Capítulo 7

Conclusiones y Trabajos Futuros

7.1 Conclusiones

Existen varias estrategias de gamificación como se investigaron en la fase uno del desarrollo conceptual sin embargo para hacer un aplicativo apropiado para su propósito y para el usuario es necesario escoger adecuadamente un grupo de estas estrategias, porque deben ser compatibles entre sí para que el usuario se sienta inmerso en una experiencia diseñada específicamente para él. Es decir, para este proyecto una forma adecuada que se encontró para combinar estas estrategias fue utilizando "Puntos y niveles", "Desafíos y misiones", "Recompensas virtuales" y "Personalización".

La gamificación como herramienta tiene un método para su utilización, este método sigue diversos pasos para poder definir que un aplicativo efectivamente esta gamificado como muestra la documentación investigada [1] y [2] en el capitulo marco teórico de este documento, pero en la practica se notó que en el proceso de desarrollo pueden intervenir sujetos externos y es necesario aplicar una nueva estrategia de gamificación para solventar este inconveniente sin necesidad de seguir los pasos del método, como sucedió con la planeación en la fase uno del desarrollo conceptual para integrar juegos en el aplicativo al no tener un objetivo especifico referente al tema que trata de enseñar, sin embargo es una estrategia para motivar al usuario mediante una actividad lúdica en el proceso de aprendizaje, mejor conocido como pausa activa.

El desarrollo del aplicativo que inicialmente fue desarrollado sobre la plataforma *Android* fue fácilmente portable a las diferentes plataformas como *iOS GNU/linux* y *Windows*, lo anterior se deduce de la fase de verificación dentro del documento.

La integración de los diferentes módulos fue realizado mediante la creación de arboles de *Widgets* dentro de la carpeta *lib* del proyecto *Flutter* diseñando las relaciones entre *Widgets* y la limitación de servicios de la arquitectura de micro-servicios, lo anterior se deduce de la fase de implementación e integración dentro del aplicativo.

La elección de los contenidos de las secciones de los libros *Artificial Intelligence: A Modern Approach. 4ed* y *Automate the Boring Stuff with Python* de los autores *Stuart Russell, Peter Norvig y Al Sweigart*, fue establecida mediante la relación del plan de estudios de los estudiantes de básica-media en Colombia y los requerimientos de conocimientos para el aprendizaje de dichas secciones, lo anterior se deduce de la fase de investigación dentro del documento.

El uso de la metodología *SCRUM* y la implementación de la arquitectura basada en micro-componentes en el desarrollo del aplicativo permiten la agilidad de desarrollo y despliegue de componentes dentro de la arquitectura de manera individual, lo anterior se deduce de las fases de implementación y verificación dentro del documento.

7.2 Trabajos futuros

Como trabajos futuros, se plantea el estudio del impacto del aplicativo en el aprendizaje de Inteligencia Artificial en estudiantes de básica-media dentro de Colegios de la cuidad de Bogotá seleccionados (Ver en Anexo AC).

Bibliografía

[1] Oriol Borrás Gené. *Fundamentos de la gamificación Universidad Politécnica de Madrid Autor: Oriol Borrás Gené*. Universidad Politecnica de Madrid, June 2015. URL: http://www.flickr.com/photos/89458386@N07/16124943257.

[2] González González Carina Soledad and Mora Alberto. "Tecnicas de gamificación aplicadas en la adocencia de IngenieriaInformatica". In: 8.1 (2015).

[3] La Guía Definitiva. *La Guía de Scrum*.

[4] Amazon Web Services. *AWS - What is a RESTful API?* 2023. URL: https://aws.amazon.com/what-is/restful-api/.

[5] *TRABAJO PRACTICO INVESTIGACION: FLUTTER 1. ¿Qué es Flutter?* URL: https://itsallwidgets.com/.

[6] Google. *Build apps for any screen*. 2022. URL: https://docs.flutter.dev/ (visited on 08/23/2022).

[7] Flutter (sitio web en español). *Flutter - Introducción a los Widgets*. 2023. URL: https://esflutter.dev/docs/development/ui/widgets.com.

[8] Microsoft .Net. *What is Xamarin?* 2022. URL: https://dotnet.microsoft.com/en-us/learn/xamarin/what-is-xamarin (visited on 08/23/2022).

[9] React Native. *React Native Getting Started*. 2022. URL: https://reactnative.dev/docs/getting-started (visited on 08/23/2022).

[10] Marina Castellote garcía. *Desarrollo de una aplicación Android de apuestas utilizando Firebase para la sincronización de datos*. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://repositori.uji.es/xmlui/bitstream/handle/10234/174192/TFG_2017_Castellote_Garcia_Marina.pdf?sequence=1#:~:text=Firebase%5C%20es%5C%20una%5C%20plataforma%5C%20desarrollada,%5C%3A%5C%20Android%5C%2C%5C%20IOS%5C%20y%5C%20web.&text=datos%5C%20y%5C%20los%5C%20sincroniza%5C%20en%5C%20tiempo%5C%20real..

[11] Amazon. *¿Qué es AWS Lambda?* 2023. URL: https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html (visited on 03/04/2023).

[12] Asun. Pié Balaguer. *Educació social i Teoria Queer*. ISBN: 8497882695.

[13] Dario Jose Espinal Ruiz, Gildardo Scarpetta Calero, and Natali Cruz Gonzalez. "Anlisis prospectivo estratégico de la educación superior en Colombia". In: *Cultura educación y sociedad* 11.1 (2020), pp. 177–196.

[14] STEM EDUCATION COLOMBIA. *¿Qué es la Educación STEM?* 2018. URL: https://www.stemeducol.com/que-es-stem (visited on 08/23/2022).

[15] Colombia Aprende Ministerio de Educación de Colombia. *Enfoque educativo STEM+ para Colombia*. 2020. URL: https://colombiaaprende.edu.co/recurso-coleccion/estrategia-de-apropiacion-stem-con-estudiantes-novacamp (visited on 08/23/2022).

[16] Ministerio de Educación Nacional. *Abiertas las inscripciones al Novacamp 2022*. 2022. URL: https://www.mineducacion.gov.co/portal/salaprensa/Noticias/409998:Abiertas-las-inscripciones-al-Novacamp-2022 (visited on 08/23/2022).

[17] Oxford Insights IDRC. "Indice de preparación del gobierno para la Inteligencia Artificial." In: (2020).

[18] Sanguinetti P. et al. "Datos e Inteligencia Artificial en el sector público". In: (2021).

[19] Technnova Angie Cárdenas. *Política Nacional para la Transformación Digital e Inteligencia Artificial*. 2019. URL: https://siteal.iiep.unesco.org/pt/node/3166 (visited on 08/23/2022).

[20] Technnova Angie Cárdenas. *La Inteligencia Artificial en Colombia: panorama general*. 2020. URL: https://tecnnova.org/la-inteligencia-artificial-en-colombia-panorama-general/ (visited on 08/23/2022).

[21] Diego Armando Bautista Díaz, Mario Francisco Suarez-Moreno, and Jhonny Gómez-Amaya. "Educación STEM en las actitudes de los estudiantes de secundaria hacia la ingeniería". In: *Revista Educación en Ingeniería* 15.29 (2020), pp. 89–103.

[22] Estela Nuñez-Barriopedro, Yeray Sanz-Gómez, and Rafael Ravina-Ripoll. "Videogames in Education: BenefitsandHarms". In: *Revista Electrónica Educare* 24.2 (2020), pp. 240–257.

[23] U.S Embassy in the Philippines. *AMERICAN SPACES EDUCATION GAMIFICATION (EDGAME) PROGRAM*. 2022. URL: https://ph.usembassy.gov/american-spaces-education-gamification-edgame-program/ (visited on 08/23/2022).

[24] Ministerio de Educación Nacional. *Aplicaciones de la inteligencia artificial en la educación*. 2022. URL: https://www.colombiaaprende.edu.co/agenda/tips-y-orientaciones/aplicaciones-de-la-inteligencia-artificial-en-la-educacion2 (visited on 08/23/2022).

[25] Ministerio de Educación Nacional. *Ministerio de Educación realizó taller de Gamificación para docentes en 'Colombia 4.0'*. 2018. URL: https://www.colombiaaprende.edu.co/agenda/tips-y-orientaciones/aplicaciones-de-la-inteligencia-artificial-en-la-educacion (visited on 08/23/2022).

[26] Yahoo Finance. *Duolingo Stock Price*. 2022. URL: https://finance.yahoo.com/quote/DUOL/ (visited on 08/17/2022).

[27] Duolingo. *Duolingo The world's best way to learn a language*. 2022. URL: https://www.duolingo.com/info (visited on 08/17/2022).

[28] Santiago Galindo Mosquera Pedro Rizzo Bajaña and Lic Juan Fernández Escobar Abg Sebastián Cadena Alvarado. *II UNIVERSIDAD DE GUAYAQUIL FACULTAD DE FILOSOFÍA, LETRAS Y CIENCIAS DE LA EDUCACIÓN CARRERA SISTEMAS MULTIMEDIA DIRECTIVOS DECANO VICEDECANO*.

[29] ACIS. *En Colombia solo el 30% de las mujeres se desempeñan en el sector tecnológico*. 2022. URL: https://acis.org.co/portal/content/noticiasdelsector/en-colombia-s%5C%C3%5C%B3lo-el-30-de-las-mujeres-se-desempe%5C%C3%5C%B1an-en-el-sector-tecnol%5C%C3%5C%B3gico.. (visited on 08/19/2022).

[30] Paloma Moreda Elena Torró Martínez Elena Lloret. *PLAYLINGUA: Exploring the application of natural language processing technologies and gamification in language learning*. Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante, 2020. (Visited on 08/17/2022).

[31] José Francisco Avila-Tomás, Miguel Angel Mayer-Pujadas, and Victor Julio Quesada-Varela. "La inteligencia artificial y sus aplicaciones en medicina I: introducción antecedentes a la IA y robótica". In: *Atención Primaria* 52.10 (2020), pp. 778–784.

[32] Lili Wu. "Construction of STEM Interdisciplinary Integration Model Supported by Educational Artificial Intelligence". In: Institute of Electrical and Electronics Engineers Inc. ISBN: 9780738131221. DOI: 10.1109/ICBAIE52039.2021.9389955.

[33] Jinsu Kim and Namje Park. "Development of a board game-based gamification learning model for training on the principles of artificial intelligence learning in elementary courses". In: *Journal of The Korean Association of Information Education* 23.3 (2019), pp. 229–235.

[34] Dona Sashini and Lankari Liyanage. *The Impact of Artificial Intelligence and Gamification Usage in Learning Management Systems*. Staffordshire University.

[35] Nisreen Alzahrani, Maram Meccawy, and Muazzam Ahmed AbdAlsamei Siddiqu. "Automatic prediction of learning styles in learning management systems: a literature review". In: *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. 2020, pp. 1–7. DOI: 10.1109/CSDE50874.2020.9411535.

[36] IBM Cloud Education. *Natural Language Processing (NLP)*. 2020. URL: https://www.ibm.com/cloud/learn/natural-language-processing (visited on 08/17/2022).

[37] Sebastian Deterding et al. "From Game Design Elements to Gamefulness: Defining "Gamification"". In: *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*. MindTrek '11. Tampere, Finland: Association for Computing Machinery, 2011, pp. 9–15. ISBN: 9781450308168. DOI: 10.1145/2181037.2181040. URL: https://doi.org/10.1145/2181037.2181040.

[38] Stefan Stieglitz Christoph Lattemann Susanne Robra-Bissantz. *Gamification*. Ed. by Stefan Stieglitz and Christof Lattemann. Springer International Publishing, 2017. ISBN: 978-3-319-45555-6. DOI: 10.1007/978-3-319-45557-0.

[39] Juho Juho Hamari Kai Huotari. "Defining Gamification: A Service Marketing Perspective". In: *Proceeding of the 16th International Academic MindTrek Conference*. MindTrek '12. Tampere, Finland: Association for Computing Machinery, 2012, pp. 17–22. ISBN: 9781450316378. DOI: 10.1145/2393132.2393137. URL: https://doi.org/10.1145/2393132.2393137.

[40] Sánchez-Cubo and Francisco. *¿Existen diferencias entre estudiantes al 'gamificar' un aula? Análisis de un Escape Room en Ciencias Sociales*. ISBN: 978-84-09-35708-6.

[41] Mercedes Fuentes-Hurtado and Juan González-Martínez. "Qué gana stem con la gamificación". In: *Academia y Virtualidad* 12 (2 Mar. 2020), pp. 79–94. DOI: 10.18359/ravi.3694.

[42] Cellular. *Cellular Mobile Gaming Grows to EUR 6 Billion in 2006*. 2006. URL: http://www.cellular.co.za/news%202003/101503-mobile%20gaming%20grows%20to%20eur%206%20bil.htm (visited on 08/17/2022).

[43] B. Gibson. *Casual Gamers and Female Gamers to Drive Mobile Games Revenues over the $ 10 Billion Mark by 2009*. 2009. URL: http://www.juniperresearch.com/shop/viewpressrelease.php?%20id=19&pr=16 (visited on 08/17/2022).

[44] Claudio E. Palazzi Fabio Aiolli. *Enhancing Artificial Intelligence on a Real Mobile Game*. Department of Pure and Applied Mathematics, 2009. DOI: https://doi.org/10.1155/2009/456169. (Visited on 08/17/2022).

[45] Du-Mim Yoon and Kyung-Joong Kim. "Challenges and Opportunities in Game Artificial Intelligence Education Using Angry Birds". In: *IEEE Access* 3 (2015), pp. 793–804. DOI: 10.1109/ACCESS.2015.2442680.

[46] Julian Togelius Georgios Yannakakis. *Artificial Intelligence and Games*. Dec. 2018. DOI: https://doi.org/10.1007/978-3-319-63519-4.

[47] Ministerio de Tecnologías de la Información y las comunicaciones. *Habilidades digitales inteligencia artificial*. 2020. URL: https://colombiatic.mintic.gov.co/679/w3-propertyvalue-228351.html (visited on 08/19/2022).

[48] Luis Corral, Alberto Sillitti, and Giancarlo Succi. "Mobile multiplatform development: An experiment for performance analysis". In: *Procedia Computer Science* 10 (2012), pp. 736–743.

[49] Wafaa S El-Kassas et al. "Taxonomy of cross-platform mobile applications development approaches". In: *Ain Shams Engineering Journal* 8.2 (2017), pp. 163–190.

[50] Yoke Seng Wong and Maizatul Hayati Mohamad Yatim. "A Propriety Multiplatform Game-Based Learning Game to Learn Object-Oriented Programming". In: *2018 7th International Congress on Advanced Applied Informatics (IIAI-AAI)*. IEEE. 2018, pp. 278–283.

[51] Andrés Aguilera Castillo et al. *Aprende jugando: el uso de técnicas de gamificación en entornos de aprendizaje*. URL: https://ciencia.lasalle.edu.co/im.

[52] Northware. *Desarrollo multiplataforma*. 2023. URL: https://www.northware.mx/blog/desarrollo-multiplataforma/#:~:text=El%5C%20desarrollo%5C%20de%5C%20software%5C%20multiplataforma,plataforma%5C%20de%5C%20hardware%5C%20en%5C%20particular. (visited on 03/12/2023).

[53] Antonio Corral, Liria Víctor, and Vázquez Rodríguez. *UNIVERSIDAD DE ALMERIA ESCUELA SUPERIOR DE INGENIERÍA TRABAJO FIN DE GRADO Desarrollo de aplicaciones móviles multiplataforma con Flutter*. 2018.

[54] Carolina pascual Villalobos. "La inteligencia artificial". In: *Atención Primaria* 1.5 (2020), pp. 2–3.

[55] Google. *Overview of Firebase*. 2023. URL: https://firebase.google.com/docs (visited on 06/15/2023).

[56] Amazon. *¿Que es python?* 2023. URL: https://aws.amazon.com/es/what-is/python/ (visited on 03/04/2023).

[57] Google Cloud. *Serverless computing*. 2023. URL: https://cloud.google.com/serverless?hl=es-419#:~:text=Desarrolla%5C%20e%5C%20implementa%5C%20aplicaciones%5C%20y,pagas%5C%20por%5C%20lo%5C%20que%5C%20usas. (visited on 03/04/2023).

[58] Amazon. *Conceptos de Amazon API Gateway*. 2023. URL: https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/api-gateway-basic-concept.html (visited on 03/04/2023).

[59] Xataka. *ChatGPT: qué es, cómo usarlo y qué puedes hacer con este chat de inteligencia artificial GPT*. 2023. URL: https://www.xataka.com/basics/chatgpt-que-como-usarlo-que-puedes-hacer-este-chat-inteligencia-artificial#:~:text=Se%5C%20trata%5C%20de%5C%20una%5C%20inteligencia,manera%5C%20convencional%5C%20y%5C%20las%5C%20entender%5C%C3%5C%A1. (visited on 03/04/2023).

[60] Sangkyun Kim et al. *Gamification Strategy*. Springer International Publishing, 2018, pp. 91–107. DOI: 10.1007/978-3-319-47283-6_8.

[61] David L. Poole and Alan K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. 3rd. Cambridge University Press, 2022.

[62] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4ed. Pearson, Apr. 2020, pp. 1–1136.

[63] Michael Negnevitsky. *Artificial Intelligence: A Systems Approach*. Pearson, 2019.

[64] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[65] George F. Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Pearson, 2008.

[66] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[67] David L. Poole and Alan K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 2017.

[68] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[69] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[70] George Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*.

[71] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009.

[72] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

[73] Al Sweigart. *Automate the Boring Stuff with Python*. Al Sweigart, June 2015, pp. 1–505.

[74] Mark Lutz. *Learning Python*. O'Reilly Media, 2013.

[75] Python Software Foundation. *Introducción a Python*. 2021. URL: https://docs.python.org/es/3/tutorial/index.html.

[76] Charles Severance. *Python for Everybody*. 2016. URL: https://www.py4e.com/book.

[77] Fireslime. *firedart*. 2023. URL: https://pub.dev/packages/firedart.

[78] microservices.io. *Microservices Architecture - microservices.io*. 2023. URL: https://microservices.io/.

[79] IBM. *IBM Cloud Architecture - Microservices*. 2023. URL: https://www.ibm.com/cloud/architecture/architectures/microservices.

[80] Federal Trade Commission. *Children's Online Privacy Protection Rule ("COPPA")*. 2023. URL: https://www.ftc.gov/legal-library/browse/rules/childrens-online-privacy-protection-rule-coppa.

[81] Gov.com. *Ley 1581 de 2012*. 2023. URL: https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=49981.

[82] GitHub. *GitHub Docs*. 2023. URL: https://docs.github.com/en.

[83] Open Source Initiative. *Open Source Initiative - The Open Source Definition*. 2023. URL: https://opensource.org/osd/.

[84] Sam Newman. *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 2015.

[85] Chris Richardson. *Microservices Patterns: With examples in Java*. Manning Publications, 2018.

[86] Irakli Nadareishvili et al. *Microservice Architecture: Aligning Principles, Practices, and Culture*. O'Reilly Media, 2016.

Capítulo 8

Anexos

8.1 Anexo A:

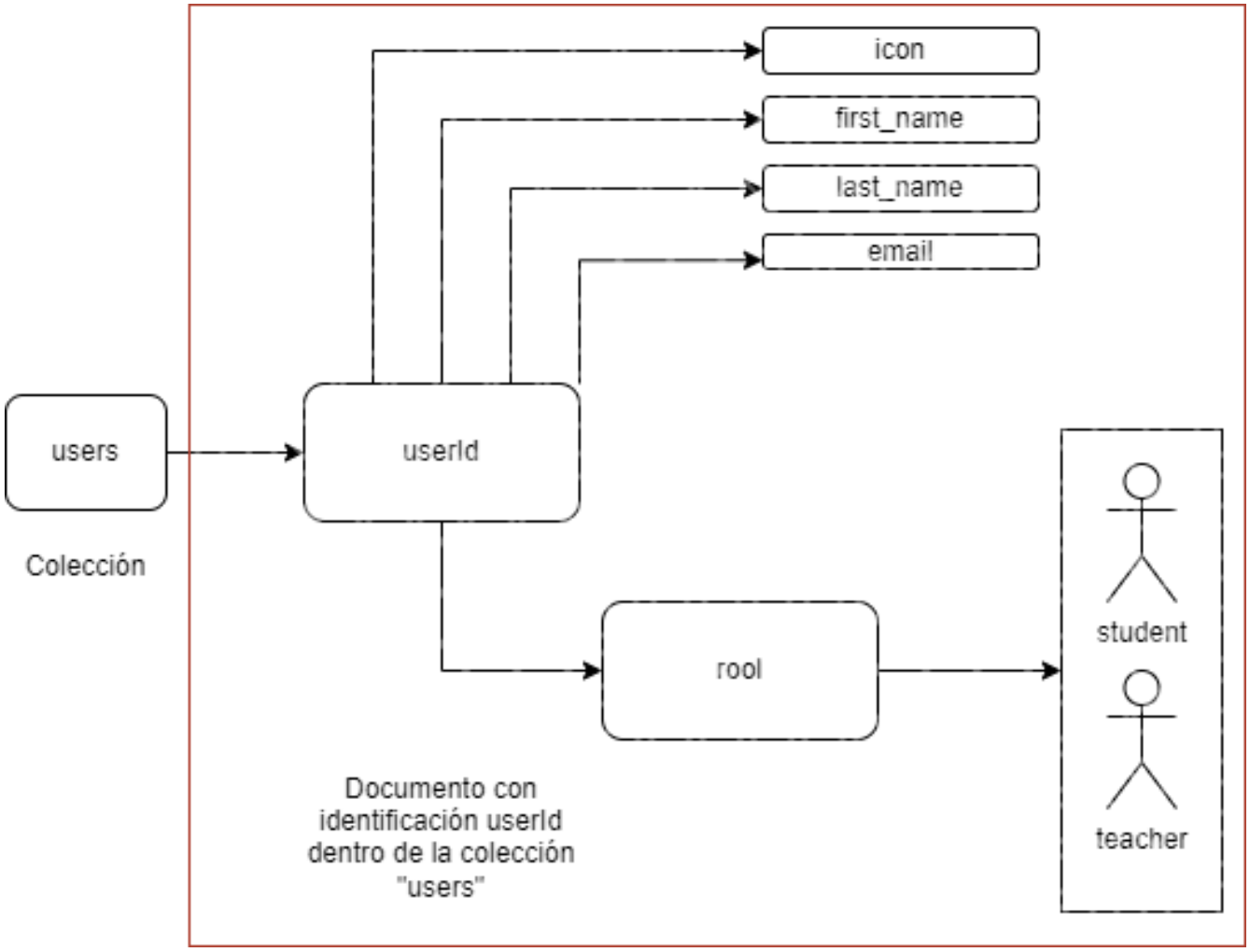

Figure 8.1: Gráfico representativo del documento *user* sin agregar los campos adicionales que dependen del campo *rool*. Fuente: Autoría propia.

8.2 Anexo B:

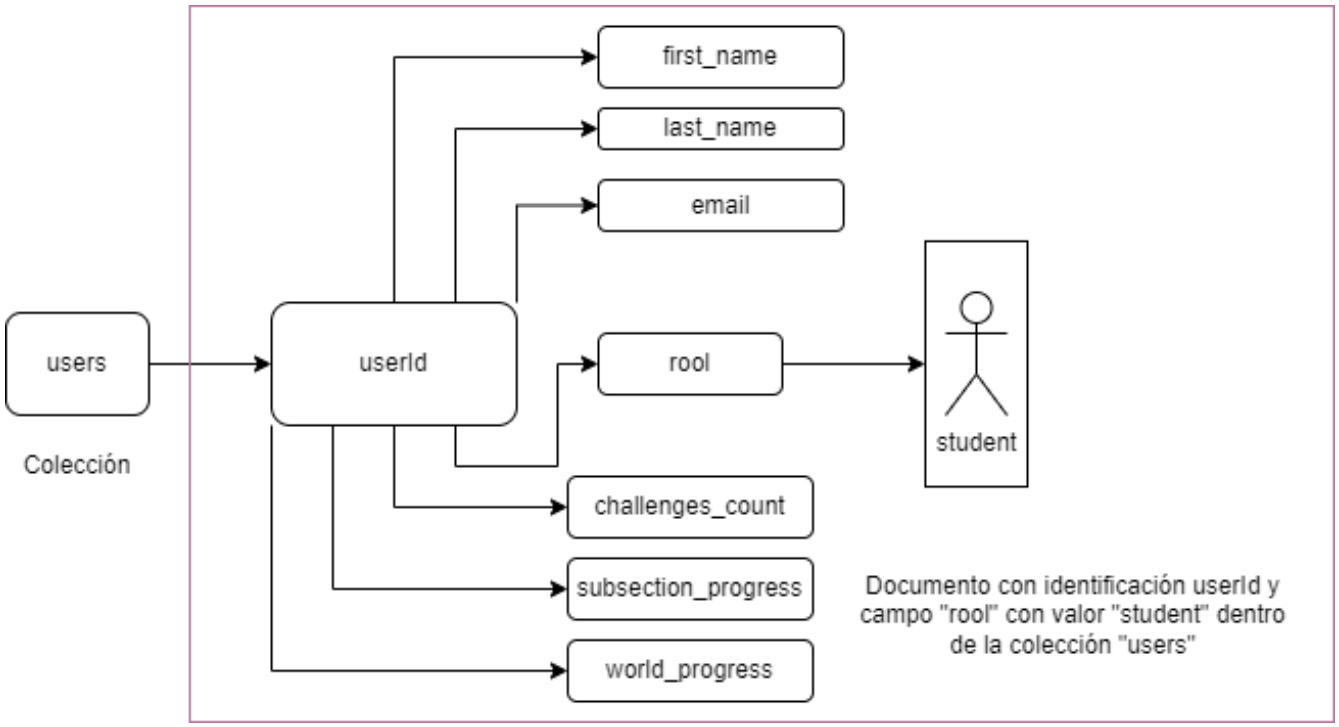

Figure 8.2: Gráfico representativo del documento *user* sin agregar los campos adicionales que dependen del campo *rool*. Fuente: Autoría propia.

8.3 Anexo C:

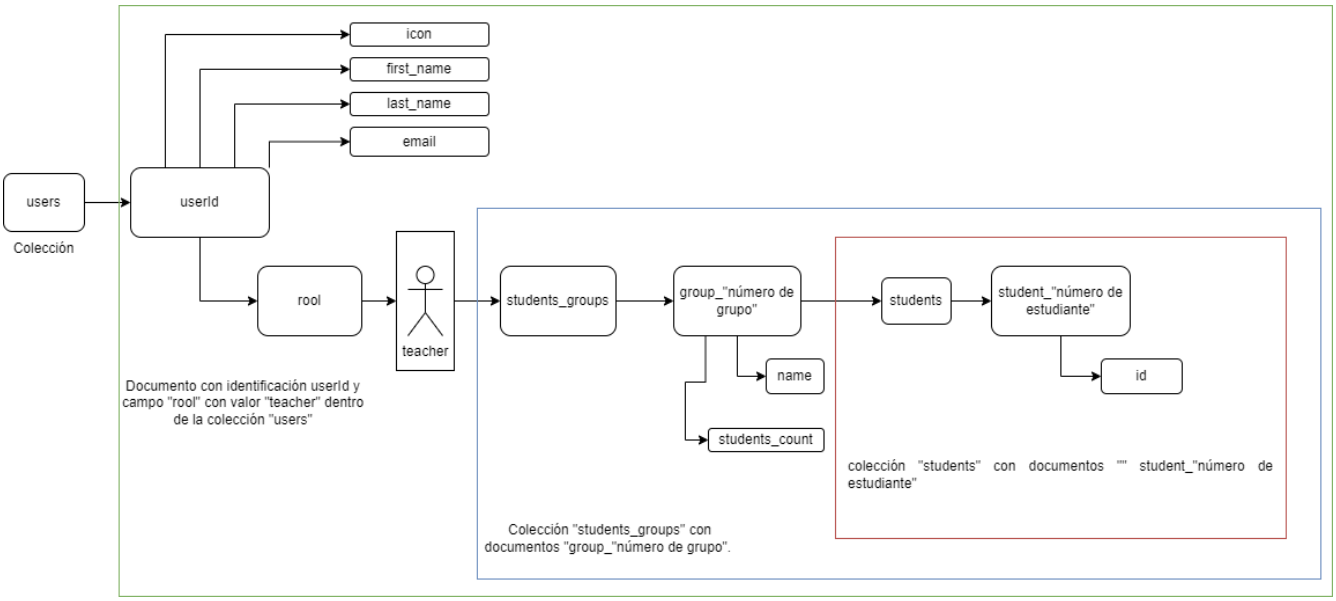

Figure 8.3: Gráfico representativo del documento *user* sin agregar los campos adicionales que dependen del campo *rool* tipo *teacher*. Fuente: Autoría propia.

8.4 Anexo D:

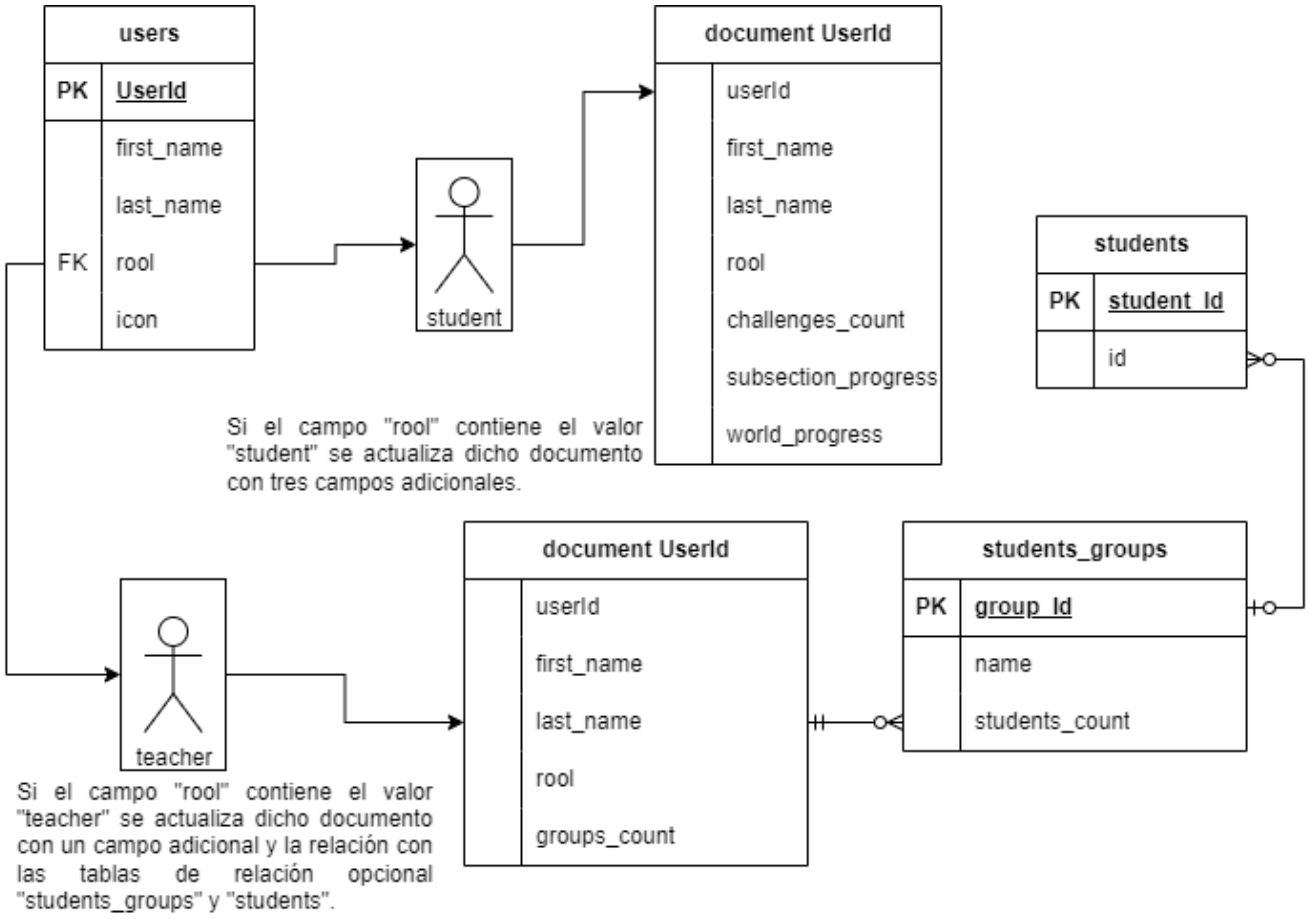

Figure 8.4: Modelo entidad-relación del modelo de datos de usuarios representando las adaptaciones del documento y colecciones dependiendo del campo *rool* tipo *student*. Fuente: Autoría propia.

8.5 Anexo E:

ID	Nombre del Mundo	Subsecciones del Mundo	Descripción del Mundo
1	Introducción a la inteligencia artificial	1. Introducción. 2. Agentes Inteligentes.	Los estudiantes pueden aprender los conceptos básicos de la inteligencia artificial, incluyendo los diferentes enfoques para su estudio.
2	Lógica Proposicional y de Primer Orden	1. Introducción a la Lógica Proposicional. 2. Inferencia en Lógica Proposicional. 3. Representación del Conocimiento en la Lógica de Primer Orden. 4. Razonamiento en Lógica de Primer Orden 5. Lógica de Descripción.	Los estudiantes pueden aprender sobre la lógica proposicional y de primer orden, que son las bases para la representación del conocimiento en la inteligencia artificial.
3	Búsqueda e Introducción a la Planificación	1. Introducción a la búsqueda en Inteligencia Artificial. 2. Búsqueda Informada (heurística) y Búsqueda No Informada. 3. Algoritmos de Búsqueda. 4. Introducción a la Planificación.	Los estudiantes pueden aprender sobre los algoritmos de búsqueda y planificación que se utilizan en la inteligencia artificial para resolver problemas.
4	Introducción al Aprendizaje Automático (Machine Learning)	1. Introducción al Aprendizaje Automático.	Los estudiantes pueden aprender sobre los algoritmos de aprendizaje automático, que se utilizan para entrenar a las computadoras para reconocer patrones y tomar decisiones basadas en datos.
5	Introducción al Procesamiento del Lenguaje Natural	1. Introducción al Procesamiento del Lenguaje Natural.	Los estudiantes pueden aprender sobre la capacidad de las computadoras para entender y procesar el lenguaje humano.
6	Introducción a la Visión Computacional	1. Introducción a la Visión Computacional.	Los estudiantes pueden aprender sobre la capacidad de las computadoras para interpretar y analizar imágenes y videos.
7	Introducción a Python	1. Fundamentos de Python. 2. Control de Flujo. 3. Funciones. 4. Listas.	Los estudiantes pueden aprender acerca del lenguaje de programación más utilizado para aplicaciones de IA conocido como Python y ser capaces de escribir código funcional para probar todos los conceptos aprendidos en los demás mundos.

Tabla 8.1: Valor de los campos de los documentos de la colección *worlds*. Fuente: Autoría propia.

8.6 Anexo F:

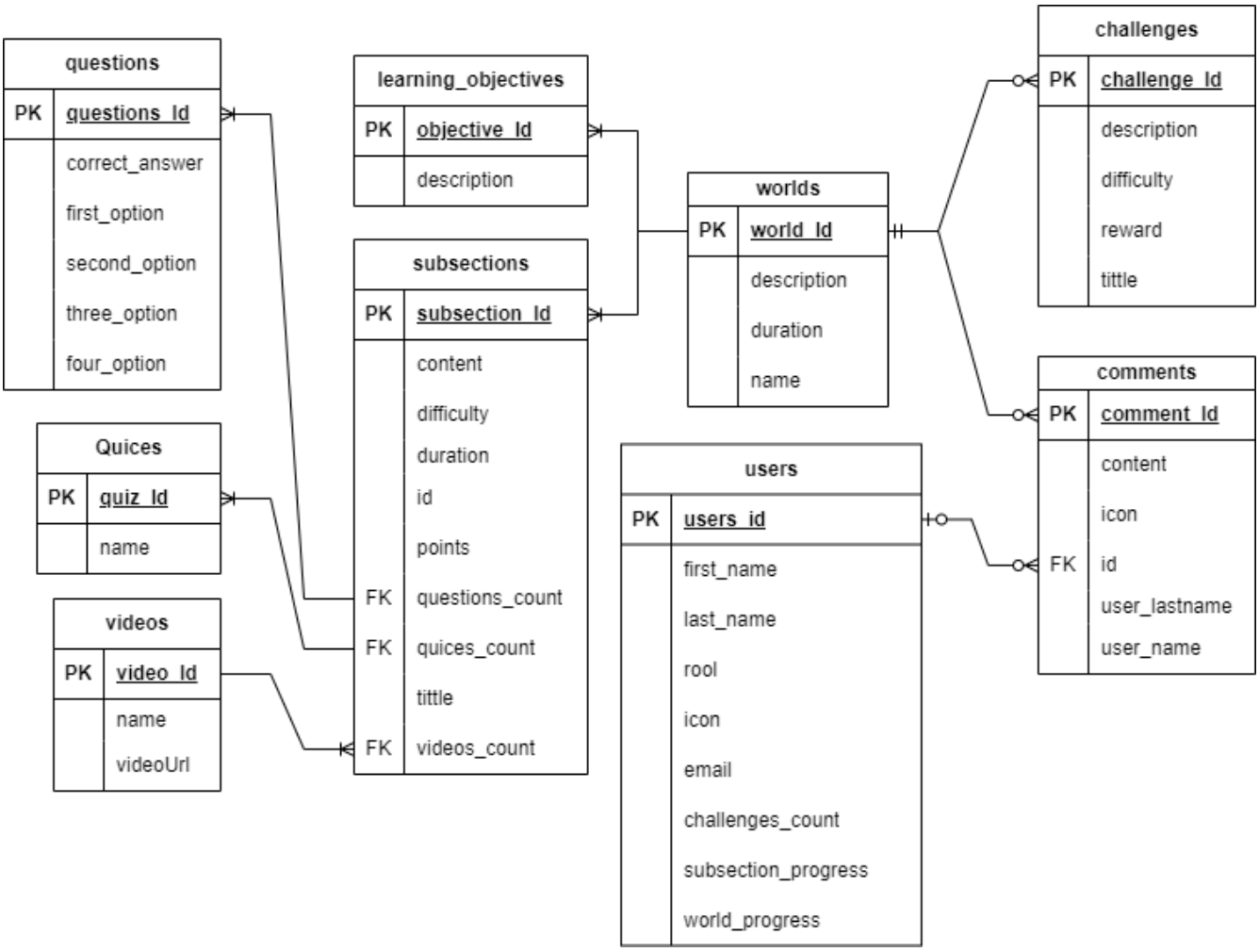

Figure 8.5: Modelo entidad-relación del modelo de datos de usuarios representando las adaptaciones del documento y colecciones dependiendo del campo *rool*. Fuente: Autoría propia.

8.7 Anexo G:

Documentos dentro de la Colección "challenges"					
world_Id	name	description	difficulty	reward	tittle
world_01	challenge_01	Vuelve a los juegos retro participando en el maravilloso mundo de PacMan.	2	3	Retro Pacman
world_02	challenge_01	En este challenge tendrás jugar el famoso juego de cartas de memoria.	4	4	Juego de Memorio

Tabla 8.2: Documentos dentro de la colección *challenges*. Fuente: Autoría propia.

8.8 Anexo H:

Documentos dentro de la Colección "worlds"			
world_Id	name	duration	description
world_01	Introducción a la Inteligencia Artificial	16min14seg	Los estudiantes pueden aprender los conceptos básicos de la inteligencia artificial, incluyendo los diferentes enfoques para su estudio.
world_02	Lógica Proposicional y de Primer Orden	28min35seg	Los estudiantes pueden aprender sobre la lógica proposicional y de primer orden, que son las bases para la representación del conocimiento en la inteligencia artificial.
world_03	Búsqueda e Introducción a la Planificación	23min31seg	Los estudiantes pueden aprender sobre los algoritmos de búsqueda y planificación que se utilizan en la inteligencia artificial para resolver problemas.
world_04	Introducción al AprendizajeAutomático (Machine Learning)	9min55seg	Los estudiantes pueden aprender sobre los algoritmos de aprendizaje automático, que se utilizan para entrenar a las computadoras para reconocer patrones y tomar decisiones basadas en datos.
world_05	Introducción al Procesamiento del Lenguaje Natural (NPL)	7min8seg	Los estudiantes pueden aprender sobre la capacidad de las computadoras para entender y procesar el lenguaje humano.
world_06	Introducción a la Visión Computacional	12min50seg	Los estudiantes pueden aprender sobre la capacidad de las computadoras para interpretar y analizar imágenes y videos.
world_07	Introducción a Python	1h2min36seg	Los estudiantes pueden aprender acerca del lenguaje de programación más utilizado para aplicaciones de IA conocido como Python y ser capaces de escribir código funcional para probar todos los conceptos aprendidos en los demás mundos.

Tabla 8.3: Documentos dentro de la colección *worlds*. Fuente: Autoría propia.

8.9 Anexo I:

Colección learning_objectives		
Id documento	Cantidad de documentos dentro de la colección	Contenido campo "description" dentro de cada documento
world_01	4	1, Comprender el concepto y las aplicaciones de la inteligencia artificial. 2. Conocer las diferentes ramas de la inteligencia artificial y sus aplicaciones. 3. Identificar las áreas de aplicación de la inteligencia artificial en la vida diaria y en diferentes industrias. 4. Explorar el concepto de agentes inteligentes dentro del marco de la inteligencia artificial.
world_02	5	1.Comprender los conceptos fundamentales de la lógica proposicional y de primer orden. 2. Entender de manera básica el concepto de inferencia en lógica proposicional. 3. Comprender los conceptos fundamentales de la representación del conocimiento en lógica de primer orden. 4. Aprender sobre el concepto de razonamiento en lógica de primer orden. 5. Comprender los conceptos fundamentales de la lógica de descripción.
world_03	3	1. Comprender los conceptos básicos de búsqueda dentro del marco de la Inteligencia Artificial y los diferentes tipos de búsqueda. 2. Definir y diferenciar los diferentes tipos de búsqueda. 3. Comprender el concepto de Algoritmos de búsqueda, sus diferentes tipos y su desarrollo en el lenguaje de programación Python.
world_04	2	1. Comprender los conceptos introductorios al aprendizaje automático, así como su definición y aplicaciones. 2. Comprender sobre los diferentes tipos de aprendizaje automáticos dentro del marco de aprendizaje automático y sus respectivas diferencias.
world_05	2	1. Comprender los conceptos introductorios al procesamiento del lenguaje natural, así como su definición y aplicaciones. 2. Comprender sobre los diferentes tipos de procesamiento del lenguaje natural dentro del marco de inteligencia artificial y sus respectivas diferencias.
world_06	1	1. Comprender los conceptos introductorios a la visión computacional, así como su definición y aplicaciones.
world_07	4	1. Comprender los conceptos básicos del lenguaje de programación Python, como su sintaxis, historia, proceso de instalación, condicionales, ciclos repetitivos y tipos de datos.2. Comprender el concepto de control de flujo y ser capaz de aplicarlo dentro de Python. 3. Comprender el concepto de Funciones dentro del marco de Python, como definirlas, tipos de funciones y ser capaz de realizar funciones dentro del lenguaje. 4. Comprender el concepto de Listas dentro del marco de Python, como definirlas, creación de estas y ser capaz de realizar listas dentro del lenguaje.

Tabla 8.4: Valores de campo de los documentos *world_Id* de la colección *learning_objectives*. Fuente: Autoría propia.

8.10 Anexo J:

En este anexo se encuentra la tabla correspondiente a los valores de campo de la colección *subsections* (Ver siguiente página).

Colección subsections									
Id del mundo	Cantidad de documentos dentro de la colección	content	duration	points	tittle	difficulty	questions _count	quices _count	videos _count
world_01	2	1. En esta sección queremos que te embarques con nosotros en el estudio de la Inteligencia Artificial y que veas por que es un tema de estudio importante, para ello es importante que sepas exactamente que es. 2. La naturaleza de los agentes es un concepto fundamental para el estudio de la Inteligencia Artificial en esta sección te explicaremos que es precisamente un agente, la diversidad de los ambientes en los que se envuelven y sus diferentes tipos.	1. 10min32seg 2. 5min42seg	1. 100 pts 2. 100 pts	1. Introducción. 2. Agentes Inteligentes.	1. 1 2. 2	1. 5 2. 5	1. 1 2. 1	1. 1 2. 1
world_02	5	1. En esta subsección se aborda la lógica proposicional como un sistema formal utilizado para representar y razonar sobre proposiciones y sus relaciones lógicas. Se explicarán los conceptos básicos y la resolución de problemas utilizando lógica proposicional. 2. Aquí se exploran los métodos y técnicas utilizados para realizar inferencias en lógica proposicional. Se explicarán los principios básicos dentro de las inferencias en lógica proposicional. 3. En esta subsección se introduce la lógica de primer orden como un sistema más expresivo que la lógica proposicional. Se exploran los conceptos básicos dentro de la lógica de primer orden. 4. Aquí se profundiza en el razonamiento utilizando la lógica de primer orden. Se exploran los métodos de inferencia y resolución utilizados en la lógica de primer orden. 5. Esta subsección se centra en la lógica de descripción, que es una extensión de la lógica de primer orden utilizada para representar y razonar sobre conceptos complejos. Se explicarán los conceptos básicos de la lógica de descripción.	1. 4min36seg 2. 4min11seg 3. 3min55seg 4. 8min13seg 5. 7min13seg	1. 200 pts 2. 200 pts 3. 200 pts 4. 200 pts 5. 200 pts	1. Introducción a la Lógica Proposicional. 2. Inferencia en Lógica Proposicional. 3. Representación del Conocimiento en la Lógica de primer Orden. 4. Razonamiento en Lógica de Primer Orden. 5. Lógica de Descripción.	1. 2 2. 2 3. 3 4. 3 5. 3	1. 5 2. 5 3. 3 4. 4 5. 5	1. 1 2. 1 3. 1 4. 1 5. 1	1. 1 2. 1 3. 1 4. 1 5. 1
world_03	4	1. En esta subsección se aborda la búsqueda en espacios de estados como un enfoque fundamental para resolver problemas en inteligencia artificial. Se explican de manera introductoria y básica los conceptos fundamentales dentro de la búsqueda en el marco de la inteligencia artificial. 2. Aquí se exploran algoritmos de búsqueda de manera básica dentro de la búsqueda informada y no informada. 3. Esta subsección se centra en la explicación del funcionamiento de algunos algoritmos de búsqueda, así como de su implementación en Python. 4. Aquí se introduce el concepto de planificación de manera básica dentro del marco de la Inteligencia Artificial.	1. 8min1seg 2. 5min45seg 3. 6min12seg 4. 3min32seg	1. 300 pts 2. 300 pts 3. 300 pts 4. 300 pts	1. Introducción a la búsqueda en Inteligencia Artificial. 2. Búsqueda Informada(heurística) y Búsqueda No Informada (ciega). 3. Algoritmos de búsqueda. 4. Introducción a la planificación.	1. 3 2. 3 3. 3 4. 4 5. 4	1. 5 2. 5 3. 5 4. 5	1. 1 2. 1 3. 1 4. 1	1. 1 2. 1 3. 1 4. 1
world_04	1	1. En esta subsección se explora conceptos básicos sobre el área del Aprendizaje Automático (Machine Learning) una de las áreas de aplicación más importantes dentro de la Inteligencia Artificial.	1. 9min54seg	1. 400 pts	1. Introducción al Aprendizaje Automático.	1. 5	1. 5	1. 1	1. 1
world_05	1	1. En esta subsección se explora conceptos básicos sobre el área del Procesamiento del Lenguaje Natural, una de las áreas de aplicación más importantes dentro de la Inteligencia Artificial.	1. 7min8seg	1. 500 pts	1. Introducción al Procesamiento del Lenguaje Natural.	1. 5	1. 5	1. 1	1. 1
world_06	1	1. En esta subsección se explora conceptos básicos sobre el área de la Visión Computacional, una de las áreas de aplicación más importantes dentro de la Inteligencia Artificial.	1. 12min49seg	1. 600 pts	1. Introducción a la Visión Computacional	1. 5	1. 5	1. 1	1. 1
world_07	4	1. En esta sección aprenderás sobre los fundamentos del lenguaje de programación Python, historia, sintaxis, instalación y demás aspectos básicos del lenguaje. 2. En este apartado entraremos en lo que es control de flujo y la lógica de programación en Python. 3. Nos adentraremos en el mundo de las funciones, como declararlas, los tipos de funciones disponibles y demás que nos ayudarán a comprender y escribir código en Python de manera más ordenada. 4. La subsección de listas en Python es fundamental para comprender cómo trabajar con conjuntos de datos y estructuras de almacenamiento en programas. En esta subsección, se aborda el de manera básica el concepto de listas.	1. 28min41seg 2. 16min31seg 3. 7min45seg 4. 8min46seg	1. null 2. null 3. null 4. null	1. Fundamentos de Python. 2. Control de Flujo. 3. Funciones. 4. Listas.	1. 1 2. 1 3. 1 4. 1	1. null 2. null 3. null 4. null	1. null 2. null 3. null 4. null	1. 3 2. 1 3. 1 4. 1

Tabla 8.5: Valores de campo de la colección *subsections*. Fuente: Autoría propia.

8.11 Anexo K:

Tabla con el contenido de la colección *questions* dentro de los mundos con sistema de calificación.

Colección quiess									
Id del documento dentro de la colección "worlds"	Id del documento dentro de la colección "subsection"	Id del documento dentro de la colección "quizes"	Id del documento dentro de la colección "questions"	question	correct_answer	first_option	second_option	three_option	four_option
world_01	subsection_01	quiz_01	question_01	¿Qué es la inteligencia artificial?	Un campo de estudio que se ocupa de crear sistemas inteligentes	Un campo de estudio que se ocupa de crear sistemas inteligentes	Un videojuego popular.	Una forma de vida extraterrestre.	Un tipo de robot.
			question_02	¿Cuál de las siguientes tareas NO está relacionada con la IA?	Reparación de automóviles	Clasificación de imágenes	Traducción automática de idiomas	Análisis de datos.	Reparación de automóviles
			question_03	Cuál de los siguientes ejemplos NO representa un caso de uso de la IA?	Envío de correos electrónicos	Asistentes virtuales como Siri o Alexa	Reconocimiento facial en fotografías	Sistemas de recomendación en plataformas de streaming	Envío de correos electrónicos
			question_04	¿Qué es el aprendizaje automático (machine learning)?	Un enfoque de la IA que permite a las máquinas aprender a partir de datos	Un algoritmo utilizado para construir robots autónomos	Un enfoque de la IA que permite a las máquinas aprender a partir de datos	Una forma de programación manual de sistemas inteligentes	Una técnica utilizada para el procesamiento de imágenes
			question_05	¿Cuál de los siguientes es un subcampo de la IA?	Redes neuronales artificiales	Redes neuronales artificiales	Ciencias biológicas	Mecánica cuántica	Economía internacional
	subsection_02	quiz_01	question_01	¿Qué es un agente inteligente?	Un sistema capaz de tomar decisiones y actuar en un entorno para alcanzar objetivos	Un programa de computadora que realiza tareas sin supervisión	Un sistema capaz de tomar decisiones y actuar en un entorno para alcanzar objetivos	Una colección de algoritmos de clasificación de datos	Un programa de computadora que realiza tareas sin supervisión
			question_02	¿Cuál de las siguientes características NO describe a un agente inteligente?	Capacidad de ejecutar tareas complejas sin programación previa	Capacidad de percepción del entorno	Capacidad de razonamiento y toma de decisiones	Capacidad de aprendizaje automático	Capacidad de ejecutar tareas complejas sin programación previa
			question_03	¿Cuál de las siguientes técnicas se utilizan comúnmente para implementar agentes inteligentes?	Algoritmos de búsqueda y planificación	Programación lineal	Programación orientada a objetos	Algoritmos de búsqueda y planificación	Protocolo de transferencia de hipertexto (HTTP)
			question_04	¿Qué es el aprendizaje por refuerzo en un agente inteligente?	La capacidad de recibir recompensas o castigos según sus acciones y ajustar su comportamiento en consecuencia	El proceso de adquirir conocimientos y habilidades por sí mismo	La capacidad de recibir recompensas o castigos según sus acciones y ajustar su comportamiento en consecuencia	La habilidad de comunicarse con otros agentes inteligentes	Un enfoque basado en reglas lógicas para la toma de decisiones
			question_05	¿Cuál de los siguientes es un ejemplo de un agente inteligente en el mundo real?	Un sistema de recomendación de películas	Un programa de correo electrónico	Un algoritmo de compresión de archivos	Una sistema de recomendación de películas	Un reloj despertador
world_02	subsection_01	quiz_01	question_01	¿Qué es la lógica proposicional?	Un sistema formal para el razonamiento matemático	Un sistema formal para el razonamiento matemático	Un conjunto de reglas para la programación de computadoras	Una rama de la filosofía que estudia la naturaleza del conocimiento	Un enfoque de la psicología para el estudio de la cognición
			question_02	¿Cuál es el elemento básico de la lógica proposicional?	Variables	Variables	Funciones matemáticas	Números enteros	Cláusulas condicionales
			question_03	¿Cuál de las siguientes afirmaciones es verdadera sobre la negación en la lógica proposicional?	La negación de una proposición falsa siempre es verdadera	La negación de una proposición verdadera siempre es falsa	La negación de una proposición verdadera siempre es verdadera	La negación de una proposición verdadera siempre es verdadera	La negación de una proposición falsa siempre es falsa
			question_04	¿Cuál es la conectiva lógica que representa la conjunción (AND)?	~	~	v	->	~
			question_05	¿Cuál de las siguientes afirmaciones describe la implicación lógica (IF-THEN)?	Si p es verdadera, entonces q es falsa	Si p es verdadera, entonces q es verdadera	Si p es falsa, entonces q es verdadera	Si p es verdadera, entonces q es falsa	Si q es verdadera, entonces p es verdadera
	subsection_02	quiz_01	question_01	¿Qué es la inferencia en lógica proposicional?	El proceso de obtener nuevas proposiciones a partir de proposiciones existentes	El proceso de obtener nuevas proposiciones a partir de proposiciones existentes	El proceso de evaluar la verdad o falsedad de una proposición	El estudio de las reglas de deducción matemática	La aplicación de la lógica a la inteligencia artificial
			question_02	¿Cuál de las siguientes reglas de inferencia permite concluir una disyunción?	Silogismo Disyuntivo	Modus Ponens	Modus Tollens	Silogismo Disyuntivo	Silogismo Hipotético
			question_03	¿Cuál de las siguientes afirmaciones es verdadera sobre la regla de inferencia de la adición (Addition)?	Permite concluir una disyunción a partir de dos proposiciones	Permite concluir una conjunción a partir de dos proposiciones	Permite concluir una disyunción a partir de dos proposiciones	Permite concluir una implicación a partir de dos proposiciones	Permite concluir una negación a partir de una proposición
			question_04	¿Cuál de las siguientes reglas de inferencia permite eliminar una doble negación en una proposición?	Doble Negación	Modus Ponens	Modus Tollens	Dilema Constructivo	Doble Negación
			question_05	¿Cuál de las siguientes reglas de inferencia permite concluir una implicación a partir de una conjunción?	Silogismo Hipotético	Modus Ponens	Modus Tollens	Silogismo Hipotético	Silogismo Disyuntivo
	subsection_03	quiz_01	question_01	¿Qué es la lógica de primer orden en la representación del conocimiento?	Un enfoque para representar conocimiento con mayor expresividad, incluyendo variables cuantificadas y predicados	Un sistema formal para representar proposiciones lógicas básicas	Un enfoque para representar conocimiento con mayor expresividad, incluyendo variables cuantificadas y predicados	Un conjunto de reglas para la programación de sistemas expertos	Una técnica de aprendizaje automático para la adquisición de conocimiento
			question_02	¿Qué es un símbolo de función en la lógica de primer orden?	Un símbolo utilizado para representar operaciones o funciones que toman argumentos y devuelven un resultado	Un símbolo utilizado para representar objetos o individuos específicos	Un símbolo utilizado para representar objetos o individuos específicos	Un símbolo utilizado para representar operaciones o funciones que toman argumentos y devuelven un resultado	Un símbolo utilizado para representar conceptos o categorías generales
			question_03	¿Cuál de las siguientes opciones representa correctamente la cuantificación existencial ( ) en la lógica de primer orden?	Existe al menos un x	Para todo x	No existe x	Existe al menos un x	Para algún x
	subsection_04	quiz_01	question_01	¿Qué es el razonamiento en lógica de primer orden?	El proceso de inferir nuevas conclusiones a partir de premisas utilizando reglas de inferencia	El proceso de inferir nuevas conclusiones a partir de premisas utilizando reglas de inferencia	El proceso de codificar conocimiento en una base de datos relacional	El proceso de evaluar la verdad o falsedad de una proposición mediante pruebas formales	El proceso de realizar operaciones lógicas en un circuito digital
			question_02	¿Cuál de las siguientes reglas de inferencia se utiliza para eliminar una cuantificación existencial ()?	Generalización Existencial	Modus Ponens	Modus Tollens	Generalización Existencial	Eliminación Universal
			question_03	¿Cuál de las siguientes opciones describe correctamente la resolución en lógica de primer orden?	Una técnica de inferencia que busca encontrar una contradicción entre cláusulas para inferir una conclusión	Una técnica de inferencia que utiliza la regla de modus ponens para derivar conclusiones lógicas	Una técnica de inferencia que busca encontrar una contradicción entre cláusulas para inferir una conclusión	Una técnica de inferencia que utiliza la cuantificación universal () para establecer la verdad de una proposición	Una técnica de inferencia que utiliza la regla de eliminación existencial para inferir una conclusión
			question_04	¿Cuál de las siguientes opciones representa correctamente la regla de modus tollens en lógica de primer orden?	Si p entonces q, q es falsa, por lo tanto p es falsa	Si p entonces q, p es verdadera, por lo tanto q es verdadera	Si p entonces q, q es falsa, por lo tanto p es falsa	Si p entonces q, p es falsa, por lo tanto q es verdadera	Si p entonces q, q es verdadera, por lo tanto p es verdadera
	subsection_05	quiz_01	question_01	¿Qué es la lógica de descripción?	Un lenguaje formal para describir conceptos y relaciones en dominios de conocimiento	Un enfoque de la inteligencia artificial que utiliza algoritmos de búsqueda	Un sistema formal para el razonamiento lógico en filosofía	Un lenguaje formal para describir conceptos y relaciones en dominios de conocimiento	Una técnica de programación para el desarrollo de aplicaciones web
			question_02	¿Cuál de las siguientes afirmaciones describe correctamente una restricción de lógica de descripción?	Es una expresión lógica que impone una condición sobre los individuos que satisfacen una descripción	Es una expresión lógica que impone una condición sobre los individuos que satisfacen una descripción	Es una regla de inferencia que permite derivar nuevas descripciones a partir de premisas	Es una función matemática que relaciona diferentes descripciones entre sí	Es un operador lógico utilizado para combinar múltiples descripciones en una sola
			question_03	¿Cuál de las siguientes opciones describe correctamente la lógica de descripción DL (Description Logic)?	Es una variante de la lógica proposicional que permite la cuantificación y la descripción de conceptos	Es una variante de la lógica proposicional que permite la cuantificación y la descripción de conceptos	Es una variante de la lógica de primer orden que utiliza cláusulas de Horn para representar conocimiento	Es una variante de la lógica de primer orden que utiliza árboles de decisión para el razonamiento lógico	Es una variante de la lógica de primer orden que se utiliza exclusivamente en el ámbito de la inteligencia artificial
			question_04	¿Cuál de las siguientes afirmaciones es verdadera sobre la lógica de descripción?	Se utiliza en áreas como la inteligencia artificial y la web semántica	Permite la representación de conocimiento incierto o probabilístico	No permite la representación de relaciones entre conceptos o individuos	Se utiliza exclusivamente en la programación de bases de datos relacionales	Se utiliza en áreas como la inteligencia artificial y la web semántica
			question_05	¿Cuál de las siguientes opciones describe correctamente el razonamiento en la lógica de descripción?	Permite inferir relaciones y propiedades sobre individuos y conceptos utilizando restricciones lógicas	Permite inferir nuevas descripciones a partir de descripciones existentes utilizando reglas de inferencia	Permite inferir conclusiones lógicas a partir de proposiciones utilizando operadores lógicos	Permite inferir relaciones y propiedades sobre individuos y conceptos utilizando restricciones lógicas	Permite inferir la verdad o falsedad de una proposición utilizando árboles de decisión
	subsection_01	quiz_01	question_01	¿Qué es la búsqueda en Inteligencia Artificial?	Un proceso de encontrar soluciones óptimas a problemas complejos mediante la exploración sistemática de un espacio de estados	Un proceso de encontrar soluciones óptimas a problemas complejos mediante la exploración sistemática de un espacio de estados	Un proceso de recopilación y análisis de datos para entrenar modelos de aprendizaje automático	Un proceso de generar de forma aleatoria posibles soluciones a un problema	Un proceso de inferir conclusiones a partir de premisas utilizando reglas de inferencia
			question_02	¿Cuál de las siguientes estrategias de búsqueda es informada?	Búsqueda heurística	Búsqueda en anchura (BFS)	Búsqueda en profundidad (DFS)	Búsqueda de costo uniforme	Búsqueda heurística
			question_03	¿Cuál de las siguientes afirmaciones es verdadera sobre la búsqueda en anchura (BFS)?	Explora primero los nodos más cercanos al nodo inicial del árbol de búsqueda	Explora primero los nodos más profundos del árbol de búsqueda	Explora primero los nodos más cercanos al nodo inicial del árbol de búsqueda	Utiliza una función heurística para evaluar la calidad de los nodos en el árbol de búsqueda	Utiliza una pila para almacenar los nodos frontera en el árbol de búsqueda
			question_04	¿Cuál de las siguientes opciones describe correctamente la búsqueda de costo uniforme?	Expande los nodos del árbol de búsqueda en función del costo acumulado hasta el nodo actual	Expande los nodos del árbol de búsqueda en función del costo acumulado hasta el nodo actual	Expande los nodos del árbol de búsqueda de forma aleatoria sin considerar el costo	Expande los nodos del árbol de búsqueda en función de una función heurística estimada	Expande los nodos del árbol de búsqueda en función de la profundidad del nodo en el árbol
			question_05	¿Cuál de las siguientes opciones describe correctamente la búsqueda heurística?	Utiliza una función de evaluación heurística para estimar el costo o la calidad de un estado	Utiliza una estrategia de búsqueda en anchura para explorar el espacio de estados	Utiliza una función de evaluación heurística para estimar el costo o la calidad de un estado	Utiliza una estrategia de búsqueda en profundidad para explorar el espacio de estados	Utiliza una función de evaluación heurística para generar soluciones aleatorias
	subsection_02	quiz_01	question_01	¿Cuál es la principal diferencia entre la búsqueda informada (heurística) y la búsqueda no informada (ciega)?	La búsqueda informada utiliza funciones heurísticas para guiar la exploración, mientras que la búsqueda no informada no las utiliza	La búsqueda informada utiliza funciones heurísticas para guiar la exploración, mientras que la búsqueda no informada no las utiliza	La búsqueda informada siempre encuentra la solución óptima, mientras que la búsqueda no informada puede encontrar soluciones subóptimas	La búsqueda informada utiliza algoritmos de aprendizaje automático, mientras que la búsqueda no informada no los utiliza	La búsqueda informada es más rápida que la búsqueda no informada en la mayoría de los casos
			question_02	¿Cuál es un ejemplo de algoritmo de búsqueda informada?	Algoritmo A*	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Algoritmo A*
			question_03	¿Cuál es un ejemplo de algoritmo de búsqueda no informada?	Búsqueda en profundidad (DFS)	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Algoritmo A*	Búsqueda de costo uniforme
			question_04	¿Qué tipo de información utiliza la búsqueda informada para guiar la exploración?	Información sobre la distancia estimada desde el estado actual hasta el objetivo	Información sobre el costo acumulado desde el estado inicial hasta el estado actual	Información sobre la profundidad de los nodos en el árbol de búsqueda	Información sobre la distancia estimada desde el estado actual hasta el objetivo	Información sobre la cantidad de nodos visitados durante la exploración
			question_05	¿Cuál de las siguientes afirmaciones es verdadera sobre la búsqueda informada?	Puede encontrar soluciones subóptimas en algunos casos	Siempre encuentra la solución óptima	Es más lenta que la búsqueda no informada en la mayoría de los casos	No utiliza ninguna información adicional para guiar la exploración	Puede encontrar soluciones subóptimas en algunos casos
	subsection_03	quiz_01	question_01	¿Qué algoritmo de búsqueda utiliza una cola FIFO (First-In, First-Out) para expandir los nodos en el árbol de búsqueda?	Búsqueda en anchura (BFS)	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Búsqueda A*
			question_02	¿Cuál de los siguientes algoritmos de búsqueda utiliza una pila LIFO (Last-In, First-Out) para expandir los nodos en el árbol de búsqueda?	Búsqueda en profundidad (DFS)	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Búsqueda A*
			question_03	¿Qué algoritmo de búsqueda utiliza una función heurística para guiar la exploración hacia los nodos más prometedores en el árbol de búsqueda?	Búsqueda A*	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Búsqueda A*
			question_04	¿Cuál de los siguientes algoritmos de búsqueda expande los nodos del árbol de búsqueda en función del costo acumulado hasta el nodo actual?	Búsqueda de costo uniforme	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Búsqueda A*
			question_05	¿Cuál de los siguientes algoritmos de búsqueda es una combinación de búsqueda en anchura y búsqueda de costo uniforme, utilizando una función heurística para estimar el costo restante?	Búsqueda A*	Búsqueda en profundidad (DFS)	Búsqueda en anchura (BFS)	Búsqueda de costo uniforme	Búsqueda A*
world_03	subsection_04	quiz_01	question_01	¿Qué es la planificación en el contexto de la Inteligencia Artificial?	Un proceso de encontrar una secuencia de acciones para alcanzar un objetivo en un entorno dinámico	Un proceso de tomar decisiones óptimas basadas en reglas predefinidas	Un proceso de generar soluciones aleatorias para un problema	Un proceso de encontrar una secuencia de acciones para alcanzar un objetivo en un entorno dinámico	Un proceso de recopilación y análisis de datos para lentrenar modelos de aprendizaje automático
			question_02	¿Cuál es el objetivo principal de la planificación en Inteligencia Artificial?	Encontrar una solución óptima para un problema dado	Encontrar una solución óptima para un problema dado	Encontrar una solución óptima para un problema dado	Aprender a tomar decisiones basadas en datos históricos	Explorar todas las posibles soluciones de un problema
			question_03	¿Cuál de las siguientes afirmaciones describe correctamente la diferencia entre la planificación determinista y la planificación no determinista?	La planificación determinista considera solo acciones deterministas, mientras que la planificación no determinista considera acciones probabilísticas	La planificación determinista considera solo un estado inicial, mientras que la planificación no determinista considera múltiples estados iniciales	La planificación determinista encuentra siempre la solución óptima, mientras que la planificación no determinista puede encontrar soluciones subóptimas	La planificación determinista considera solo acciones determinadas, mientras que la planificación no determinista considera acciones probabilísticas	La planificación determinista utiliza algoritmos de búsqueda, mientras que la planificación no determinista utiliza algoritmos genéticos
			question_04	¿Qué tipo de conocimiento se utiliza en la planificación basada en reglas?	Conocimiento declarativo	Conocimiento heurístico	Conocimiento declarativo	Conocimiento estadístico	Conocimiento procedural
			question_05	¿Cuál es uno de los desafíos principales en la planificación en entornos dinámicos y parcialmente observables?	La incertidumbre sobre el estado del entorno	La falta de algoritmos de búsqueda adecuados	La falta de recursos computacionales	La incertidumbre sobre el estado del entorno	La falta de conocimiento heurístico
world_04	subsection_01	quiz_01	question_01	¿Qué es el aprendizaje automático (machine learning)?	Un enfoque de la inteligencia artificial que permite a las máquinas aprender automáticamente a partir de datos	Un proceso de enseñanza de habilidades a las máquinas	Un proceso de programación de algoritmos complejos	Un enfoque de la inteligencia artificial que permite a las máquinas aprender automáticamente a partir de datos	Un proceso de automatización de tareas rutinarias
			question_02	¿Cuál de los siguientes opciones describe mejor el aprendizaje supervisado?	El modelo aprende a partir de ejemplos de entrada y su salida correspondiente	El modelo aprende a partir de ejemplos de entrada y su salida correspondiente	El modelo aprende a partir de su propia experiencia sin ejemplos de entrada y salida	El modelo aprende a partir de recompensas y castigos	El modelo aprende a partir de la interacción con un entorno
			question_03	¿Cuál de las siguientes opciones describe mejor el aprendizaje no supervisado?	El modelo aprende a partir de su propia experiencia sin ejemplos de entrada y salida	El modelo aprende a partir de ejemplos de entrada y su salida correspondiente	El modelo aprende a partir de su propia experiencia sin ejemplos de entrada y salida	El modelo aprende a partir de recompensas y castigos	El modelo aprende a partir de la interacción con un entorno
			question_04	¿Cuál de las siguientes opciones describe mejor el aprendizaje por refuerzo?	El modelo aprende a partir de recompensas y castigos	El modelo aprende a partir de ejemplos de entrada y su salida correspondiente	El modelo aprende a partir de su propia experiencia sin ejemplos de entrada y salida	El modelo aprende a partir de recompensas y castigos	El modelo aprende a partir de la interacción con un entorno
			question_05	¿Cuál es el objetivo principal del aprendizaje automático?	Permitir a las máquinas aprender automáticamente a partir de datos	Automatizar tareas rutinarias	Resolver problemas de manera eficiente	Permitir a las máquinas aprender automáticamente a partir de datos	Realizar predicciones precisas
world_05	subsection_01	quiz_01	question_01	¿Qué es el Procesamiento del Lenguaje Natural (NLP)?	Un campo de estudio que se ocupa de la interacción entre las computadoras y el lenguaje humano	Un enfoque de la inteligencia artificial para el análisis de imágenes	Un campo de estudio que se ocupa de la interacción entre las computadoras y el lenguaje humano	Un conjunto de técnicas utilizadas para el desarrollo de aplicaciones móviles	Un método para el análisis de grandes volúmenes de datos estructurados
			question_02	¿Cuál de las siguientes tareas NO está relacionada con el Procesamiento del Lenguaje Natural?	Detección de fraudes en transacciones financieras	Traducción automática de texto	Reconocimiento de voz	Análisis de sentimientos en redes sociales	Detección de fraudes en transacciones financieras
			question_03	¿Qué es la tokenización en el Procesamiento del Lenguaje Natural?	La división de un texto en unidades más pequeñas, como palabras o frases	El proceso de traducir texto en otro idioma	La tarea de etiquetar palabras en un texto con su categoría gramatical	La eliminación de palabras comunes sin significado en un texto	La división de un texto en unidades más pequeñas, como palabras o frases
			question_04	¿Qué es el análisis de sentimientos en el Procesamiento del Lenguaje Natural?	El proceso de evaluar el contenido emocional de un texto	La identificación de las emociones de una persona a través de su tono de voz	El proceso de evaluar el contenido emocional de un texto	La tarea de traducir un texto a diferentes idiomas	La detección de entidades nombradas en un texto
			question_05	¿Cuál de las siguientes técnicas se utiliza en el Procesamiento del Lenguaje Natural para comprender el significado de las palabras en un contexto determinado?	Word2Vec	Tokenización	Stemming	Lematización	Word2Voc
world_06	subsection_01	quiz_01	question_01	¿Qué es la Visión Computacional?	Un proceso que permite a las computadoras reconocer y comprender imágenes y videos	Un campo de estudio que analiza la estructura y función del ojo humano	Un proceso que permite a las computadoras reconocer y comprender imágenes y videos	Una técnica para el desarrollo de algoritmos de encriptación de datos	Un método para el diseño de interfaces gráficas de usuario
			question_02	¿Cuál de las siguientes tareas NO está relacionada con la Visión Computacional?	Detección de objetos en audio	Detección de objetos en imágenes	Seguimiento de objetos en videos	Reconocimiento facial	Extracción de características de audio
			question_03	¿Qué es la segmentación de imágenes en la Visión Computacional?	La identificación de objetos en una imagen y la asignación de una etiqueta a cada píxel correspondiente a ese objeto	La tarea de clasificar imágenes en diferentes categorías	El proceso de eliminar el ruido de una imagen	La identificación de objetos en una imagen y la asignación de una etiqueta a cada píxel correspondiente a ese objeto	La transformación de una imagen en un conjunto de características numéricas
			question_04	¿Qué es un descriptor de características en la Visión Computacional?	Un conjunto de atributos numéricos que describen una característica específica de una imagen	Un método para reducir la resolución de una imagen	Un algoritmo de compresión de imágenes	Un conjunto de atributos numéricos que describen una característica específica de una imagen	Un proceso para mejorar la calidad visual de una imagen
			question_05	¿Cuál de las siguientes técnicas se utiliza en la Visión Computacional para reconocer patrones en imágenes?	Redes neuronales convolucionales	Clustering	Regresión lineal	Redes neuronales convolucionales	Análisis de componentes principales (PCA)

Tabla 8.6: Valores de campo de la colección *questions*. Fuente: Autoría propia.

8.12 Anexo L:

colección "quices"			
Id del Mundo	Id de la sub-sección	Id del quiz	name
world_01	subsection_01	quiz_01	Quiz 1 Mundo 1
	subsection_02	quiz_01	Quiz 2 Mundo 1
world_02	subsection_01	quiz_01	Quiz 1 Mundo 2
	subsection_02	quiz_01	Quiz 2 Mundo 2
	subsection_03	quiz_01	Quiz 3 Mundo 2
	subsection_04	quiz_01	Quiz 4 Mundo 2
	subsection_05	quiz_01	Quiz 5 Mundo 2
world_03	subsection_01	quiz_01	Quiz 1 Mundo 3
	subsection_02	quiz_01	Quiz 2 Mundo 3
	subsection_03	quiz_01	Quiz 3 Mundo 3
	subsection_04	quiz_01	Quiz 4 Mundo 3
world_04	subsection_01	quiz_01	Quiz 1 Mundo 4
world_05	subsection_02	quiz_01	Quiz 1 Mundo 5
world_06	subsection_03	quiz_01	Quiz 1 Mundo 6

Tabla 8.7: Valores del campo *quiz_Id* de la colección quices. Fuente: Autoría propia.

8.13 Anexo M:

colección "videos"				
Id del Mundo	Id de la sub-sección	Id del video	name	video_url
world_01	subsection_01	video_01	Introducción a la inteligencia artificial	Lae9BefikUk
	subsection_02	video_01	Agentes Inteligentes	1wKkFY_JJFg
world_02	subsection_01	video_01	Introducción a la Lógica Proposicional	cL6c9SZwnMI
	subsection_02	video_01	Inferencia en Lógica Proposional	IYI15yf71dk
	subsection_03	video_01	Representación del Conocimiento en Lógica de Primer Orden	6CTrlR8EeqE
	subsection_04	video_01	Razonamiento en Lógica de Primer Orden	PYh9-CNvmHE
	subsection_05	video_01	Lógica de Descripción	gnqmN8qnIcY
world_03	subsection_01	video_01	Introducción a la búsqueda en Inteligencia Artificial	awsxi7odfTQ
	subsection_02	video_01	Búsqueda informada(heurística) y búsqueda no informada (ciega)	hrWc9K3TPdA
	subsection_03	video_01	Algoritmos de búsqueda	JUSquXxuu3Y
	subsection_04	video_01	Introducción a la Planificación	DKp8m2XGHQU
world_04	subsection_01	video_01	Introducción al Aprendizaje Automático	LBB1vg8Objc
world_05	subsection_02	video_01	Introducción al Procesamiento del Lenguaje Natural	Cn_D45heaWM
world_06	subsection_03	video_01	Introducción a la Visión Computacional	Fo_b8599B78
world_07	subsection_01	video_01	Instalación de Python	fsvP9sMSgm0
		video_02	Fundamentos de Python Teoría	yvX_F1Y45hs
		video_03	Fundamentos de Python Práctica	0ulv1Vno3Aw
	subsection_02	video_01	Control de Flujo	D3msIbArmuw
	subsection_03	video_01	Funciones	xI2_OeIetG0
	subsection_04	video_01	Listas	dUHfN3s6k6M

Tabla 8.8: Valores del campo *world_Id* de la colección videos. Fuente: Autoría propia.

8.14 Anexo N:

```
dependencies:
firebase_core: ^2.7.0
firebase_auth: ^4.2.2
cloud_firestore: ^4.4.3
firedart: ^0.9.2
firedart_cloud: ^0.9.5
```

8.15 Anexo O:

```
void main() async {
  if (!kIsWeb) {
    FirebaseAuth.initialize(apiKey, VolatileStore());
    Firestore.initialize(projectId);
  } else {
    await web_core.Firebase.initializeApp(
        options: web_core.FirebaseOptions(
      apiKey: 'apikey',
      appId: 'appId',
      messagingSenderId: 'messagingSenderId',
      projectId: 'projectId',
      authDomain: 'authDomain',
      storageBucket: 'storageBucket',
      measurementId: 'measurementId',
    ));
  }
  runApp(const MyApp());
}
```

8.16 Anexo P:

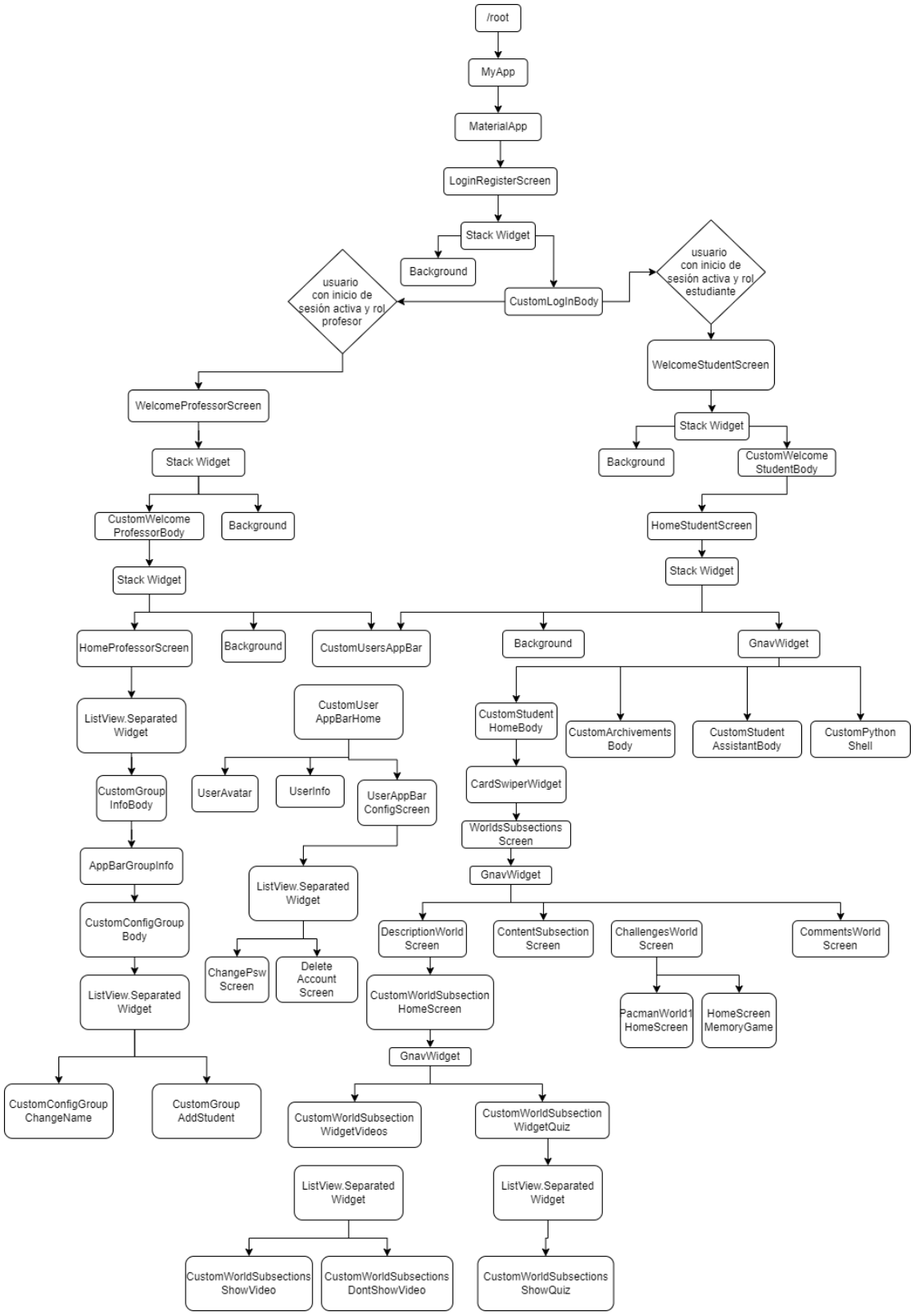

Figure 8.6: Árbol de *Widgets* del aplicativo. Fuente: Autoría propia.

8.17 Anexo Q:

```
import 'package:flutter/material.dart';
import 'dart:io';
import 'package:firedart/firedart.dart';
import 'package:logger/logger.dart';
import 'package:Openedu.IA/screens/screens.dart';
import 'package:toggle_switch/toggle_switch.dart';
import 'package:cloud_firestore/cloud_firestore.dart' as web_firestore;
import 'package:firebase_auth/firebase_auth.dart' as web_auth;
import 'package:flutter/foundation.dart' show kIsWeb;

class Register extends StatefulWidget {
  const Register({super.key});

  @override
  // ignore: library_private_types_in_public_api
  _RegisterState createState() => _RegisterState();
}

class _RegisterState extends State<Register> {
  _RegisterState();

  bool showProgress = false;
  bool visible = false;

  final _formkey = GlobalKey<FormState>();
  final TextEditingController passwordController = TextEditingController();
  final TextEditingController confirmpassController = TextEditingController();
  final TextEditingController firstnameController = TextEditingController();
  final TextEditingController lastnameController = TextEditingController();
  final TextEditingController emailController = TextEditingController();

  bool _isObscure = true;
  bool _isObscure2 = true;
  File? file;
  var rool = "teacher";

  void setLoginScreen(BuildContext context) {
    Navigator.pop(context,
        MaterialPageRoute(builder: (context) => const LoginRegisterScreen()));
  }

  @override
  Widget build(BuildContext context) {
    return SingleChildScrollView(
      child: Column(
        children: [
          Container(
            margin:
                const EdgeInsets.only(right: 5, left: 5, top: 10, bottom: 10),
            child: Form(
              key: _formkey,
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                crossAxisAlignment: CrossAxisAlignment.center,
```

```
children: [
  const Text(
    "  Ests  listo para",
    style: TextStyle(
      fontWeight: FontWeight.bold,
      color: Colors.black,
      fontSize: 25,
    ),
  ),
  const Text(
    "esta nueva",
    style: TextStyle(
      fontWeight: FontWeight.bold,
      color: Colors.black,
      fontSize: 25,
    ),
  ),
  const Text(
    "aventura?",
    style: TextStyle(
      fontWeight: FontWeight.bold,
      color: Colors.black,
      fontSize: 25,
    ),
  ),
  const SizedBox(
    height: 10,
  ),
  TextFormField(
    controller: firstnameController,
    decoration: InputDecoration(
      suffixIcon: Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          mainAxisSize: MainAxisSize.min,
          children: [
            if (!kIsWeb)
              if (Platform.isLinux ||
                  Platform.isMacOS ||
                  Platform.isWindows)
                IconButton(
                    onPressed: () =>
                        firstnameController.clear(),
                    icon: Icon(Icons.clear,
                        color: Colors.indigo)),
          ]),
      filled: true,
      fillColor: Colors.white,
      hintText: 'Nombres',
      hintStyle: const TextStyle(
          color: Colors.indigo, fontWeight: FontWeight.bold),
      enabled: true,
      contentPadding: const EdgeInsets.only(
          left: 14.0, bottom: 8.0, top: 8.0),
      focusedBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.indigo),
        borderRadius: BorderRadius.circular(20),
```

```
    ),
    enabledBorder: OutlineInputBorder(
      borderSide: const BorderSide(color: Colors.indigo),
      borderRadius: BorderRadius.circular(20),
    ),
  ),
),
const SizedBox(
  height: 20,
),
TextFormField(
  controller: lastnameController,
  decoration: InputDecoration(
    suffixIcon: Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        mainAxisSize: MainAxisSize.min,
        children: [
          if (!kIsWeb)
            if (Platform.isLinux ||
                Platform.isMacOS ||
                Platform.isWindows)
              IconButton(
                  onPressed: () => lastnameController.clear(),
                  icon: Icon(Icons.clear,
                      color: Colors.indigo)),
        ]),
    filled: true,
    fillColor: Colors.white,
    hintText: 'Apellidos ',
    hintStyle: const TextStyle(
        color: Colors.indigo, fontWeight: FontWeight.bold),
    enabled: true,
    contentPadding: const EdgeInsets.only(
        left: 14.0, bottom: 8.0, top: 8.0),
    focusedBorder: OutlineInputBorder(
      borderSide: const BorderSide(color: Colors.indigo),
      borderRadius: BorderRadius.circular(20),
    ),
    enabledBorder: OutlineInputBorder(
      borderSide: const BorderSide(color: Colors.indigo),
      borderRadius: BorderRadius.circular(20),
    ),
  ),
),
const SizedBox(
  height: 20,
),
TextFormField(
  controller: emailController,
  decoration: InputDecoration(
    suffixIcon: Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        mainAxisSize: MainAxisSize.min,
        children: [
          if (!kIsWeb)
            if (Platform.isLinux ||
```

```
                    Platform.isMacOS ||
                    Platform.isWindows)
                  IconButton(
                      onPressed: () => emailController.clear(),
                      icon: Icon(Icons.clear,
                          color: Colors.indigo)),
            ]),
        filled: true,
        fillColor: Colors.white,
        hintText: 'Correo␣Electr nico',
        hintStyle: const TextStyle(
            color: Colors.indigo, fontWeight: FontWeight.bold),
        enabled: true,
        contentPadding: const EdgeInsets.only(
            left: 14.0, bottom: 8.0, top: 8.0),
        focusedBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.indigo),
          borderRadius: BorderRadius.circular(20),
        ),
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.indigo),
          borderRadius: BorderRadius.circular(20),
        ),
      ),
      validator: (value) {
        if (value!.isEmpty) {
          return "El␣Correo␣electr nico␣no␣puede␣ser␣nulo.";
        }
        if (!RegExp("^[a-zA-Z0-9+_.-]+@[a-zA-Z0-9.-]+.[a-z]")
            .hasMatch(value)) {
          return ("Por␣favor␣ingresa␣un␣correo␣electr nico␣v lido");
        } else {
          return null;
        }
      },
      onChanged: (value) {},
      keyboardType: TextInputType.emailAddress,
    ),
    const SizedBox(
      height: 20,
    ),
    TextFormField(
      obscureText: _isObscure,
      controller: passwordController,
      decoration: InputDecoration(
        suffixIcon: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            mainAxisSize: MainAxisSize.min,
            children: [
              if (!kIsWeb)
                if (Platform.isLinux ||
                    Platform.isMacOS ||
                    Platform.isWindows)
                  IconButton(
                      onPressed: () => passwordController.clear(),
                      icon: Icon(Icons.clear,
```

```
                        color: Colors.indigo)),
              IconButton(
                  color: Colors.indigo,
                  icon: Icon(_isObscure
                      ? Icons.visibility_off
                      : Icons.visibility),
                  onPressed: () {
                    setState(() {
                      _isObscure = !_isObscure;
                    });
                  }),
            ]),
        filled: true,
        fillColor: Colors.white,
        hintText: 'Contrase a',
        hintStyle: const TextStyle(
            color: Colors.indigo, fontWeight: FontWeight.bold),
        enabled: true,
        contentPadding: const EdgeInsets.only(
            left: 14.0, bottom: 8.0, top: 15.0),
        focusedBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(20),
        ),
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.indigo),
          borderRadius: BorderRadius.circular(20),
        ),
      ),
      validator: (value) {
        RegExp regex = RegExp(r'^.{6,}$');
        if (value!.isEmpty) {
          return "La contrase a no puede ser vacia";
        }
        if (!regex.hasMatch(value)) {
          return ("Por favor solicita una contrase a valida min 6 caracteres")
        } else {
          return null;
        }
      },
      onChanged: (value) {},
    ),
    const SizedBox(
      height: 20,
    ),
    TextFormField(
      obscureText: _isObscure2,
      controller: confirmpassController,
      decoration: InputDecoration(
        suffixIcon: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            mainAxisSize: MainAxisSize.min,
            children: [
              if (!kIsWeb)
                if (Platform.isLinux ||
                    Platform.isMacOS ||
```

```
                  Platform.isWindows)
                IconButton(
                    onPressed: () =>
                        confirmpassController.clear(),
                    icon: Icon(Icons.clear,
                        color: Colors.indigo)),
            IconButton(
                color: Colors.indigo,
                icon: Icon(_isObscure2
                    ? Icons.visibility_off
                    : Icons.visibility),
                onPressed: () {
                  setState(() {
                    _isObscure2 = !_isObscure2;
                  });
                }),
          ]),
      filled: true,
      fillColor: Colors.white,
      hintText: 'Confirma tu Contrase a',
      hintStyle: const TextStyle(
          color: Colors.indigo, fontWeight: FontWeight.bold),
      enabled: true,
      contentPadding: const EdgeInsets.only(
          left: 14.0, bottom: 8.0, top: 15.0),
      focusedBorder: OutlineInputBorder(
        borderSide:
            const BorderSide(color: Colors.indigo, width: 3),
        borderRadius: BorderRadius.circular(20),
      ),
      enabledBorder: OutlineInputBorder(
        borderSide:
            const BorderSide(color: Colors.indigo, width: 1),
        borderRadius: BorderRadius.circular(20),
      ),
    ),
    validator: (value) {
      if (confirmpassController.text !=
          passwordController.text) {
        return "Las contrase as no son iguales !";
      } else {
        return null;
      }
    },
    onChanged: (value) {},
  ),
  const SizedBox(
    height: 20,
  ),
  ToggleSwitch(
    minWidth: 150.0,
    initialLabelIndex: 1,
    cornerRadius: 20.0,
    activeFgColor: Colors.white,
    inactiveBgColor: Colors.grey,
    inactiveFgColor: Colors.white,
```

```
  totalSwitches: 2,
  labels: const ['Estudiante', 'Profesor'],
  icons: const [Icons.person, Icons.book],
  activeBgColors: const [
    [Colors.indigo],
    [Colors.indigo]
  ],
  onToggle: (index) {
    if (index == 0) {
      rool = "student";
    } else {
      rool = "teacher";
    }
  },
),
const SizedBox(
  height: 20,
),
MaterialButton(
  shape: const RoundedRectangleBorder(
      borderRadius: BorderRadius.all(Radius.circular(20.0))),
  elevation: 5.0,
  height: 50,
  onPressed: () {
    setState(() {
      showProgress = true;
    });
    if ((confirmpassController.text ==
            passwordController.text) &&
        (lastnameController.text.isNotEmpty) &&
        (firstnameController.text.isNotEmpty) &&
        (emailController.text.isNotEmpty)) {
      if (confirmpassController.text.isNotEmpty &&
          passwordController.text.isNotEmpty) {
        signUp(emailController.text, passwordController.text,
            rool);
      }
    } else {
      if (confirmpassController.text !=
          passwordController.text)
        showInSnackBar("Contrase a no coincide", context);
      if (confirmpassController.text.isEmpty ||
          passwordController.text.isNotEmpty)
        showInSnackBar(
            "La contrase a no puede estar vacia", context);
      if (lastnameController.text.isEmpty ||
          firstnameController.text.isEmpty)
        showInSnackBar(
            'Los nombres no pueden estar vacios', context);
      if (emailController.text.isEmpty)
        showInSnackBar(
            'El correo electronico no puede estar vacio',
            context);
    }
  },
  color: Colors.indigo,
```

```
                    child: const Text(
                      "Registrate ",
                      style: TextStyle(fontSize: 20, color: Colors.white),
                    ),
                  ),
                  const SizedBox(
                    height: 10,
                  ),
                  Row(
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                      const Text(
                        ' Ya  est s registrado?',
                        style: TextStyle(
                            color: Colors.black,
                            fontSize: 16,
                            fontWeight: FontWeight.bold),
                      ),
                      const SizedBox(width: 4),
                      GestureDetector(
                        onTap: () => setLoginScreen(context),
                        child: const Text(
                          ' Ingresa ahora !',
                          style: TextStyle(
                            color: Colors.indigo,
                            fontSize: 18,
                            fontWeight: FontWeight.bold,
                          ),
                        ),
                      ),
                      const SizedBox(
                        height: 20,
                      )
                    ],
                  ),
                ],
              ),
            ),
          ),
        ],
      ),
    );
}

void signUp(String email, String password, String rool) async {
  var logger = Logger();
  if (kIsWeb) {
    CircularProgressIndicator();
    final _auth = web_auth.FirebaseAuth.instance;
    await _auth
        .createUserWithEmailAndPassword(email: email, password: password)
        .then(
          (value) => showInSnackBar('Cuenta creada!', context),
        )
        .catchError((e) {
      showInSnackBar('Error creando cuenta ${e}!', context);
```

```
    });
    try {
      var user = _auth.currentUser;
      web_firestore.CollectionReference ref =
          web_firestore.FirebaseFirestore.instance.collection('users');
      if (rool == "student") {
        ref.doc(user!.uid).set({
          'email': emailController.text,
          'rool': rool,
          'first_name': firstnameController.text,
          'last_name': lastnameController.text,
          'icon': 'user_00',
          'world_progress': '1',
          'subsection_progress': '0',
          'challenges_count': '0',
        });
      } else {
        ref.doc(user!.uid).set({
          'email': emailController.text,
          'rool': rool,
          'first_name': firstnameController.text,
          'last_name': lastnameController.text,
          'icon': 'user_00',
          'groups_count': '0',
        });
      }
      Navigator.pop(
          context,
          MaterialPageRoute(
              builder: (context) => const LoginRegisterScreen()));
    } catch (e) {
      logger.e(e.toString());
      // ignore: use_build_context_synchronously
      Navigator.pop(context);
      wrongRegisterMessage(e.toString());
    }
  } else {
    FirebaseAuth firebaseAuth = FirebaseAuth.instance;

    // ignore: use_build_context_synchronously
    showDialog(
      context: context,
      builder: (context) {
        return const Center(
          child: CircularProgressIndicator(),
        );
      },
    );
    try {
      await firebaseAuth.signUp(email, password).then((value) => {
            showInSnackBar('Cuenta creada!', context),
          });
      var user = await firebaseAuth.getUser();
      CollectionReference ref = Firestore.instance.collection('users');
      if (rool == "student") {
        ref.document(user.id).set({
```

```
            'email': emailController.text,
            'rool': rool,
            'first_name': firstnameController.text,
            'last_name': lastnameController.text,
            'icon': 'user_00',
            'world_progress': '1',
            'subsection_progress': '0',
            'challenges_count': '0',
          });
        } else {
          ref.document(user.id).set({
            'email': emailController.text,
            'rool': rool,
            'first_name': firstnameController.text,
            'last_name': lastnameController.text,
            'icon': 'user_00',
            'groups_count': '0',
          });
        }
        // ignore: use_build_context_synchronously
        Navigator.pop(
            context,
            MaterialPageRoute(
                builder: (context) => const LoginRegisterScreen()));
      } catch (e) {
        logger.e(e.toString());
        // ignore: use_build_context_synchronously
        Navigator.pop(context);
        wrongRegisterMessage(e.toString());
        // ignore: use_build_context_synchronously
      }
    }
  }

  void wrongRegisterMessage(String error) {
    showDialog(
      context: context,
      builder: (context) {
        return AlertDialog(
          backgroundColor: Colors.indigo,
          title: Center(
            child: Text(
              error,
              style: const TextStyle(color: Colors.white),
            ),
          ),
        );
      },
    );
  }

  void showInSnackBar(String value, BuildContext context) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(value)));
  }
}
```

8.18 Anexo R:

```
class CustomLogInBody extends StatefulWidget {
  const CustomLogInBody({Key? key}) : super(key: key);

  @override
  State<CustomLogInBody> createState() => _CustomLogInBodyState();
}

class _CustomLogInBodyState extends State<CustomLogInBody> {
  bool _isObscure3 = true;
  bool visible = false;
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();
  void setRegisterScreen(BuildContext context) {
    Navigator.push(context,
        MaterialPageRoute(builder: (context) => const RegisterScreen()));
  }

  void setForgotPasswordScreen(BuildContext context) {
    Navigator.push(context,
        MaterialPageRoute(builder: (context) => const ForgotPasswordScreen()));
  }

  @override
  Widget build(BuildContext context) {
    return SingleChildScrollView(
      child: Container(
        margin: const EdgeInsets.only(top: 5, right: 15, left: 15),
        child: Center(
          child: Column(mainAxisAlignment: MainAxisAlignment.center, children: [
            const SizedBox(
              height: 50,
            ),
            const Text('Bienvenido␣de␣nuevo,␣',
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 25,
                    fontWeight: FontWeight.bold)),
            const SizedBox(height: 10),
            const Text('te␣hemos␣extra ado!␣',
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 25,
                    fontWeight: FontWeight.bold)),
            const SizedBox(height: 30),
            TextFormField(
              controller: emailController,
              decoration: InputDecoration(
                suffixIcon: Row(
                    mainAxisAlignment: MainAxisAlignment.spaceBetween,
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      if (!kIsWeb)
                        if (Platform.isLinux ||
                            Platform.isMacOS ||
```

```
                Platform.isWindows ||
                Platform.isFuchsia)
              IconButton(
                  onPressed: () => emailController.clear(),
                  icon: Icon(Icons.clear, color: Colors.indigo)),
          ]),
      filled: true,
      fillColor: Colors.white,
      hintText: 'Correo␣Electr nico',
      hintStyle: TextStyle(
          color: Colors.indigo, fontWeight: FontWeight.bold),
      enabled: true,
      contentPadding:
          const EdgeInsets.only(left: 14.0, bottom: 8.0, top: 8.0),
      focusedBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.indigo),
        borderRadius: BorderRadius.circular(10),
      ),
      enabledBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.indigo),
        borderRadius: BorderRadius.circular(10),
      ),
    ),
    validator: (value) {
      if (value!.isEmpty) {
        return "El␣correo␣electr nico␣no␣puede␣estar␣vacio.";
      }
      if (!RegExp("^[a-zA-Z0-9+_.-]+@[a-zA-Z0-9.-]+.[a-z]")
          .hasMatch(value)) {
        return ("Por␣favor␣solicita␣un␣correo␣electr nico␣v lido.");
      } else {
        return null;
      }
    },
    onSaved: (value) {
      emailController.text = value!;
    },
    keyboardType: TextInputType.emailAddress,
  ),
  const SizedBox(
    height: 10,
  ),
  TextFormField(
    controller: passwordController,
    obscureText: _isObscure3,
    decoration: InputDecoration(
      hintStyle: TextStyle(
          color: Colors.indigo, fontWeight: FontWeight.bold),
      suffixIcon: Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          mainAxisSize: MainAxisSize.min,
          children: [
            if (!kIsWeb)
              if (Platform.isLinux ||
                  Platform.isMacOS ||
                  Platform.isWindows ||
```

```
                  Platform.isFuchsia)
                IconButton(
                    onPressed: () => passwordController.clear(),
                    icon: Icon(Icons.clear, color: Colors.indigo)),
            IconButton(
                color: Colors.indigo,
                icon: Icon(_isObscure3
                    ? Icons.visibility
                    : Icons.visibility_off),
                onPressed: () {
                  setState(() {
                    _isObscure3 = !_isObscure3;
                  });
                }),
          ]),
      filled: true,
      fillColor: Colors.white,
      hintText: 'Contrase a',
      enabled: true,
      contentPadding:
          const EdgeInsets.only(left: 14.0, bottom: 8.0, top: 15.0),
      focusedBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.indigo),
        borderRadius: BorderRadius.circular(10),
      ),
      enabledBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.indigo),
        borderRadius: BorderRadius.circular(10),
      ),
    ),
    validator: (value) {
      RegExp regex = RegExp(r'^.{6,}$');
      if (value!.isEmpty) {
        return "La contrase a no puede ser vac a";
      }
      if (!regex.hasMatch(value)) {
        return "Por favor solicita una contrase a v lida Min. 6 car cteres";
      } else {
        return null;
      }
    },
    onSaved: (value) {
      passwordController.text = value!;
    },
    textCapitalization: TextCapitalization.none,
  ),
  const SizedBox(
    height: 10,
  ),
  const SizedBox(height: 35),
  CustomButton(onTap: () {
    setState(() {
      visible = true;
    });
    signIn();
  }),
```

```
          const SizedBox(height: 135),
          Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              const Text(
                " No est s registrado?",
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 14,
                    fontWeight: FontWeight.bold),
              ),
              const SizedBox(width: 4),
              GestureDetector(
                onTap: () => setRegisterScreen(context),
                child: const Text(
                  'Registrate ahora!',
                  style: TextStyle(
                    color: Colors.indigo,
                    fontSize: 18,
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ),
            ],
          ),
          const SizedBox(
            height: 10,
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              const Text(
                '  Olvidaste  tu contrase a?',
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 14,
                    fontWeight: FontWeight.bold),
              ),
              const SizedBox(width: 4),
              GestureDetector(
                onTap: () => setForgotPasswordScreen(context),
                child: const Text(
                  'Toca aqu ',
                  style: TextStyle(
                    color: Colors.indigo,
                    fontSize: 18,
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ),
            ],
          ),
        ]),
      ),
    ),
  );
```

```
}

Future signIn() async {
  var logger = Logger();
  if (kIsWeb) {
    final _auth = web_auth.FirebaseAuth.instance;
    showDialog(
      context: context,
      builder: (context) {
        return const Center(
          child: CircularProgressIndicator(),
        );
      },
    );
    try {
      await _auth.signInWithEmailAndPassword(
          email: emailController.text, password: passwordController.text);
      var user = _auth.currentUser;
      web_firestore.CollectionReference ref =
          web_firestore.FirebaseFirestore.instance.collection('users');
      ref
          .doc(user!.uid)
          .get()
          .then((web_firestore.DocumentSnapshot documentSnapshot) {
        if (documentSnapshot.get('rool') == "teacher") {
          setState(() {
            Navigator.of(context).pushReplacement(MaterialPageRoute(
                builder: (context) => const WelcomeProfessorScreen()));
          });
        } else {
          setState(() {
            Navigator.of(context).pushReplacement(MaterialPageRoute(
                builder: (context) => const WelcomeStudentScreen()));
          });
        }
      });
    } catch (e) {
      logger.e(e.toString());
      Navigator.pop(context);
      showDialog(
        context: context,
        builder: (context) {
          return AlertDialog(
            backgroundColor: Colors.indigo,
            title: Center(
              child: Text(
                e.toString(),
                style: const TextStyle(color: Colors.white),
              ),
            ),
          );
        },
      );
    }
  } else {
    final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
```

```
    showDialog(
      context: context,
      builder: (context) {
        return const Center(
          child: CircularProgressIndicator(),
        );
      },
    );
    try {
      await firebaseAuth.signIn(
          emailController.text, passwordController.text);
      var user = await firebaseAuth.getUser();
      var ref = Firestore.instance.collection('users').document(user.id);
      if (await ref.exists) {
        var query = await ref.get();
        if (query['rool'] == "teacher") {
          setState(() {
            Navigator.of(context).pushReplacement(MaterialPageRoute(
                builder: (context) => const WelcomeProfessorScreen()));
          });
        } else {
          setState(() {
            Navigator.of(context).pushReplacement(MaterialPageRoute(
                builder: (context) => const WelcomeStudentScreen()));
          });
        }
      }
    } catch (e) {
      logger.e(e.toString());
      Navigator.pop(context);
      showDialog(
        context: context,
        builder: (context) {
          return AlertDialog(
            backgroundColor: Colors.indigo,
            title: Center(
              child: Text(
                e.toString(),
                style: const TextStyle(color: Colors.white),
              ),
            ),
          );
        },
      );
    }
  }
}

void wrongdBMessage() {
  showDialog(
    context: context,
    builder: (context) {
      return const AlertDialog(
        backgroundColor: Colors.indigo,
        title: Center(
```

```
            child: Text(
              'No se encuentra la base de datos',
              style: TextStyle(color: Colors.white),
            ),
          ),
        );
      },
    );
  }

  // wrong email message popup
  void wrongEmailMessage() {
    showDialog(
      context: context,
      builder: (context) {
        return const AlertDialog(
          backgroundColor: Colors.indigo,
          title: Center(
            child: Text(
              'Email Incorrecto',
              style: TextStyle(color: Colors.white),
            ),
          ),
        );
      },
    );
  }

  // wrong password message popup
  void wrongPasswordMessage() {
    showDialog(
      context: context,
      builder: (context) {
        return const AlertDialog(
          backgroundColor: Colors.indigo,
          title: Center(
            child: Text(
              'Contrase a Incorrecta',
              style: TextStyle(color: Colors.white),
            ),
          ),
        );
      },
    );
  }
}
```

8.19 Anexo S:

```
import 'package:flutter/material.dart';
import 'package:Openedu.IA/widgets/widgets.dart';
import 'package:Openedu.IA/screens/screens.dart';
import 'package:firebase_auth/firebase_auth.dart' as web_auth;
import 'package:flutter/foundation.dart' show kIsWeb;
import 'package:firedart/firedart.dart';
import 'package:quickalert/quickalert.dart';

Future resetPassword(BuildContext context, String email) async {
  if (kIsWeb) {
    try {
      final _auth = web_auth.FirebaseAuth.instance;
      await _auth.sendPasswordResetEmail(email: email);
      await Future.delayed(Duration(seconds: 1));
      Navigator.of(context).pushAndRemoveUntil(
          MaterialPageRoute(builder: (context) => const LoginRegisterScreen()),
          (route) => false);
      showInSnackBar(
          'Revisa tu correo para reestablecer tu contrase a', context);
    } catch (e) {
      showInSnackBar('Error: ${e}', context);
    }
  } else {
    try {
      FirebaseAuth firebaseAuth = FirebaseAuth.instance;
      await firebaseAuth.resetPassword(email);
      await Future.delayed(Duration(seconds: 1));
      Navigator.of(context).pushAndRemoveUntil(
          MaterialPageRoute(builder: (context) => const LoginRegisterScreen()),
          (route) => false);
      showInSnackBar(
          'Revisa tu correo para reestablecer tu contrase a', context);
    } catch (e) {
      showInSnackBar('Error: ${e}', context);
    }
  }
}

class ForgotPasswordScreen extends StatefulWidget {
  const ForgotPasswordScreen({Key? key}) : super(key: key);

  @override
  State<ForgotPasswordScreen> createState() => _ForgotPasswordScreenState();
}

class \_ForgotPasswordScreenState extends State<ForgotPasswordScreen> {
  void setLogInScreen(BuildContext context) {
    Navigator.push(context,
        MaterialPageRoute(builder: (context) => const LoginRegisterScreen()));
  }

  @override
  Widget build(BuildContext context) {
    final TextEditingController emailController = TextEditingController();
```

```
    return Scaffold(
        appBar: AppBar(
            automaticallyImplyLeading: false,
            title: const Center(
              child: Text(
                'OpenEduIA',
                style: TextStyle(fontWeight: FontWeight.bold, fontSize: 25),
              ),
            ),
            actions: [
              IconButton(
                  onPressed: () {
                    customshowDialog(context);
                  },
                  icon: const Icon(Icons.info))
            ]),
        body: SingleChildScrollView(
            child: Container(
          margin: const EdgeInsets.only(top: 5, right: 15, left: 15),
          child: Center(
              child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text(
                "Por favor ingresa tu correo electr nico para enviarte las
 instrucciones a seguir para reestablecer tu contrase a.",

                textAlign: TextAlign.justify,
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 20,
                    fontWeight: FontWeight.bold),
              ),
              SizedBox(
                height: 15,
              ),
              TextFormField(
                controller: emailController,
                decoration: InputDecoration(
                  filled: true,
                  fillColor: Colors.white,
                  hintText: 'Correo Electr nico',
                  hintStyle: const TextStyle(
                      color: Colors.indigo, fontWeight: FontWeight.bold),
                  enabled: true,
                  contentPadding:
                      const EdgeInsets.only(left: 14.0, bottom: 8.0, top: 8.0),
                  focusedBorder: OutlineInputBorder(
                    borderSide: const BorderSide(color: Colors.indigo),
                    borderRadius: BorderRadius.circular(20),
                  ),
                  enabledBorder: OutlineInputBorder(
                    borderSide: const BorderSide(color: Colors.indigo),
                    borderRadius: BorderRadius.circular(20),
                  ),
                ),
```

```
                onChanged: (value) {},
                keyboardType: TextInputType.emailAddress,
              ),
              SizedBox(
                height: 15,
              ),
              MaterialButton(
                shape: const RoundedRectangleBorder(
                    borderRadius: BorderRadius.all(Radius.circular(20.0))),
                elevation: 5.0,
                color: Colors.indigo,
                height: 50,
                onPressed: () async {
                  if (emailController.text.isNotEmpty &&
                      emailController.text.contains('@')) {
                    QuickAlert.show(
                      context: context,
                      type: QuickAlertType.warning,
                      title: "  Ests  ␣Seguro?",
                      text:
                          "Este␣proceso␣envia␣un␣link␣de␣correo
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣electr nico␣para␣recuperar␣tu␣cuenta.",

                      confirmBtnText: "Estoy␣seguro",
                      onConfirmBtnTap: () async {
                        resetPassword(context, emailController.text);
                      },
                    );
                  } else {
                    await Future.delayed(Duration(seconds: 1));
                    showInSnackBar(
                        "El␣correo␣electr nico␣es␣invalido", context);
                  }
                },
                child: Text(
                  "Reestablecer␣Contrase a",
                  style: TextStyle(fontSize: 20, color: Colors.white),
                ),
              ),
              SizedBox(height: 20),
              Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  const Text(
                    '  Quieres  ␣volver?',
                    style: TextStyle(
                        color: Colors.black,
                        fontSize: 14,
                        fontWeight: FontWeight.bold),
                  ),
                  const SizedBox(width: 4),
                  GestureDetector(
                    onTap: () => setLogInScreen(context),
                    child: const Text(
                      'Toca␣Aqu ',
                      style: TextStyle(
```

```
                        color: Colors.indigo,
                        fontSize: 18,
                        fontWeight: FontWeight.bold,
                      ),
                    ),
                  ),
                ],
              ),
            ],
          )),
        )));
  }
}
```

8.20 Anexo T:

```
import 'package:flutter/material.dart';

import 'package:Openedu.IA/widgets/widgets.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:flutter/services.dart';

class HomeStudentScreen extends StatefulWidget {
  @override
  State<HomeStudentScreen> createState() => _HomeStudentScreenState();
}

class _HomeStudentScreenState extends State<HomeStudentScreen> {
  int _currentIndex = 0;

  final List<Widget> _children = [
    CustomStudentHomeBody(),
    CustomStudentArchivementsBody(),
    CustomStudentAssistantBody(),
    CustomStudentPythonShell(),
  ];

  @override
  Widget build(BuildContext context) {
    SystemChrome.setPreferredOrientations(
        [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);
    final size = MediaQuery.of(context).size;
    return WillPopScope(
        onWillPop: () async => false,
        child: Scaffold(
            body: SizedBox(
              width: size.width,
              height: size.height,
              child: Stack(
                children: [
                  const CustomStudentBackground(),
                  if (_currentIndex == 0)
                    CustomStudentHomeBody()
                  else
                    _children[_currentIndex],
                ],
              ),
            ),
            bottomNavigationBar: Container(
              color: Colors.indigo,
              child: Padding(
                padding:
                    const EdgeInsets.symmetric(horizontal: 15.0, vertical: 20),
                child: GNav(
                  backgroundColor: Colors.transparent,
                  color: Colors.transparent,
                  activeColor: Colors.white,
                  tabBackgroundColor: Colors.purpleAccent.shade100,
                  gap: 8,
                  onTabChange: (index) {
```

```
                    setState(() {
                      _currentIndex = index;
                    });
                  },
                  padding: const EdgeInsets.all(5),
                  tabs: const [
                    GButton(
                      icon: Icons.home,
                      iconColor: Colors.white,
                      iconActiveColor: Colors.white,
                      iconSize: 30,
                      text: 'Inicio',
                      textColor: Colors.white,
                    ),
                    GButton(
                      icon: Icons.workspace_premium_outlined,
                      iconColor: Colors.white,
                      iconActiveColor: Colors.white,
                      iconSize: 30,
                      text: 'Logros',
                      textColor: Colors.white,
                    ),
                    GButton(
                      icon: Icons.support_agent_outlined,
                      iconColor: Colors.white,
                      iconActiveColor: Colors.white,
                      iconSize: 30,
                      text: 'Asistencia virtual',
                      textColor: Colors.white,
                    ),
                    GButton(
                      icon: Icons.terminal,
                      iconColor: Colors.white,
                      iconActiveColor: Colors.white,
                      iconSize: 30,
                      text: 'Compilador Python',
                      textColor: Colors.white,
                    )
                  ],
                ),
              ),
            )));
  }
}

class CustomStudentHomeBody extends StatelessWidget {
  final int selected_avatar = 0;
  CustomStudentHomeBody({Key? key, selected_avatar}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;
    return Container(
      alignment: Alignment.center,
      color: Colors.transparent,
      child: Column(
```

```
          mainAxisAlignment: MainAxisAlignment.start,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            CustomStudentAppBarHome(),
            Column(
              mainAxisAlignment: MainAxisAlignment.start,
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                SizedBox(
                  height: 20,
                ),
                SizedBox(
                  height: size.height - 235,
                  width: size.width - 50,
                  child: WorldsCardSwiper(),
                ),
              ],
            )
          ],
        ),
      );
  }
}
List<String> getWorldsImage() {
  List<String> worldsImages = [];

  for (int i = 1; i <= globals.numberWorlds; i++) {
    worldsImages.add('assets/worlds/worlds_cards_icons/world_0${i}.jpg');
  }
  return worldsImages;
}

Future getWorldsName() async {
  List<String> worldsNames = [];
  String world_progress = "1";
  if (kIsWeb) {
    final _auth = web_auth.FirebaseAuth.instance;
    var user = await _auth.currentUser;
    web_firestore.CollectionReference user_ref =
        web_firestore.FirebaseFirestore.instance.collection('users');
    await user_ref
        .doc(user!.uid)
        .get()
        .then((web_firestore.DocumentSnapshot documentSnapshot) {
      world_progress = documentSnapshot['world_progress'].toString();
    });
    for (int i = 1; i <= globals.numberWorlds; i++) {
      try {
        web_firestore.CollectionReference ref =
            web_firestore.FirebaseFirestore.instance.collection('worlds');
        await ref
            .doc('world_0${i}')
            .get()
            .then((web_firestore.DocumentSnapshot documentSnapshot) {
          worldsNames.add(documentSnapshot['name'].toString());
        });
```

```
      } catch (e) {
        worldsNames.add('Error!');
        world_progress = "1";
      }
    }
    return [worldsNames, world_progress];
  } else {
    final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
    var user = await firebaseAuth.getUser();
    var user_ref = Firestore.instance.collection('users').document(user.id);
    var user_query = await user_ref.get();
    world_progress = user_query['world_progress'];
    for (int i = 1; i <= globals.numberWorlds; i++) {
      try {
        var ref =
            Firestore.instance.collection('worlds').document('world_0${i}');
        var query = await ref.get();
        worldsNames.add(query['name']);
      } catch (e) {
        worldsNames.add('Error!');
        world_progress = "1";
      }
    }
    return [worldsNames, world_progress];
  }
}

class WorldsCardSwiper extends StatefulWidget {
  const WorldsCardSwiper({Key? key}) : super(key: key);

  @override
  State<WorldsCardSwiper> createState() => _WorldsCardSwiperState();
}

class _WorldsCardSwiperState extends State<WorldsCardSwiper> {
  @override
  Widget build(BuildContext context) {
    double width = MediaQuery.of(context).size.width;
    double height = MediaQuery.of(context).size.height;
    List<String> worldsImages = getWorldsImage();
    return FutureBuilder(
        future: getWorldsName(),
        builder: (context, AsyncSnapshot snapshot) {
          if (snapshot.hasData) {
            return SizedBox(
              width: width,
              height: height,
              child: Swiper(
                itemCount: globals.numberWorlds,
                itemBuilder: (context, index) {
                  return SingleChildScrollView(
                    child: Column(
                      children: [
                        Text(
                          snapshot.data[0][index].toString(),
                          textAlign: TextAlign.justify,
```

```
  style: TextStyle(
      color: Colors.indigo,
      fontWeight: FontWeight.bold,
      fontSize: 20),
),
SizedBox(height: 15),
if (kIsWeb)
  GestureDetector(
    onTap: () {
      setState(() {});
      if (index + 1 <= int.parse(snapshot.data[1]) ||
          index == 6)
        Navigator.of(context).push(MaterialPageRoute(
            builder: (context) =>
                WorldsSubsectionsScreen(
                  world_index: index,
                  world_name: snapshot.data[0][index]
                      .toString(),
                )));
      if (index + 1 > int.parse(snapshot.data[1]) &&
          index != 6)
        QuickAlert.show(
            context: context,
            type: QuickAlertType.info,
            title: "Informaci n",
            titleColor: Colors.indigo,
            textColor: Colors.black,
            text:
                'Tienes que completar el mundo anterior',
            confirmBtnText: 'Aceptar');
    },
    child: Container(
      width: width,
      height: 280,
      child: FadeInImage(
        image:
            AssetImage(worldsImages[index].toString()),
        placeholder: const AssetImage(
          'assets/icon/loading.gif',
        ),
        fit: BoxFit.fitHeight,
      ),
    ),
  ),
if (!kIsWeb)
  if (Platform.isAndroid || Platform.isIOS)
    GestureDetector(
      onTap: () {
        setState(() {});
        if (index + 1 <= int.parse(snapshot.data[1]) ||
            index == 6)
          Navigator.of(context).push(MaterialPageRoute(
              builder: (context) =>
                  WorldsSubsectionsScreen(
                    world_index: index,
                    world_name: snapshot.data[0][index]
```

```
                                .toString(),
                          )));
                  if (index + 1 > int.parse(snapshot.data[1]) &&
                      index != 6)
                    QuickAlert.show(
                        context: context,
                        type: QuickAlertType.info,
                        title: "Informaci n",
                        titleColor: Colors.indigo,
                        textColor: Colors.black,
                        text:
                            'Tienes que completar el mundo anterior',
                        confirmBtnText: 'Aceptar');
                },
                child: Container(
                  width: width,
                  height: 300,
                  child: FadeInImage(
                    image: AssetImage(
                        worldsImages[index].toString()),
                    placeholder: const AssetImage(
                      'assets/icon/loading.gif',
                    ),
                    fit: BoxFit.fitWidth,
                  ),
                ),
              ),
            if (!kIsWeb)
              if (Platform.isFuchsia ||
                  Platform.isMacOS ||
                  Platform.isWindows ||
                  Platform.isLinux)
                GestureDetector(
                  onTap: () {
                    setState(() {});
                    if (index + 1 <= int.parse(snapshot.data[1]) ||
                        index == 6)
                      Navigator.of(context).push(MaterialPageRoute(
                          builder: (context) =>
                              WorldsSubsectionsScreen(
                                world_index: index,
                                world_name: snapshot.data[0][index]
                                    .toString(),
                              )));
                    if (index + 1 > int.parse(snapshot.data[1]) &&
                        index != 6)
                      QuickAlert.show(
                          context: context,
                          type: QuickAlertType.info,
                          title: "Informaci n",
                          titleColor: Colors.indigo,
                          textColor: Colors.black,
                          text:
                              'Tienes que completar el mundo anterior',
                          confirmBtnText: 'Aceptar');
                  },
```

```
                                child: Container(
                                  width: width,
                                  height: 300,
                                  child: FadeInImage(
                                    image: AssetImage(
                                        worldsImages[index].toString()),
                                    placeholder: const AssetImage(
                                      'assets/icon/loading.gif',
                                    ),
                                    fit: BoxFit.fitHeight,
                                  ),
                                ),
                              ),
                            ],
                          ),
                        );
                      },
                      indicatorLayout: PageIndicatorLayout.COLOR,
                      autoplay: false,
                      pagination: const SwiperPagination(),
                      control: const SwiperControl(),
                    ),
                  );
                } else {
                  return Center(
                      child: Text(
                    'Cargando ...',
                    style: TextStyle(
                        color: Colors.indigo,
                        fontSize: 30,
                        fontWeight: FontWeight.bold),
                  ));
                }
              });
        }
    }
```

8.21 Anexo U:

```
class CustomStudentArchivementsBody extends StatelessWidget {
  const CustomStudentArchivementsBody({Key? key}) : super(key: key);
  Future getChallengesInfo(BuildContext context) async {
    String challenge_count = "0";
    int _rewards_count = 0;
    String rewards_count = "0";
    String world_progress = "";
    String subsections_progress = "";

    if (kIsWeb) {
      try {
        final _auth = web_auth.FirebaseAuth.instance;
        var user = await _auth.currentUser;

        web_firestore.CollectionReference ref =
            web_firestore.FirebaseFirestore.instance.collection('users');
        await ref
            .doc(user!.uid)
            .get()
            .then((web_firestore.DocumentSnapshot documentSnapshot) {
          challenge_count = documentSnapshot['challenges_count'].toString();
          world_progress = documentSnapshot['world_progress'].toString();
          subsections_progress =
              documentSnapshot['subsection_progress'].toString();
        });
      } catch (e) {
        showInSnackBar('Error->${e}', context);
      }
    } else {
      try {
        final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
        var user = await firebaseAuth.getUser();
        var ref = Firestore.instance.collection('users').document(user.id);
        var query = await ref.get();
        challenge_count = query['challenges_count'].toString();
        world_progress = query['world_progress'].toString();
        subsections_progress = query['subsection_progress'].toString();
      } catch (e) {
        showInSnackBar('Error->${e}', context);
      }
    }
    if (challenge_count != "0") {
      for (int index = 1; index <= int.parse(challenge_count); index++) {
        _rewards_count += globals.numberRewardsByChallengeWorlds[index - 1];
      }
      rewards_count = _rewards_count.toString();
    }
    return [
      challenge_count,
      rewards_count,
      world_progress,
      subsections_progress
    ];
  }
```

```
@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;
  //final screenHeight = MediaQuery.of(context).size.height;
  return Scaffold(
      backgroundColor: Colors.white,
      body: SingleChildScrollView(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.end,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            CustomStudentAppBarHome(),
            FutureBuilder(
                future: getChallengesInfo(context),
                builder: (context, AsyncSnapshot snapshot) {
                  if (snapshot.hasData) {
                    int totalPoints = globals.sumTwoDimensionalList(
                        globals.valuesPointsSubsectionsperWorld,
                        int.parse(snapshot.data[2]) - 1,
                        int.parse(snapshot.data[3]));
                    print(
                        "Points: ${totalPoints}, current world: ${int.parse(snapshot.data[2
                    return SingleChildScrollView(
                      child: Padding(
                        padding: const EdgeInsets.only(left: 10, top: 10),
                        child: Column(
                          mainAxisAlignment: MainAxisAlignment.start,
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [
                            Text(
                              "Desaf os Completados",
                              textAlign: TextAlign.start,
                              style: TextStyle(
                                  color: Colors.indigo,
                                  fontWeight: FontWeight.bold,
                                  fontSize: 20),
                            ),
                            if (int.parse(snapshot.data[0].toString()) > 0)
                              Padding(
                                padding:
                                    const EdgeInsets.only(right: 5, top: 10),
                                child: Text(
                                  "En este secci n encontrar s informaci n sobre los des
                                  textAlign: TextAlign.justify,
                                  style: TextStyle(
                                      color: globals.pinkColor,
                                      fontWeight: FontWeight.bold,
                                      fontSize: 15),
                                ),
                              ),
                            SizedBox(
                              height: 15,
                            ),
                            if (int.parse(snapshot.data[0].toString()) > 0)
                              SizedBox(
```

```
                height: 150,
                child: ListView.builder(
                    itemCount: int.parse(
                        snapshot.data[0].toString()),
                    scrollDirection: Axis.horizontal,
                    itemBuilder: (context, index) =>
                        Container(
                          decoration: BoxDecoration(
                            color: Colors.indigo,
                            borderRadius:
                                BorderRadius.circular(20.0),
                          ),
                          height: 150,
                          width: 150,
                          margin: EdgeInsets.all(10),
                          child: Center(
                            child: Text(
                              "${globals.namesChallenges[index]}",
                              style: TextStyle(
                                  color: Colors.white,
                                  fontSize: 20,
                                  fontWeight:
                                      FontWeight.bold),
                            ),
                          ),
                        )),
          )
        else
          Padding(
            padding: const EdgeInsets.only(right: 10),
            child: Text(
                "No has completado ning n desaf o por el momento.",
                textAlign: TextAlign.justify,
                style: TextStyle(
                    color: Colors.grey,
                    fontWeight: FontWeight.bold,
                    fontSize: 20)),
          ),
        if (int.parse(snapshot.data[0].toString()) > 0)
          SingleChildScrollView(
            scrollDirection: Axis.horizontal,
            child: SizedBox(
              height: 50,
              width: size.width,
              child: Padding(
                padding: const EdgeInsets.only(
                    left: 10, right: 10),
                child: StepProgressIndicator(
                  totalSteps: globals.totalChallenges,
                  currentStep:
                      int.parse(snapshot.data[0]),
                  size: 40,
                  selectedColor: Colors.indigo,
                  unselectedColor: Colors.grey,
                  roundedEdges: Radius.circular(10),
                  customStep: (index, color, _) =>
```

```
                    color == Colors.indigo
                        ? Container(
                            width: 50,
                            color: color,
                            child: Icon(
                              Icons.military_tech,
                              color: Colors.white,
                              size: 20,
                            ),
                          )
                        : Container(
                            width: 50,
                            color: color,
                            child: Icon(
                              Icons
                                  .do_disturb_off_outlined,
                            ),
                          ),
                ),
              ),
            ),
          ),
          if (int.parse(snapshot.data[0].toString()) > 0 &&
              int.parse(snapshot.data[0].toString()) != 1)
            Text(
              "Vas ${snapshot.data[0]} desaf os completados de ${globals
              textAlign: TextAlign.start,
              style: TextStyle(
                  color: Colors.black,
                  fontWeight: FontWeight.bold,
                  fontSize: 15),
            ),
          if (int.parse(snapshot.data[0].toString()) > 0 &&
              int.parse(snapshot.data[0].toString()) == 1)
            Text(
              "Vas ${snapshot.data[0]} desafio completado de ${globals.to
              textAlign: TextAlign.start,
              style: TextStyle(
                  color: Colors.black,
                  fontWeight: FontWeight.bold,
                  fontSize: 15),
            ),
          SizedBox(
            height: 15,
          ),
          Text(
            "Recompensas obtenidas",
            textAlign: TextAlign.start,
            style: TextStyle(
                color: Colors.indigo,
                fontWeight: FontWeight.bold,
                fontSize: 20),
          ),
          SizedBox(
            height: 5,
          ),
```

```
if (int.parse(snapshot.data[0].toString()) > 0)
  Padding(
    padding:
        const EdgeInsets.only(right: 5, top: 5),
    child: Text(
      "En␣este␣secci n␣encontrar s␣informaci n␣sobre␣las␣rec
      textAlign: TextAlign.justify,
      style: TextStyle(
          color: globals.pinkColor,
          fontWeight: FontWeight.bold,
          fontSize: 15),
    ),
  ),
if (int.parse(snapshot.data[0].toString()) == 0)
  Padding(
    padding: const EdgeInsets.only(right: 10),
    child: Text(
        "No␣tienes␣ninguna␣recompensa␣por␣el␣momento.",
        textAlign: TextAlign.justify,
        style: TextStyle(
            color: Colors.grey,
            fontWeight: FontWeight.bold,
            fontSize: 20)),
  ),
SizedBox(
  height: 15,
),
if (int.parse(snapshot.data[0].toString()) > 0)
  Row(
    children: [
      CircularStepProgressIndicator(
        totalSteps: globals.rewardsTotalPoints,
        currentStep: int.parse(
            snapshot.data[1].toString()),
        selectedColor: Colors.indigo,
        unselectedColor: Colors.grey,
        padding: 0,
        width: 100,
        child: Icon(
          Icons.monetization_on,
          color: Colors.indigo,
          size: 84,
        ),
      ),
      SizedBox(
        width: 5,
      ),
      Text(
          "Total␣de␣Recompensas␣${snapshot.data[1]}",
          textAlign: TextAlign.start,
          style: TextStyle(
              color: Colors.black,
              fontWeight: FontWeight.bold,
              fontSize: 20)),
    ],
  ),
```

```
SizedBox(
  height: 5,
),
if (int.parse(snapshot.data[0].toString()) > 0)
  Text(
    "Vas␣${snapshot.data[1]}␣recompensas␣obtenidas␣de␣${globals
    textAlign: TextAlign.start,
    style: TextStyle(
        color: Colors.black,
        fontWeight: FontWeight.bold,
        fontSize: 15),
  ),
SizedBox(
  height: 10,
),
Text(
  "Puntos␣totales",
  textAlign: TextAlign.start,
  style: TextStyle(
      color: Colors.indigo,
      fontWeight: FontWeight.bold,
      fontSize: 20),
),
SizedBox(
  height: 10,
),
Padding(
  padding:
      const EdgeInsets.only(right: 5, bottom: 10),
  child: Text(
    "En␣este␣secci n␣encontrar s␣informaci n␣sobre␣los␣punto
    textAlign: TextAlign.justify,
    style: TextStyle(
        color: globals.pinkColor,
        fontWeight: FontWeight.bold,
        fontSize: 15),
  ),
),
SizedBox(
  height: 10,
),
Padding(
  padding: const EdgeInsets.only(
      right: 10, bottom: 10),
  child: StepProgressIndicator(
    totalSteps: globals.totalpointsSubsections,
    currentStep: totalPoints,
    size: 8,
    padding: 0,
    selectedColor: Colors.indigo,
    unselectedColor: Colors.grey,
    roundedEdges: Radius.circular(10),
    selectedGradientColor: LinearGradient(
      begin: Alignment.topLeft,
      end: Alignment.bottomRight,
      colors: [Colors.indigo, Colors.indigo],
```

```
      ),
      unselectedGradientColor: LinearGradient(
        begin: Alignment.topLeft,
        end: Alignment.bottomRight,
        colors: [Colors.grey, Colors.grey],
      ),
    ),
  ),
  Text(
    "Vas ${totalPoints} puntos de ${globals.totalpointsSubsection
    textAlign: TextAlign.start,
    style: TextStyle(
        color: Colors.black,
        fontWeight: FontWeight.bold,
        fontSize: 15),
  ),
  Text(
    "Progreso de Mundos",
    textAlign: TextAlign.start,
    style: TextStyle(
        color: Colors.indigo,
        fontWeight: FontWeight.bold,
        fontSize: 20),
  ),
  SizedBox(
    height: 10,
  ),
  if (int.parse(snapshot.data[0].toString()) > 0)
    Padding(
      padding: const EdgeInsets.only(
          right: 5, bottom: 10),
      child: Text(
        "En este secci n encontrar s tu progreso sobre tus munc
        textAlign: TextAlign.justify,
        style: TextStyle(
            color: globals.pinkColor,
            fontWeight: FontWeight.bold,
            fontSize: 15),
      ),
    ),
  SingleChildScrollView(
    scrollDirection: Axis.horizontal,
    child: SizedBox(
      height: 50,
      width: size.width,
      child: Padding(
        padding: const EdgeInsets.only(
            left: 10, right: 10),
        child: StepProgressIndicator(
          totalSteps: globals.numberWorlds - 1,
          currentStep: int.parse(snapshot.data[2]),
          selectedColor: Colors.indigo,
          unselectedColor: Colors.grey,
          roundedEdges: Radius.circular(10),
          size: 40,
          customStep: (index, color, _) =>
```

```
                color == Colors.indigo
                    ? Container(
                        width: 50,
                        color: color,
                        child: Icon(
                          Icons.emoji_flags,
                          color: Colors.white,
                          size: 20,
                        ),
                      )
                    : Container(
                        width: 50,
                        color: color,
                        child: Icon(
                          Icons.public_off_outlined,
                          size: 20,
                        ),
                      ),
          ),
        ),
      ),
    ),
    Text(
      "Est s␣en␣el␣mundo␣${globals.worldsNames[int.parse(snapshot.
      textAlign: TextAlign.start,
      overflow: TextOverflow.ellipsis,
      maxLines: 3,
      style: TextStyle(
          color: Colors.black,
          fontWeight: FontWeight.bold,
          fontSize: 15),
    ),
    Text(
      "Progreso␣de␣Subsecci n",
      textAlign: TextAlign.start,
      style: TextStyle(
          color: Colors.indigo,
          fontWeight: FontWeight.bold,
          fontSize: 20),
    ),
    SizedBox(
      height: 10,
    ),
    Padding(
      padding:
          const EdgeInsets.only(right: 5, bottom: 10),
      child: Text(
        "En␣este␣secci n␣encontrar s␣informaci n␣sobre␣tu␣progre
        textAlign: TextAlign.justify,
        style: TextStyle(
            color: globals.pinkColor,
            fontWeight: FontWeight.bold,
            fontSize: 15),
      ),
    ),
    SingleChildScrollView(
```

```
                    scrollDirection: Axis.horizontal,
                    child: SizedBox(
                      height: 50,
                      width: size.width,
                      child: Padding(
                        padding: const EdgeInsets.only(
                            left: 10, right: 10),
                        child: StepProgressIndicator(
                          totalSteps: globals.worldIndexSubsections[
                              int.parse(snapshot.data[2]) - 1],
                          currentStep:
                              int.parse(snapshot.data[3]) + 1,
                          selectedColor: Colors.indigo,
                          unselectedColor: Colors.grey,
                          roundedEdges: Radius.circular(10),
                          size: 40,
                          customStep: (index, color, _) => color ==
                                  Colors.indigo
                              ? Container(
                                  width: 50,
                                  color: color,
                                  child: Icon(
                                    Icons.event_available_outlined,
                                    color: Colors.white,
                                    size: 20,
                                  ),
                                )
                              : Container(
                                  width: 50,
                                  color: color,
                                  child: Icon(
                                    Icons.priority_high,
                                    size: 20,
                                  ),
                                ),
                        ),
                      ),
                    ),
                  ),
                  Padding(
                    padding: const EdgeInsets.only(right: 8),
                    child: Text(
                      "Est s cursando la subsecci n de ${globals.subsectionsNam
                      textAlign: TextAlign.justify,
                      maxLines: 3,
                      style: TextStyle(
                          color: Colors.black,
                          fontWeight: FontWeight.bold,
                          fontSize: 15),
                    ),
                  ),
                ],
              ),
            ),
          );
        } else {
```

```
                    return Column(
                      children: [
                        SizedBox(
                          height: size.height / 4,
                        ),
                        Text(
                          'Cargando ...',
                          style: TextStyle(
                              color: Colors.indigo,
                              fontWeight: FontWeight.bold,
                              fontSize: 36),
                        ),
                        SizedBox(
                          height: 10,
                        ),
                        CircularProgressIndicator(
                          color: Colors.indigo,
                        )
                      ],
                    );
                  }
                })
            ],
          ),
        ));
  }
}
```

8.22 Anexo V:

```
import subprocess
import json


def lambda_handler(event, context):
    body = json.loads(event['body'])

    command = body['command']


    process = subprocess.Popen(['python', '-c', command], stdout=subprocess.PIPE
    , stderr=subprocess.PIPE)
    stdout, stderr = process.communicate()

    result = ""

    if len(stderr.decode('utf-8')) == 0:
        result = stdout.decode('utf-8')
    else:
        if len(stdout.decode('utf-8') == 0):
            result = stderr.decode('utf-8')


    # Create the response object
    response = {
        'statusCode': 200,
        'body': json.dumps(
            stdout.decode('utf-8')),
        'headers': {
            'Content-Type': 'application/json',
            'Access-Control-Allow-Origin': '*'
        }
    }

    return response
```

8.23 Anexo W:

```
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:Openedu.IA/widgets/widgets.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';
import 'package:Openedu.IA/constants/constants.dart' as globals;

Future<String> executeCommand(String pythonCode) async {
  String result = '';
  final body = {
    'command': "$pythonCode",
  };

  try {
    final url =
        "URL␣de␣la␣funci n␣serverless";
    await http.post(Uri.parse(url), body: jsonEncode(body)).then((value) {
      result = value.body.toString();
      print("onThen>␣" + value.body.toString());
    }).onError((error, stackTrace) {
      result = error.toString();
      print("onError>␣" +
          error.toString() +
          "␣stackTrace>␣" +
          stackTrace.toString());
    });

    return result
        .replaceAll('"stdout"', 'resultado')
        .replaceAll("stderr", 'errores')
        .replaceAll(r'\n', '\n')
        .replaceAll('"', '');
  } catch (e) {
    if (!kIsWeb)
      return "Error␣compilando␣tu␣c digo ,␣recuerda␣tener␣en␣cuenta␣la
␣␣␣␣␣␣sintaxis␣y␣los␣espacios␣+␣${e}";
    else {
      print(e);
      return result.replaceAll(r'\n', '\n').replaceAll('"', '');
    }
  }
}

class CustomStudentPythonShell extends StatefulWidget {
  const CustomStudentPythonShell({Key? key}) : super(key: key);

  @override
  State<CustomStudentPythonShell> createState() =>
      _CustomStudentPythonShellState();
}

class _CustomStudentPythonShellState extends State<CustomStudentPythonShell> {
  late TextEditingController _controller;
  late FocusNode _focusNode;
```

```
@override
void initState() {
  _controller = TextEditingController();
  _focusNode = FocusNode();
  super.initState();
}

@override
void dispose() {
  _focusNode.dispose();
  _controller.dispose();
  super.dispose();
}

void addIntendation() {
  TextEditingController _updatedController = TextEditingController();

  int currentPosition = _controller.selection.start;

  String controllerText = _controller.text;
  String text = controllerText.substring(0, currentPosition) +
      "␣␣␣␣" +
      controllerText.substring(currentPosition, controllerText.length);

  _updatedController.value = TextEditingValue(
    text: text,
    selection: TextSelection(
      baseOffset: _controller.text.length + 4,
      extentOffset: _controller.text.length + 4,
    ),
  );

  setState(() {
    _controller = _updatedController;
  });
}

@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;

  return Scaffold(
    backgroundColor: Colors.transparent,
    body: SingleChildScrollView(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
          CustomStudentAppBarHome(),
          SizedBox(
            height: 5,
          ),
          Padding(
            padding: const EdgeInsets.only(left: 10, right: 10),
            child: Text(
              'Compila␣tu␣c digo␣de␣Python␣en␣este␣apartado,␣recuerda
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣tener␣en␣cuenta␣los␣espacios␣y␣los␣saltos␣de␣linea␣entre
```

```
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣tus␣lineas␣de␣c digo.',

                textAlign: TextAlign.justify,
                style: TextStyle(
                    color: Colors.indigo,
                    fontWeight: FontWeight.bold,
                    fontSize: 15),
              ),
            ),
            Row(
              children: [
                Expanded(
                  flex: 3,
                  child: Padding(
                    padding: const EdgeInsets.only(left: 5, right: 5),
                    child: TextFormField(
                        style: TextStyle(color: Colors.indigo),
                        keyboardType: TextInputType.multiline,
                        focusNode: _focusNode,
                        controller: _controller,
                        minLines: 5,
                        maxLines: null,
                        decoration: InputDecoration(
                            hintStyle: TextStyle(color: globals.pinkColor),
                            hintText:
                                "ejemplo:␣\nimport␣sys\nfor␣i␣in␣range(1,11):␣\n␣print(i,␣end
                  ),
                ),
                SizedBox(
                  width: 5,
                ),
                Container(
                  margin: EdgeInsets.only(top: 10, right: 5),
                  child: SizedBox(
                    width: 50,
                    child: MaterialButton(
                      shape: const RoundedRectangleBorder(
                          borderRadius:
                              BorderRadius.all(Radius.circular(20.0))),
                      elevation: 5.0,
                      height: 50,
                      color: Colors.indigo,
                      onPressed: () async {
                        FocusScope.of(context).unfocus();
                        setState(() {});
                      },
                      child: Icon(
                        Icons.rocket_launch,
                        color: Colors.white,
                      ),
                    ),
                  ),
                ),
              ],
            ),
            SizedBox(height: 16),
```

```
Row(
  mainAxisAlignment: MainAxisAlignment.start,
  children: [
    Padding(
      padding: const EdgeInsets.only(right: 5, left: 5),
      child: FutureBuilder(
          future: executeCommand(_controller.text),
          builder: (context, AsyncSnapshot snapshot) {
            if (snapshot.hasData) {
              if (snapshot.connectionState ==
                  ConnectionState.waiting) {
                return Center(
                    child: Column(
                  children: [
                    SizedBox(
                      height: 20,
                    ),
                    Text(
                      'Cargando ...',
                      style: TextStyle(
                          color: Colors.indigo,
                          fontSize: 30,
                          fontWeight: FontWeight.bold),
                    ),
                    SizedBox(
                      height: 20,
                    ),
                    CircularProgressIndicator(
                      color: Colors.indigo,
                    )
                  ],
                ));
              }
              return Column(
                children: [
                  Container(
                      width: size.width - 60,
                      margin: EdgeInsets.only(
                          top: 5, left: 15, right: 15, bottom: 5),
                      child: SingleChildScrollView(
                        scrollDirection: Axis.horizontal,
                        child: Row(
                          mainAxisAlignment:
                              MainAxisAlignment.start,
                          children: [
                            Text(
                              snapshot.data,
                              maxLines: null,
                              overflow: TextOverflow.ellipsis,
                              textAlign: TextAlign.justify,
                              style: TextStyle(
                                color: Colors.indigo,
                                fontWeight: FontWeight.bold,
                                fontSize: 15,
                              ),
                            ),
```

```
                                    ],
                                  ),
                                )),
                          ],
                        );
                      } else {
                        if (!snapshot.hasData ||
                            snapshot.connectionState ==
                                ConnectionState.waiting) {
                          return Text('');
                        } else {
                          return Container(
                            margin: EdgeInsets.all(5),
                            child: Center(
                              child: Text(
                                'Aqu aparecer lo que ser compilado ...',
                                style: TextStyle(
                                    color: Colors.indigo,
                                    fontWeight: FontWeight.bold,
                                    fontSize: 16),
                              ),
                            ),
                          );
                        }
                      }
                    }),
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }
}
```

8.24 Anexo X:

```
import 'package:flutter/material.dart';
import 'package:Openedu.IA/widgets/widgets.dart';
import 'dart:async';
import 'package:chatgpt_completions/chatgpt_completions.dart';

class CustomStudentAssistantBody extends StatefulWidget {
  const CustomStudentAssistantBody({Key? key}) : super(key: key);

  @override
  State<CustomStudentAssistantBody> createState() =>
      _CustomStudentAssistantBodyState();
}

class _CustomStudentAssistantBodyState
    extends State<CustomStudentAssistantBody> {
  final TextEditingController messageController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;

    Future<String> textAnswer(String prompt) async {
      ChatGPTCompletions.instance.initialize(
          apiKey: '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');
      print("prompt: ${prompt}");
      try {
        if (prompt.isEmpty) {
          return " ";
        } else {
          String? responseWithoutStream = await ChatGPTCompletions.instance
              .textCompletions(TextCompletionsParams(messagesTurbo: [
            MessageTurbo(
              role: TurboRole.user,
              content: prompt,
            ),
          ], model: GPTModel.gpt3p5turbo));
          return responseWithoutStream ??
              "Error obteniendo la respuesta, intentalo m s tarde.";
        }
      } catch (e) {
        return "No se pudo obtener la respuesta ${e}";
      }
    }

    return Scaffold(
      backgroundColor: Colors.transparent,
      body: SingleChildScrollView(
        child: Column(mainAxisAlignment: MainAxisAlignment.start, children: [
          CustomStudentAppBarHome(),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,
            children: [
              Container(
                margin: EdgeInsets.only(left: 5, top: 10),
```

```
      child: SizedBox(
        width: size.width - 70,
        child: TextFormField(
          controller: messageController,
          decoration: InputDecoration(
            filled: true,
            fillColor: Colors.white,
            hintText: 'Preguntame lo que quieras ...',
            hintStyle: const TextStyle(
                color: Colors.indigo, fontWeight: FontWeight.bold),
            enabled: true,
            contentPadding: const EdgeInsets.only(
                left: 14.0, bottom: 8.0, top: 8.0),
            focusedBorder: OutlineInputBorder(
              borderSide: const BorderSide(color: Colors.indigo),
              borderRadius: BorderRadius.circular(20),
            ),
            enabledBorder: OutlineInputBorder(
              borderSide: const BorderSide(color: Colors.indigo),
              borderRadius: BorderRadius.circular(20),
            ),
          ),
          validator: (value) {
            if (value!.isEmpty) {
              return null;
            } else {
              return null;
            }
          },
          onChanged: (value) {},
          keyboardType: TextInputType.multiline,
        ),
      ),
    ),
    Container(
      margin: EdgeInsets.only(top: 10, right: 5),
      child: SizedBox(
        width: 50,
        child: MaterialButton(
          shape: const RoundedRectangleBorder(
              borderRadius: BorderRadius.all(Radius.circular(20.0))),
          elevation: 5.0,
          height: 50,
          color: Colors.indigo,
          onPressed: () async {
            FocusScope.of(context).unfocus();
            setState(() {});
          },
          child: Icon(
            Icons.send,
            color: Colors.white,
          ),
        ),
      ),
    )
  ],
```

```
),
FutureBuilder(
    future: textAnswer(messageController.text),
    builder: (context, AsyncSnapshot snapshot) {
      if (snapshot.hasData) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          Center(
              child: Column(
            children: [
              SizedBox(
                height: 20,
              ),
              Text(
                'Cargando ...',
                style: TextStyle(
                    color: Colors.indigo,
                    fontSize: 30,
                    fontWeight: FontWeight.bold),
              ),
              SizedBox(
                height: 20,
              ),
              CircularProgressIndicator(
                color: Colors.indigo,
              )
            ],
          ));
        }
        return Container(
            margin: EdgeInsets.only(
                top: 5, left: 15, right: 15, bottom: 5),
            child: Text(
              snapshot.data,
              textAlign: TextAlign.justify,
              style: TextStyle(
                  color: Colors.indigo,
                  fontWeight: FontWeight.bold,
                  fontSize: 20),
            ));
      } else {
        if (!snapshot.hasData ||
            snapshot.connectionState == ConnectionState.waiting) {
          return Text('');
        } else {
          return Container(
            margin: EdgeInsets.all(5),
            child: Center(
              child: Text(
                'Aqu aparecer lo que quieres saber ...',
                style: TextStyle(
                    color: Colors.indigo,
                    fontWeight: FontWeight.bold,
                    fontSize: 16),
              ),
            ),
          );
```

```
                    }
                  }
                }),
            ]),
          ),
        );
    }
}
```

8.25 Anexo Y:

```
import 'package:flutter/material.dart';
import 'dart:math';
import 'package:bonfire/bonfire.dart';
import 'package:Openedu.IA/widgets/custom_world_widgets/custom_challenges_widgets/custom_pacm

class PacManWorld1HomeScreen extends StatelessWidget {
  static const double heightMap = 1004.0;
  static const double tileSize = 48.0;
  const PacManWorld1HomeScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    WidgetsFlutterBinding.ensureInitialized();
    BonfireInjector.instance.put((i) => GameState());
    Size sizeScreen = MediaQuery.of(context).size;
    double size = min(sizeScreen.width, sizeScreen.height);
    double zoom = size / heightMap;
    return Container(
      color: Colors.black,
      child: Center(
        child: SizedBox(
          width: size,
          height: size,
          child: BonfireWidget(
            map: WorldMapByTiled(
              'map.tmj',
              objectsBuilder: {
                'sensor_left': (properties) => SensorGate(
                      position: properties.position,
                      direction: DiractionGate.left,
                    ),
                'sensor_right': (properties) => SensorGate(
                      position: properties.position,
                      direction: DiractionGate.right,
                    ),
                'dot': (properties) => Dot(
                      position: properties.position,
                    ),
                'dot_power': (properties) => DotPower(
                      position: properties.position,
                    ),
                'ghost_red': (properties) => Ghost(
                      position: properties.position,
                      type: GhostType.red,
                    ),
                'ghost_pink': (properties) => Ghost(
                      position: properties.position,
                      type: GhostType.pink,
                    ),
                'ghost_orange': (properties) => Ghost(
                      position: properties.position,
                      type: GhostType.orange,
                    ),
                'ghost_blue': (properties) => Ghost(
```

```
                    position: properties.position,
                    type: GhostType.blue,
                  ),
              },
            ),
            joystick: Joystick(
              keyboardConfig: KeyboardConfig(),
              directional: JoystickDirectional(
                  size: 60,
                  color: Colors.indigo,
                  isFixed: false,
                  margin: EdgeInsets.only(left: 40, bottom: 40)),
            ),
            overlayBuilderMap: {
              'score': ((context, game) => const InterfaceGame()),
            },
            initialActiveOverlays: const ['score'],
            cameraConfig: CameraConfig(
              zoom: zoom,
            ),
            player: PacMan(
              position: PacMan.initialPosition,
            ),
          ),
        ),
      ),
    );
  }
}
```

8.26 Anexo Z:

```
import 'dart:async';
import 'dart:io';
import 'package:Openedu.IA/screens/screens.dart';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:Openedu.IA/widgets/widgets.dart';
import 'package:firedart/firedart.dart';
import 'package:cloud_firestore/cloud_firestore.dart' as web_firestore;
import 'package:firebase_auth/firebase_auth.dart' as web_auth;

Future addChallenge(BuildContext context) async {
  try {
    if (kIsWeb) {
      final _auth = web_auth.FirebaseAuth.instance;
      var user = await _auth.currentUser;
      web_firestore.CollectionReference ref =
          web_firestore.FirebaseFirestore.instance.collection('users');
      await ref
          .doc(user!.uid)
          .get()
          .then((web_firestore.DocumentSnapshot documentSnapshot) async {
        if (int.parse(documentSnapshot['challenges_count']) == 0) {
          await ref.doc(user.uid).update({
            'challenges_count': '1',
          });
        } else {
          await ref.doc(user.uid).update({
            'challenges_count': '2',
          });
        }
      });
    } else {
      final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
      var user = await firebaseAuth.getUser();
      var ref = Firestore.instance.collection('users').document(user.id);
      var userInfo = await ref.get();
      if (int.parse(userInfo['challenges_count']) == 0) {
        await ref.update({
          'challenges_count': '1',
        });
      } else {
        await ref.update({
          'challenges_count': '2',
        });
      }
    }
  } catch (e) {
    showInSnackBar('${e}', context);
  }
}

class Game extends StatefulWidget {
  Game({Key? key}) : super(key: key);
```

```
  @override
  State<Game> createState() => _GameState();
}

class _GameState extends State<Game> {
  final GameLogic _game = GameLogic();

  var levelForCardCount = 0;
  var tries = 0;
  var score = 0;
  var axisNumber = 4;
  late Timer timer;
  int startTime = 60;
  String level = '';
  var complete = 0;
  void startTimer(BuildContext context) {
    if (startTime == 0) {}
    const oneSecond = Duration(seconds: 1);
    timer = Timer.periodic(oneSecond, (timer) {
      if (startTime == 0) {
        timer.cancel();
        _showDialog(context, 'Final del juego', 'Tu puntuaci n fue de: $score');
      } else {
        setState(() {
          startTime--;
        });
      }
    });
  }

  @override
  void initState() {
    super.initState();

    startTimer(context);
  }

  @override
  void didChangeDependencies() {
    super.didChangeDependencies();

    _game.initGame(context);
  }

  @override
  Widget build(BuildContext context) {
    double screenWidth = MediaQuery.of(context).size.width;
    double screenHeight = MediaQuery.of(context).size.height;
    return WillPopScope(
      onWillPop: () async {
        bool willLeave = false;
        timer.cancel();
        // show the confirm dialog
        await showDialog(
            context: context,
            builder: (_) => AlertDialog(
```

```
            title: const Text('Seguro que quiere salir de la partida?'),
            actions: [
              ElevatedButton(
                  style: ElevatedButton.styleFrom(
                      backgroundColor: Colors.indigo),
                  onPressed: () {
                    willLeave = true;
                    Navigator.of(context).pop();
                  },
                  child: const Text('Si')),
              TextButton(
                  onPressed: () {
                    Navigator.of(context).pop();
                    startTimer(context);
                  },
                  child: const Text(
                    'No',
                    style: TextStyle(color: Colors.black),
                  ))
            ],
          ));
  return willLeave;
},
child: Scaffold(
  appBar: AppBar(
    backgroundColor: Colors.indigo,
  ),
  backgroundColor: Colors.indigo,
  body: SingleChildScrollView(
    child: Column(children: [
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          board('Tiempo', '$startTime'),
          board('Puntaje', '$score'),
          board('Movimiento', '$tries')
        ],
      ),
      if (kIsWeb)
        SizedBox(
          height: screenHeight - 150,
          width: screenHeight,
          child: GridView.builder(
              itemCount: _game.cardsImg!.length,
              gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
                crossAxisCount: _game.axiCount,
                crossAxisSpacing: 10,
                mainAxisSpacing: 10,
              ),
              padding: const EdgeInsets.all(16),
              itemBuilder: (context, index) {
                return GestureDetector(
                    onTap: () async {
                      setState(() {
                        tries++;
```

```
                    _game.cardsImg![index] = _game.card_list[index];

                    _game.matchCheck
                        .add({index: _game.card_list[index]});

                    if (_game.matchCheck.length == 2) {
                      if (_game.matchCheck[0].values.first ==
                          _game.matchCheck[1].values.first) {
                        score += 100;
                        complete += 1;

                        _game.matchCheck.clear();
                        if (complete * 2 == _game.cardCount) {
                          _showDialog(context, 'Ganaste',
                              'Tu puntuaci n fue de: $score');
                          timer.cancel();
                        }
                      } else {
                        Future.delayed(
                            const Duration(milliseconds: 500), () {
                          setState(() {
                            _game.cardsImg![_game.matchCheck[0].keys
                                .first] = _game.hiddenCard;
                            _game.cardsImg![_game.matchCheck[1].keys
                                .first] = _game.hiddenCard;

                            _game.matchCheck.clear();
                          });
                        });
                      }
                    });

                    // _game.matchCheck
                  },
                  child: Container(
                    decoration: BoxDecoration(
                        color: Colors.white,
                        borderRadius: BorderRadius.circular(10),
                        image: DecorationImage(
                            image: AssetImage(_game.cardsImg![index]),
                            fit: BoxFit.cover)),
                  ));
            }),
      ),
    if (!kIsWeb)
      if (Platform.isLinux || Platform.isMacOS || Platform.isWindows)
        SizedBox(
          height: screenHeight - 150,
          width: screenHeight,
          child: GridView.builder(
              itemCount: _game.cardsImg!.length,
              gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
                crossAxisCount: _game.axiCount,
                crossAxisSpacing: 10,
```

```
            mainAxisSpacing: 10,
          ),
          padding: const EdgeInsets.all(16),
          itemBuilder: (context, index) {
            return GestureDetector(
                onTap: () {
                  setState(() {
                    tries++;

                    _game.cardsImg![index] = _game.card_list[index];

                    _game.matchCheck
                        .add({index: _game.card_list[index]});

                    if (_game.matchCheck.length == 2) {
                      if (_game.matchCheck[0].values.first ==
                          _game.matchCheck[1].values.first) {
                        score += 100;
                        complete += 1;

                        _game.matchCheck.clear();
                        if (complete * 2 == _game.cardCount) {
                          _showDialog(context, 'Ganaste',
                              'Tu puntuaci n fue de: $score');
                          timer.cancel();
                        }
                      } else {
                        Future.delayed(
                            const Duration(milliseconds: 500), () {
                          setState(() {
                            _game.cardsImg![_game.matchCheck[0].keys
                                .first] = _game.hiddenCard;
                            _game.cardsImg![_game.matchCheck[1].keys
                                .first] = _game.hiddenCard;

                            _game.matchCheck.clear();
                          });
                        });
                      }
                    }
                  });

                  // _game.matchCheck
                },
                child: Container(
                  decoration: BoxDecoration(
                      color: Colors.white,
                      borderRadius: BorderRadius.circular(10),
                      image: DecorationImage(
                          image: AssetImage(_game.cardsImg![index]),
                          fit: BoxFit.cover)),
                ));
          }),
    ),
if (!kIsWeb)
  if (Platform.isAndroid || Platform.isIOS)
```

```
SizedBox(
  height: screenWidth,
  width: screenWidth,
  child: GridView.builder(
      itemCount: _game.cardsImg!.length,
      gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
        crossAxisCount: _game.axiCount,
        crossAxisSpacing: 10,
        mainAxisSpacing: 10,
      ),
      padding: const EdgeInsets.all(16),
      itemBuilder: (context, index) {
        return GestureDetector(
            onTap: () {
              setState(() {
                tries++;

                _game.cardsImg![index] = _game.card_list[index];

                _game.matchCheck
                    .add({index: _game.card_list[index]});

                if (_game.matchCheck.length == 2) {
                  if (_game.matchCheck[0].values.first ==
                      _game.matchCheck[1].values.first) {
                    score += 100;
                    complete += 1;

                    _game.matchCheck.clear();
                    if (complete * 2 == _game.cardCount) {
                      _showDialog(context, 'Ganaste',
                          'Tu puntuaci n fue de: $score');
                      timer.cancel();
                    }
                  } else {
                    Future.delayed(
                        const Duration(milliseconds: 500), () {
                      setState(() {
                        _game.cardsImg![_game.matchCheck[0].keys
                            .first] = _game.hiddenCard;
                        _game.cardsImg![_game.matchCheck[1].keys
                            .first] = _game.hiddenCard;

                        _game.matchCheck.clear();
                      });
                    });
                  }
                }
              });

              // _game.matchCheck
            },
            child: Container(
              decoration: BoxDecoration(
                  color: Colors.white,
                  borderRadius: BorderRadius.circular(10),
```

```
                                    image: DecorationImage(
                                        image: AssetImage(_game.cardsImg![index]),
                                        fit: BoxFit.cover)),
                                ));
                            }),
                    ),
            ]),
          ),
        ),
      );
  }
}

void _showDialog(BuildContext context, String title, String info) {
  showDialog(
      context: context,
      barrierDismissible: false,
      builder: (BuildContext context) {
        return AlertDialog(
          title: Text(title),
          content: Text(info),
          actions: [
            TextButton(
              child: Text('Ir a inicio'),
              onPressed: () {
                if (title == "Ganaste") addChallenge(context);
                Navigator.push(context, MaterialPageRoute(builder: (context) {
                  return HomeStudentScreen();
                }));
              },
            )
          ],
        );
      });
}
```

8.27 Anexo AA:

```
import 'package:Openedu.IA/widgets/widgets.dart';
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';

class CustomWorldSubsectionHomeScreen extends StatefulWidget {
  final String subsectionName;
  final String subsectionProgress;
  final String worldProgress;
  final String actualWorld;
  final String actualSubsection;
  const CustomWorldSubsectionHomeScreen({
    Key? key,
    required this.subsectionName,
    required this.subsectionProgress,
    required this.worldProgress,
    required this.actualWorld,
    required this.actualSubsection,
  }) : super(key: key);

  @override
  State<CustomWorldSubsectionHomeScreen> createState() =>
      _CustomWorldSubsectionHomeScreenState();
}

class _CustomWorldSubsectionHomeScreenState
    extends State<CustomWorldSubsectionHomeScreen> {
  int _currentIndex = 0;

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;

    return Scaffold(
        appBar: AppBar(
          centerTitle: true,
          backgroundColor: Colors.indigo,
          toolbarHeight: 100,
          elevation: 0,
          shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.only(
                  bottomRight: Radius.circular(70),
                  bottomLeft: Radius.circular(70))),
          title: Text(
            this.widget.subsectionName,
            textAlign: TextAlign.justify,
            maxLines: 3,
            style: TextStyle(color: Colors.white),
          ),
        ),
        body: SizedBox(
          width: size.width,
          height: size.height,
          child: Stack(children: [
            const CustomStudentBackground(),
```

```
            if (_currentIndex == 0)
              CustomWorldSubsectionWidgetVideos(
                  world_index: widget.actualWorld,
                  subsection_index: widget.actualSubsection),
            if (_currentIndex == 1)
              CustomWorldSubsectionWidgetQuiz(
                worldIndex: widget.actualWorld,
                subsectionIndex: widget.actualSubsection,
                )
          ]),
        ),
        bottomNavigationBar: Container(
          color: Colors.indigo,
          child: Padding(
            padding: const EdgeInsets.symmetric(horizontal: 15.0, vertical: 20),
            child: GNav(
              backgroundColor: Colors.transparent,
              color: Colors.transparent,
              activeColor: Colors.white,
              tabBackgroundColor: Colors.purpleAccent.shade100,
              gap: 8,
              onTabChange: (index) {
                print(widget.actualWorld);
                setState(() {
                  _currentIndex = index;
                });
              },
              padding: const EdgeInsets.all(5),
              tabs: [
                GButton(
                  icon: Icons.videocam,
                  iconColor: Colors.white,
                  iconActiveColor: Colors.white,
                  iconSize: 30,
                  text: 'Videos',
                  textColor: Colors.white,
                ),
                if (widget.actualWorld != 6)
                  GButton(
                    icon: Icons.quiz_outlined,
                    iconColor: Colors.white,
                    iconActiveColor: Colors.white,
                    iconSize: 30,
                    text: 'Quices',
                    textColor: Colors.white,
                  ),
              ],
            ),
          ),
        ));
  }
}

Future getUserProgress(BuildContext context) async {
  String world_progress = "";
  String subsections_progress = "";
```

```
  if (kIsWeb) {
    try {
      final _auth = web_auth.FirebaseAuth.instance;
      var user = await _auth.currentUser;

      web_firestore.CollectionReference ref =
          web_firestore.FirebaseFirestore.instance.collection('users');
      await ref
          .doc(user!.uid)
          .get()
          .then((web_firestore.DocumentSnapshot documentSnapshot) {
        world_progress = documentSnapshot['world_progress'].toString();
        subsections_progress =
            documentSnapshot['subsection_progress'].toString();
      });
    } catch (e) {
      showInSnackBar('Error-> ${e}', context);
    }
  } else {
    try {
      final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
      var user = await firebaseAuth.getUser();
      var ref = Firestore.instance.collection('users').document(user.id);
      var query = await ref.get();
      world_progress = query['world_progress'].toString();
      subsections_progress = query['subsection_progress'].toString();
    } catch (e) {
      showInSnackBar('Error-> ${e}', context);
    }
  }
  return [world_progress, subsections_progress];
}

getCorrectAnswer(List<String> quizOptions, String correctAnswer) {
  for (int index = 0; index < quizOptions.length; index++) {
    if (correctAnswer == quizOptions[index]) {
      return index + 1;
    }
  }
  return -1;
}

Future completedSection(BuildContext context, String actualWorld, worldProgress,
    actualSubsection, subsectionProgress) async {
  if (kIsWeb) {
    try {
      final _auth = web_auth.FirebaseAuth.instance;
      var user = await _auth.currentUser;
      web_firestore.CollectionReference ref =
          web_firestore.FirebaseFirestore.instance.collection('users');
      if (int.parse(actualWorld) + 1 == int.parse(worldProgress)) {
        if (int.parse(subsectionProgress) > int.parse(actualSubsection) &&
            int.parse(subsectionProgress) <
                globals.worldIndexSubsections[int.parse(actualWorld)]) {
          if (int.parse(worldProgress) < 6)
            await ref.doc(user!.uid).update({
```

```
        'subsection_progress':
            (int.parse(subsectionProgress) + 1).toString()
      });
    Navigator.pushAndRemoveUntil(context,
        MaterialPageRoute(builder: (context) {
      return HomeStudentScreen();
    }), (route) => false);
    showInSnackBar('Secci n completada!', context);
  }
  if (int.parse(subsectionProgress) ==
      globals.worldIndexSubsections[int.parse(actualWorld)] - 1) {
    if (int.parse(worldProgress) < 6)
      await ref.doc(user!.uid).update({
        'subsection_progress': '0',
        'world_progress': (int.parse(actualWorld) + 2).toString()
      });
    Navigator.pushAndRemoveUntil(context,
        MaterialPageRoute(builder: (context) {
      return HomeStudentScreen();
    }), (route) => false);
    if (actualWorld != "5")
      showInSnackBar('Mundo nuevo desbloqueado', context);
    if (actualWorld == "5")
      showInSnackBar(
          'Terminaste todos los mundos, felicitaciones!', context);
  }
  if (int.parse(actualSubsection) == int.parse(subsectionProgress)) {
    if (int.parse(subsectionProgress) <
        globals.worldIndexSubsections[int.parse(worldProgress)] - 1) {
      if (int.parse(worldProgress) < 6)
        await ref.doc(user!.uid).update({
          'subsection_progress':
              (int.parse(subsectionProgress) + 1).toString()
        });
      Navigator.pushAndRemoveUntil(context,
          MaterialPageRoute(builder: (context) {
        return HomeStudentScreen();
      }), (route) => false);
      showInSnackBar('Secci n completada!', context);
    }
  } else {
    if (int.parse(subsectionProgress) ==
            globals.worldIndexSubsections[int.parse(actualWorld)] - 1 &&
        int.parse(worldProgress) <= 6) {
      if (int.parse(worldProgress) < 6)
        await ref.doc(user!.uid).update({
          'subsection_progress': '0',
          'world_progress': (int.parse(actualWorld) + 2).toString()
        });
      Navigator.pushAndRemoveUntil(context,
          MaterialPageRoute(builder: (context) {
        return HomeStudentScreen();
      }), (route) => false);
      showInSnackBar('Mundo nuevo desbloqueado', context);
    }
    if (int.parse(subsectionProgress) > int.parse(actualSubsection) &&
```

```
            int.parse(subsectionProgress) <
                globals.worldIndexSubsections[int.parse(actualWorld)] - 1) {
          if (int.parse(worldProgress) < 6)
            await ref.doc(user!.uid).update({
              'subsection_progress':
                  (int.parse(subsectionProgress) + 1).toString()
            });
        }
      }
    }
  } catch (e) {
    showInSnackBar('Error ${e}', context);
  }
} else {
  try {
    final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
    var user = await firebaseAuth.getUser();
    var ref = Firestore.instance.collection('users').document(user.id);
    if (int.parse(actualWorld) + 1 == int.parse(worldProgress)) {
      if (worldProgress == "5") {
        await ref.update({'subsection_progress': '0', 'world_progress': '6'});
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        showInSnackBar(' completada !', context);
      }
      if (int.parse(subsectionProgress) > int.parse(actualSubsection) &&
          int.parse(subsectionProgress) <
              globals.worldIndexSubsections[int.parse(actualWorld)]) {
        if (int.parse(worldProgress) < 6)
          await ref.update({
            'subsection_progress':
                (int.parse(subsectionProgress) + 1).toString()
          });
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        showInSnackBar('Secci n completada !', context);
      }
      if (int.parse(subsectionProgress) ==
          globals.worldIndexSubsections[int.parse(actualWorld)] - 1) {
        if (int.parse(worldProgress) < 6)
          await ref.update({
            'subsection_progress': '0',
            'world_progress': (int.parse(actualWorld) + 2).toString()
          });
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        if (actualWorld != "5")
          showInSnackBar('Mundo nuevo desbloqueado', context);
        if (actualWorld == "5") {
          showInSnackBar(
```

```
            'Terminaste todos los mundos, felicitaciones!', context);
      }
    }
    if (int.parse(actualSubsection) == int.parse(subsectionProgress)) {
      if (int.parse(subsectionProgress) <
          globals.worldIndexSubsections[int.parse(worldProgress) - 1]) {
        if (int.parse(worldProgress) < 6)
          await ref.update({
            'subsection_progress':
                (int.parse(subsectionProgress) + 1).toString()
          });
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        showInSnackBar('Secci n completada!', context);
      }
    } else {
      if (int.parse(subsectionProgress) ==
              globals.worldIndexSubsections[int.parse(actualWorld)] - 1 &&
          int.parse(worldProgress) <= 6) {
        if (int.parse(worldProgress) < 6)
          await ref.update({
            'subsection_progress': '0',
            'world_progress': (int.parse(actualWorld) + 2).toString()
          });
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        showInSnackBar('Mundo nuevo desbloqueado', context);
      }
      if (int.parse(worldProgress) == 6) {
        Navigator.pushAndRemoveUntil(context,
            MaterialPageRoute(builder: (context) {
          return HomeStudentScreen();
        }), (route) => false);
        showInSnackBar(
            'Terminaste todos los mundos, felicitaciones!', context);
      }
      if (int.parse(subsectionProgress) > int.parse(actualSubsection) &&
          int.parse(subsectionProgress) <
              globals.worldIndexSubsections[int.parse(actualWorld)] - 1) {
        if (int.parse(worldProgress) < 6)
          await ref.update({
            'subsection_progress':
                (int.parse(subsectionProgress) + 1).toString()
          });
      }
    }
  } else {
    if (int.parse(actualWorld) + 1 < int.parse(worldProgress)) {
      Navigator.pushAndRemoveUntil(context,
          MaterialPageRoute(builder: (context) {
        return HomeStudentScreen();
      }), (route) => false);
```

```
          showInSnackBar(
              'Ya habias respondido correctamente el quiz anteriormente',
              context);
        }
      }
    } catch (e) {
      showInSnackBar('Error ${e}', context);
    }
  }
}

class CustomWorldSubsecionShowQuiz extends StatelessWidget {
  final String actualWorld;
  final String actualSubsection;
  final String quizQuestion;
  final String quizCorrectAnswer;
  final String quizFirstOption;
  final String quizSecondOption;
  final String quizThreeOption;
  final String quizFourOption;
  const CustomWorldSubsecionShowQuiz({
    Key? key,
    required this.actualWorld,
    required this.actualSubsection,
    required this.quizQuestion,
    required this.quizCorrectAnswer,
    required this.quizFirstOption,
    required this.quizSecondOption,
    required this.quizThreeOption,
    required this.quizFourOption,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final Size screenSize = MediaQuery.of(context).size;
    final double width = screenSize.width;
    return Scaffold(
      appBar: AppBar(),
      body: Container(
        width: width,
        child: SingleChildScrollView(
          child: FutureBuilder(
              future: getUserProgress(context),
              builder: (context, AsyncSnapshot snapshot) {
                if (snapshot.hasData) {
                  List<String> quizOptions = [
                    this.quizFirstOption,
                    this.quizSecondOption,
                    this.quizThreeOption,
                    this.quizFourOption
                  ];
                  int indexQuizAnswer = -1;
                  indexQuizAnswer =
                      getCorrectAnswer(quizOptions, this.quizCorrectAnswer);

                  print("worldProgresss -> ${snapshot.data[0]}");
```

```
if (indexQuizAnswer != -1)
  return Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Text(
        'Pregunta',
        textAlign: TextAlign.center,
        style: TextStyle(
            fontSize: 25.0,
            fontWeight: FontWeight.bold,
            color: Colors.indigo),
      ),
      SizedBox(height: 8.0),
      Padding(
        padding: const EdgeInsets.only(left: 5, right: 5),
        child: Text(
          this.quizQuestion,
          textAlign: TextAlign.justify,
          style: TextStyle(
              fontSize: 20.0,
              fontWeight: FontWeight.bold,
              color: Colors.black),
        ),
      ),
      SizedBox(height: 16.0),
      Text(
        'Opciones',
        style: TextStyle(
            fontSize: 25.0,
            fontWeight: FontWeight.bold,
            color: Colors.indigo),
      ),
      SizedBox(height: 10.0),
      Padding(
        padding: const EdgeInsets.only(left: 5, right: 5),
        child: ElevatedButton(
          style: ElevatedButton.styleFrom(
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(20.0),
            ),
          ),
          child: Text(
            this.quizFirstOption,
            textAlign: TextAlign.justify,
            style: TextStyle(
                fontSize: 20,
                fontWeight: FontWeight.bold,
                color: Colors.white),
          ),
          onPressed: () {
            if (indexQuizAnswer == 1) {
              QuickAlert.show(
                context: context,
                type: QuickAlertType.success,
                text:
                    'Respuesta␣correcta␣pasaste␣la␣secci n!',
```

```
                onConfirmBtnTap: () async {
                  completedSection(
                      context,
                      this.actualWorld,
                      snapshot.data[0],
                      this.actualSubsection,
                      snapshot.data[1]);
                },
              );
            } else {
              Navigator.pop(context);
              showInSnackBar(
                  "Respuesta incorrecta vuelve a intentarlo",
                  context);
            }
          },
        ),
      ),
      SizedBox(height: 10.0),
      Padding(
        padding: const EdgeInsets.only(left: 5, right: 5),
        child: ElevatedButton(
          style: ElevatedButton.styleFrom(
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(20.0),
            ),
          ),
          child: Text(
            this.quizSecondOption,
            textAlign: TextAlign.justify,
            style: TextStyle(
                fontSize: 20,
                fontWeight: FontWeight.bold,
                color: Colors.white),
          ),
          onPressed: () {
            if (indexQuizAnswer == 2) {
              QuickAlert.show(
                  context: context,
                  type: QuickAlertType.success,
                  text:
                      'Respuesta correcta pasaste la secci n!',
                  onConfirmBtnTap: () async {
                    completedSection(
                        context,
                        this.actualWorld,
                        snapshot.data[0],
                        this.actualSubsection,
                        snapshot.data[1]);
                  });
            } else {
              Navigator.pop(context);
              showInSnackBar(
                  "Respuesta incorrecta vuelve a intentarlo",
                  context);
            }
```

```
      },
    ),
  ),
  SizedBox(height: 10.0),
  Padding(
    padding: const EdgeInsets.only(left: 5, right: 5),
    child: ElevatedButton(
      style: ElevatedButton.styleFrom(
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(20.0),
        ),
      ),
      child: Text(
        this.quizThreeOption,
        textAlign: TextAlign.justify,
        style: TextStyle(
            fontSize: 20,
            fontWeight: FontWeight.bold,
            color: Colors.white),
      ),
      onPressed: () {
        if (indexQuizAnswer == 3) {
          QuickAlert.show(
              context: context,
              type: QuickAlertType.success,
              text:
                  'Respuesta␣correcta␣pasaste␣la␣secci n!',
              onConfirmBtnTap: () async {
                completedSection(
                    context,
                    this.actualWorld,
                    snapshot.data[0],
                    this.actualSubsection,
                    snapshot.data[1]);
              });
        } else {
          Navigator.pop(context);
          showInSnackBar(
              "Respuesta␣incorrecta␣vuelve␣a␣intentarlo",
              context);
        }
      },
    ),
  ),
  SizedBox(height: 10.0),
  Padding(
    padding: const EdgeInsets.only(left: 5, right: 5),
    child: ElevatedButton(
      style: ElevatedButton.styleFrom(
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(20.0),
        ),
      ),
      child: Text(
        this.quizFourOption,
        textAlign: TextAlign.justify,
```

```
                  style: TextStyle(
                      fontSize: 20,
                      fontWeight: FontWeight.bold,
                      color: Colors.white),
                ),
                onPressed: () async {
                  print(indexQuizAnswer);
                  if (indexQuizAnswer == 4) {
                    QuickAlert.show(
                        context: context,
                        type: QuickAlertType.success,
                        text:
                            'Respuesta␣correcta␣pasaste␣la␣secci n!',
                        onConfirmBtnTap: () {
                          completedSection(
                              context,
                              this.actualWorld,
                              snapshot.data[0],
                              this.actualSubsection,
                              snapshot.data[1]);
                        });
                  } else {
                    Navigator.pop(context);
                    showInSnackBar(
                        "Respuesta␣incorrecta␣vuelve␣a␣intentarlo",
                        context);
                  }
                },
              ),
            ),
          ],
        );
      else
        return Padding(
          padding:
              const EdgeInsets.only(left: 5, right: 5, top: 10),
          child: Text(
            "Error␣Cargando␣Pregunta␣intentalo␣de␣nuevo",
            style: TextStyle(
              fontSize: 18.0,
              fontWeight: FontWeight.bold,
            ),
          ),
        );
    } else {
      return Center(
          child: Column(
        children: [
          SizedBox(
            height: 20,
          ),
          Text(
            'Cargando␣...',
            style: TextStyle(
                color: Colors.indigo,
                fontSize: 30,
```

```
                                fontWeight: FontWeight.bold),
                          ),
                          SizedBox(
                            height: 20,
                          ),
                          CircularProgressIndicator(
                            color: Colors.indigo,
                          )
                        ],
                      ));
                  }
                }),
          ),
        ),
      );
  }
}
class CustomWorldSubsectionShowVideo extends StatelessWidget {
  final String videoURL;
  final String videoName;
  const CustomWorldSubsectionShowVideo(
      {Key? key, required this.videoURL, required this.videoName})
      : super(key: key);

  @override
  Widget build(BuildContext context) {
    final Size screenSize = MediaQuery.of(context).size;
    final double width = screenSize.width;
    final double height = screenSize.height;

    final _controller = YoutubePlayerController.fromVideoId(
      videoId: this.videoURL,
      autoPlay: false,
      params: const YoutubePlayerParams(showFullscreenButton: true),
    );

    return Scaffold(
        appBar: AppBar(
          title: Text(
            this.videoName,
            textAlign: TextAlign.justify,
            style: TextStyle(
                color: Colors.white, fontSize: 20, fontWeight: FontWeight.bold),
          ),
        ),
        body: Column(
          children: [
            SingleChildScrollView(
              child: Container(
                  width: width,
                  height: height - 100,
                  child: YoutubePlayer(
                    controller: _controller,
                    aspectRatio: 16 / 9,
                  )),
            ),
```

```
            ],
        ));
    }
}
```

8.28 Anexo AB:

```
class HomeProfessorScreen extends StatefulWidget {
  const HomeProfessorScreen({Key? key}) : super(key: key);

  @override
  State<HomeProfessorScreen> createState() => _HomeProfessorScreenState();
}

class _HomeProfessorScreenState extends State<HomeProfessorScreen> {
  @override
  Widget build(BuildContext context) {
    SystemChrome.setPreferredOrientations(
        [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);
    final size = MediaQuery.of(context).size;

    return WillPopScope(
      onWillPop: () async => false,
      child: Scaffold(
        body: SingleChildScrollView(
          child: SizedBox(
            width: size.width,
            height: size.height,
            child: Stack(children: [
              const ProfessorBackground(),
              CustomProfessorActiveGroupsScreen()
            ]),
          ),
        ),
      ),
    );
  }
}


class CustomProfessorActiveGroupsScreen extends StatelessWidget {
  const CustomProfessorActiveGroupsScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;
    return SingleChildScrollView(
      child: Container(
        alignment: Alignment.center,
        color: Colors.transparent,
        child: Column(
            mainAxisAlignment: MainAxisAlignment.start,
            crossAxisAlignment: CrossAxisAlignment.center,
            children: [
              CustomStudentAppBarHome(),
              Column(
                  mainAxisAlignment: MainAxisAlignment.start,
                  crossAxisAlignment: CrossAxisAlignment.center,
                  children: [
                    SizedBox(
                      height: 20,
```

```
                    ),
                    SingleChildScrollView(
                      child: SizedBox(
                        height: size.height - 200,
                        width: size.width - 50,
                        child: CustomActiveGroupWidget(),
                      ),
                    ),
                  ])
            ]),
      ),
    );
  }
}


class CustomActiveGroupWidget extends StatefulWidget {
  const CustomActiveGroupWidget({Key? key}) : super(key: key);

  @override
  State<CustomActiveGroupWidget> createState() =>
      _CustomActiveGroupWidgetState();
}

class _CustomActiveGroupWidgetState extends State<CustomActiveGroupWidget> {
  Future getGroupsInfo(BuildContext context) async {
    String groups_count = "0";
    List<String> groups_names = [];
    List<String> groups_active_students = [];
    if (kIsWeb) {
      try {
        final _auth = web_auth.FirebaseAuth.instance;
        var user = await _auth.currentUser;
        web_firestore.CollectionReference ref =
            web_firestore.FirebaseFirestore.instance.collection('users');
        await ref
            .doc(user!.uid)
            .get()
            .then((web_firestore.DocumentSnapshot documentSnapshot) {
          groups_count = documentSnapshot['groups_count'].toString();
        });
        if (int.parse(groups_count) != 0) {
          for (int i = 1; i <= int.parse(groups_count); i++) {
            await ref
                .doc(user.uid)
                .collection('students_groups')
                .doc('group_${i}')
                .get()
                .then((web_firestore.DocumentSnapshot documentSnapshot) {
              groups_names.add(documentSnapshot['name']);
              groups_active_students.add(documentSnapshot['students_count']);
            });
          }
        }
      } catch (e) {
        showInSnackBar('Error->␣${e}', context);
```

```
    }
    return [groups_count, groups_names, groups_active_students];
  } else {
    try {
      final FirebaseAuth firebaseAuth = FirebaseAuth.instance;
      var user = await firebaseAuth.getUser();
      var ref = Firestore.instance.collection('users').document(user.id);
      var query = await ref.get();
      groups_count = query['groups_count'].toString();
      if (int.parse(groups_count) != 0) {
        for (int i = 1; i <= int.parse(groups_count); i++) {
          var student_groups =
              ref.collection('students_groups').document('group_${i}');
          var student_query = await student_groups.get();
          groups_names.add(student_query['name']);
          groups_active_students.add(student_query['students_count']);
        }
      }
    } catch (e) {
      showInSnackBar('Error-> ${e}', context);
    }
    return [groups_count, groups_names, groups_active_students];
  }
}

@override
Widget build(BuildContext context) {
  final size = MediaQuery.of(context).size;

  return SingleChildScrollView(
    child: SizedBox(
        width: size.width,
        height: size.height,
        child: FutureBuilder(
            future: getGroupsInfo(context),
            builder: (context, AsyncSnapshot snapshot) {
              if (snapshot.hasData) {
                return Column(
                    mainAxisAlignment: MainAxisAlignment.start,
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      if (int.parse(snapshot.data[0]) == 0)
                        Text('No tienes ning n grupo creado.')
                      else
                        Row(
                          mainAxisAlignment: MainAxisAlignment.spaceBetween,
                          children: [
                            Text(
                              "Grupos Creados",
                              textAlign: TextAlign.start,
                              style: TextStyle(
                                  color: Colors.indigo,
                                  fontWeight: FontWeight.bold,
                                  fontSize: 25),
                            ),
                            ElevatedButton(
```

```
                onPressed: () {
                  Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) =>
                              HomeProfessorScreen()));
                },
                style: ElevatedButton.styleFrom(
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(
                        20.0), // Adjust the radius as needed
                  ),
                ),
                child: Icon(
                  Icons.refresh_sharp,
                  color: Colors.white,
                ),
              ),
            ],
          ),
          if (int.parse(snapshot.data[0]) != 0)
            ListView.separated(
              scrollDirection: Axis.vertical,
              physics: NeverScrollableScrollPhysics(),
              shrinkWrap: true,
              itemCount: int.parse(snapshot.data[0]),
              itemBuilder: (context, index) => GestureDetector(
                onTap: () {
                  Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) =>
                              CustomGroupInfoBody(
                                  groupName: snapshot.data[1]
                                          [index]
                                      .toString(),
                                  studentsCount: snapshot.data[2]
                                          [index]
                                      .toString(),
                                  groupIndex: index.toString())));
                },
                child: Card(
                  child: Column(
                    children: [
                      ListTile(
                        leading: CircleAvatar(
                          backgroundColor: Colors.white,
                          child: FadeInImage(
                            image: AssetImage(
                                'assets/worlds/worlds_subsec
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ions_icons/${globals.subsecti

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□onsIcons[index]}'),
                            placeholder: AssetImage(
                                'assets/icon/loading.gif'),
                          ),
```

```
                                    ),
                                    subtitle: Text(
                                      "Hay ${snapshot.data[2][index]}

␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣estudiante/s activos dentro de este grupo.",
                                      textAlign: TextAlign.justify,
                                      style: TextStyle(
                                          color: Colors.black,
                                          fontWeight: FontWeight.bold,
                                          fontSize: 15),
                                    ),
                                    title: Text(
                                      snapshot.data[1][index],
                                      textAlign: TextAlign.justify,
                                      style: TextStyle(
                                          color: globals.pinkColor,
                                          fontWeight: FontWeight.bold,
                                          fontSize: 20),
                                    ),
                                  )
                                ],
                              ),
                            ),
                          ),
                          separatorBuilder: (BuildContext _, int __) =>
                              const Divider(),
                        ),
                      SizedBox(
                        height: 10,
                      ),
                      Row(
                        mainAxisAlignment: MainAxisAlignment.start,
                        children: [
                          const Text(
                            'Crear Grupo',
                            style: TextStyle(
                                color: Colors.black,
                                fontSize: 15,
                                fontWeight: FontWeight.bold),
                          ),
                          const SizedBox(width: 4),
                          GestureDetector(
                            onTap: () => {
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          CustomProfessorAddGroupsScreen()))
                            },
                            child: const Text(
                              'Toca aqu ',
                              style: TextStyle(
                                color: Colors.indigo,
                                fontSize: 20,
                                fontWeight: FontWeight.bold,
                              ),
```

```
                        ),
                      ),
                    ],
                  ),
                ]);
          } else {
            return Column(
              children: [
                SizedBox(
                  height: size.height / 4,
                ),
                Text(
                  'Cargando ...',
                  style: TextStyle(
                      color: Colors.indigo,
                      fontWeight: FontWeight.bold,
                      fontSize: 36),
                ),
                SizedBox(
                  height: 10,
                ),
                CircularProgressIndicator(
                  color: Colors.indigo,
                )
              ],
            );
          }
        })),
    );
  }
}
```

8.29 Anexo AC:

Colegio	Dirección	Teléfono
Colegio San Bartolomé La Merced	Calle 78 # 8- 31	(571) 345 6789
Colegio Los Nogales	Carrera 11 # 171-52	(571) 123 4567
Gimnasio Moderno	Calle 74 # 9-24	(571) 987 6543
Colegio San Carlos	Calle 140 # 15-50	(571) 234 5678
Colegio Anglo Colombiano	Calle 127 # 7-58	(571) 654 3210
Colegio La Presentación	Carrera 13 # 65-36	(571) 987 6543
Colegio Santa Francisca Romana	Calle 68 # 14-38	(571) 876 5432
Colegio San Jorge de Inglaterra	Calle 140 # 12-27	(571) 345 6789
Colegio El Rosario	Calle 122 # 9B-52	(571) 234 5678
Colegio San Viator	Calle 140 # 7-05	(571) 654 3210
Colegio San Pedro Claver	Calle 97 # 14A-61	(571) 987 6543
Colegio San Francisco Javier	Calle 70 # 9-24	(571) 234 5678
Colegio Santa Ana	Calle 140 # 12-23	(571) 345 6789
Colegio Santa María del Camino	Calle 222 # 7-58	(571) 876 5432
Colegio San Ignacio de Loyola	Calle 83 # 7-26	(571) 654 3210
Colegio La Salle	Carrera 7 # 164-78	(571) 987 6543
Colegio Nueva Granada	Carrera 2 # 21-55	(571) 234 5678
Colegio San Juan Bosco	Carrera 20 # 61-10	(571) 345 6789
Colegio Marymount	Carrera 7 # 170-31	(571) 876 5432
Colegio Los Alpes	Calle 139 # 9-14	(571) 234 5678

Tabla 8.9: Colegios seleccionados para aplicar trabajos futuros dentro del proyecto. Fuente: Autoría propia.