UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE INGENIERÍA DE SISTEMAS

DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE SIMULADORES LABORALES 3D, BASADO EN EJERCITARIOS EL TIEMPO E INFORMACIÓN DIFÍCIL, DENTRO DEL MARCO DEL PROYECTO EDUTECH

Trabajo de titulación previo a la obtención del título de Ingeniera de Sistemas

AUTORA: NARCISA DE JESÚS ARAUJO PÉREZ

TUTORA: ING. PAOLA CRISTINA INGAVÉLEZ GUERRA

Cuenca - Ecuador

2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Narcisa de Jesús Araujo Pérez con documento de identificación N° 0107624637, manifiesto que:

Soy la autora y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 30 de octubre del 2022

Atentamente,

Narcisa de Jesús Araujo Pérez

0107624637

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, Narcisa de Jesús Araujo Pérez con documento de identificación N° 0107624637, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autora del Proyecto técnico: "Diseño, desarrollo e implementación de simuladores laborales 3D, basado en ejercitarios el tiempo e información difícil, dentro del marco del proyecto EduTech", el cual ha sido desarrollado para optar por el título de: Ingeniera de Sistemas, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 30 de octubre del 2022

Atentamente,

Narcisa de Jesús Araujo Pérez

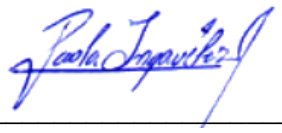
0107624637

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Paola Cristina Ingavélez Guerra con número de identificación N° 1712214616, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación "DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE SIMULADORES LABORALES 3D, BASADO EN EJERCITARIOS EL TIEMPO E INFORMACIÓN DIFÍCIL, DENTRO DEL MARCO DEL PROYECTO EDUTECH", realizado por Narcisa de Jesús Araujo Pérez con documento de identificación N° 0107624637, obteniendo como resultado final el trabajo de titulación bajo la opción de Proyecto técnico que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana.

Cuenca, 30 de octubre del 2022

Atentamente,



Ing. Paola Cristina Ingavélez Guerra

1712214616

DEDICATORIA

Dedico este proyecto de titulación a Dios, mis padres, mi esposo Diego Faican por creer en mí y brindarme su apoyo incondicional durante toda mi carrera universitaria, mis hijos y demás personas que me han apoyado a lo largo de este camino.

Narcisa de Jesús Araujo Pérez

AGRADECIMIENTO

Primero agradezco a Dios por todas las bendiciones recibidas, a mi familia que ha estado conmigo apoyándome durante todo este camino.

Agradezco a mis profesores por toda su paciencia y dedicación, de manera especial al PhD. Vladimir Robles; al Dr. Remigio Hurtado por ser un excelente docente y una gran persona, gracias por su dedicación y apoyo incondicional y a mi tutora y maestra la Ingeniera Paola Ingavélez gracias por su apoyo durante todo este proyecto, agradecida con Dios por haberlos puesto en mi camino.

Narcisa de Jesús Araujo Pérez

Resumen

Este proyecto de titulación se basa en la implementación de simuladores laborales 3D accesibles, los cuales están enfocados a facilitar la inserción laboral de estudiantes universitarios con discapacidad de las universidades socias del proyecto Edutech. Los simuladores tienen como fin fortalecer competencias a través de la interacción con situaciones reales recreadas con el propósito que los estudiantes practiquen y con ello puedan enfrentarse a este tipo de situaciones en el campo laboral.

Por consiguiente, los simuladores estuvieron enfocados al usuario, el proyecto se basó en la metodología "Diseño centrado en el usuario" y se dividió en 3 fases, que parte del análisis de los usuarios; diseño en Unity 3D de escenarios, personaje e interfaz de usuario; desarrollo de programación de lógica de simuladores. Finalmente, se realizó evaluaciones con un grupo de personas, las evaluaciones se enfocaron en medir accesibilidad, usabilidad y navegabilidad.

Los simuladores laborales fueron alojados en el servidor web del proyecto Edutech, por tanto, la información resultante de la interacción de los estudiante-simuladores se almacena en la base de datos del mismo.

Finalmente, la información resultante podrá ser accedida por colaboradores expertos del proyecto Edutech, con el objetivo de medir el nivel de aprendizaje de dichos estudiantes.

Se implementa un módulo de accesibilidad para los simuladores laborales 3D, en donde se contempla todas las discapacidades, para ello se realizó un estudio de las pautas de accesibilidad del contenido web de la norma WCAG 2.0. Finalmente, el módulo podrá ser integrado para los futuros simuladores planteados por parte del proyecto Edutech.

Palabras clave

Accesibilidad Web, usabilidad, recursos educativos, simuladores laborales, Unity 3D, scripts en Unity 3D, lenguajes de programación para Unity 3D, C#, Visual Studio Code, simuladores 3D, ambientes simulados, modelado 3D, WCAG, interfaz de usuario, Edutech.

Abstract

This degree project is based on the implementation of accessible 3D job simulators which are focused on facilitating the employment of university students with disabilities from the partner universities of the Edutech project. These simulators are intended to strengthen skills through interaction with real situations recreated with the purpose that said students practice and with it they can face this type of situations in the labor field.

Since the simulators were focused on the user, the project was based on the "User-Centered Design" methodology where it was divided into 3 phases, starting from the analysis of the users; Unity 3D design of scenarios, character and user interface; development of simulator logic programming and finally evaluations were carried out with a group of people, where said evaluations were focused on measuring accessibility, usability and navigability.

The work simulators were hosted on the web server of the Edutech project, therefore the information resulting from the interaction of the student-simulators is stored in its database.

Finally, the resulting information can be accessed by expert collaborators of the Edutech project with the aim of measuring the level of learning of said students.

An accessibility module is implemented for 3D work simulators, where all disabilities are considered, for this purpose a study of the accessibility guidelines of the web content of the WCAG 2.0 standard was carried out, where finally said module can be integrated for future simulators proposed by the Edutech project.

Keywords

Web accessibility, usability, educational resources, job simulators, Unity 3D, Unity 3D scripts, programming languages for Unity 3D, C#, Visual Studio Code, 3D simulators, simulated environments, 3D modeling, WCAG, user interface, Edutech.

Índice de contenido

Índice de tablas

Índice de figuras

I. Introducción

Para que un sitio web o un juego sea accesible se debe caracterizar por ser fáciles de utilizar por cualquier usuario independientemente si tiene o no alguna discapacidad (Evvano, 2021). Por otro lado, la simulación consiste en desarrollar un escenario que posibilite a los usuarios interactuar con acontecimientos reales recreados con el propósito que practiquen, aprendan, evalúen y adquieran conocimientos de actuaciones humanas (Aguilar Ortega et al, 2021). Las universidades y centros educativos en general, están equipados con diferentes espacios con el fin de brindar la mejor experiencia educativa a sus alumnos. No obstante, ciertas habilidades y destrezas necesitan ser fortalecidas y perfeccionadas mediante prácticas preprofesionales, es decir, que los estudiantes entren en contacto con el campo o rama de especialización, para llevar a la práctica todos los conocimientos que les han sido impartidos en las aulas. Este ir de la teoría a la práctica viene acompañado de una serie de obstáculos que los estudiantes tendrán que hacer frente al momento de realizar dichas prácticas.

La utilización de simuladores en los espacios educativos permite que los estudiantes lleven los conocimientos adquiridos en las aulas a la práctica, sin tener que enfrentar obstáculos como el estrés, el miedo a cometer errores que signifiquen pérdida de tiempo y recursos, el temor de recibir críticas negativas, entre otros.

Teniendo en cuenta lo expuesto anteriormente, se analiza las necesidades de formación y fortalecimiento de competencias para la inserción laboral de estudiantes universitarios con discapacidad (Asistencia tecnológica a la accesibilidad en la Educación Superior Virtual, 2020).

Por ello, se pretende implementar simuladores laborales 3D orientados a accesibilidad, los cuales tendrán como objetivo fortalecer competencias para la inserción laboral de estudiantes universitarios con discapacidad, con el fin que dichos simuladores permitan a los estudiantes interactuar con situaciones reales recreadas, con el fin que practiquen y adquieran conocimientos que se puedan aplicar en el campo laboral (Aguilar Ortega et al., 2021).

II. Problema

De acuerdo con la Organización Internacional del Trabajo (OIT, 2021) las personas con discapacidad representan un aproximado de mil millones de personas, lo que representa un 15% de la población mundial. Las personas con discapacidad experimentan las mayores tasas de desempleo e inactividad económica.

Lo más llamativo con respecto a la situación de estudiantes con discapacidad en el mercado laboral durante el último año, ha sido el veloz crecimiento de su desempleo 10.023 demandantes de empleo con discapacidad menores de 25 años, hoy la cifra asciende hasta 11.637 (El desempleo entre los jóvenes con discapacidad se dispara un 16%, 2021). Todos estos datos nos dan una importante pauta de la problemática que existe actualmente. En esta línea, es importante mencionar que el programa ERASMUS + realizó un informe sobre la asistencia tecnológica a la accesibilidad en la Educación Superior que se enfoca en el desarrollo de competencias para la inclusión laboral de personas con discapacidad con el objetivo de analizar las necesidades de formación y desarrollo de competencias de las personas con discapacidad que faciliten su inclusión laboral a nivel de Europa y Latinoamérica. Por lo que, surge la necesidad de interactuar con varias profesiones para fortalecer competencias laborales en estudiantes universitarios con discapacidad por medio del diseño, desarrollo e implementación de Simuladores Laborales basados en diseño 3D, basados en ejercitarios dentro del marco del proyecto EduTech. Por lo tanto, se trabajó de manera multidisciplinaria (UDA y su carrera de Psicología con UPS y su carrera de Computación) (Asistencia tecnológica a la accesibilidad en la Educación Superior Virtual, 2020).

III. Justificación

Las prácticas implantadas en los procesos formativos en Europa, han contribuido a mejorar la inclusión laboral de personas con discapacidad y la experiencia observada en los principales programas formativos sugieren el predominio de un modelo de colocación-formación-mantenimiento, con una combinación de acciones de formación y apoyo dentro y fuera de los entornos educativos. La supervisión inicial de las tareas laborales del trabajador con discapacidad, por parte de un trabajador laboral en el propio puesto de trabajo fue la estrategia más recurrente, combinada con talleres de formación en competencias laborales y transversales fuera del puesto de trabajo (Asistencia tecnológica a la accesibilidad en la Educación Superior Virtual, 2020). A partir de esta investigación, se desarrollarán simuladores laborales basados en diseño 3D con la herramienta Unity y se diseñarán escenarios que cumplan con todas las expectativas que solicitan los distintos ejercitarios (ejercicios). Los simuladores tendrán como objetivo fortalecer competencias de estudiantes universitarios con discapacidad, de tal manera que ellos puedan ejercer de manera eficaz sus capacidades y tener éxito en el campo laboral prestando servicios o resultados que satisfagan al cliente.

El programa EduTech de Erasmus+ tiene la finalidad de desarrollar herramientas que faciliten la inserción laboral de estudiantes universitarios con discapacidad, el cual requiere la implementación de Simuladores Laborales 3D en la herramienta que incluya las propuestas de diseño y accesibilidad. En este sentido, los simuladores laborales están dirigidos a aquellas personas con discapacidad, los cuales ayudaran a fortalecer competencias como: "Planificación", "manejo del tiempo", "toma de decisiones" y "comunicación efectiva". El proyecto actualmente trabaja con la UDA en el levantamiento de requerimientos en competencias para el sector empresarial de la ciudad de Cuenca.

IV. Grupo Objetivo

Los simuladores laborales serán utilizados por estudiantes universitarios con discapacidad de las universidades socias del proyecto Edutech en coordinación con la Cátedra UNESCO de la UPS y la UDA. Además, el proyecto podrá ser adaptado a futuro en distintas áreas gracias a la relación del GI-IATa y la Cátedra UNESCO de la UPS, con el cual se colabora y está directamente ligado al proyecto EduTech (Asistencia tecnológica a la accesibilidad en la Educación Superior Virtual) de ERASMUS +, lo desarrollado formará parte de los entregables del proyecto.

V. Objetivos

5.1. Objetivo General

Desarrollo de simuladores laborales 3D basado en los ejercitarios: "El Tiempo" e "Información Difícil" dentro del marco del proyecto EduTech.

5.2. Objetivos Específicos

- Estudiar y conocer los principios de la accesibilidad web e interfaces de usuario.
- Diseñar y desarrollar dos ejercitarios basados en diseño 3D considerando accesibilidad.
- Integración de los ejercitarios al módulo de control dentro de la arquitectura del sistema Edutech.
- Diseñar y ejecutar un plan de experimentación que permita determinar el adecuado funcionamiento y utilidad con usuarios con discapacidad.
- Desarrollo de los manuales técnico y de usuario.

VI. Revisión de la literatura

6.1. Antecedentes

Ayón-Parrales y Victores-Pérez (2020) sostienen que la juventud actual se ha desenvuelto en un entorno, en el que la tecnología interviene en casi todos los aspectos de la vida diaria. Por ello, en el campo educativo las estrategias de aprendizaje deben tener como punto de partida la manera de aprender que los estudiantes tienen en la actualidad. De ahí que, es necesario aprovechar el vertiginoso avance de las tecnologías de la información y la comunicación en beneficio de la educación. En ese sentido, la simulación es una herramienta de aprendizaje que hace posible que los estudiantes alcancen un aprendizaje significativo.

En esa misma línea de pensamiento, para el desarrollo de competencias en ciertas disciplinas, a más de abordar la entrega de contenidos se requiere de explicaciones prácticas, con el fin que los estudiantes adquieran conocimientos acordes a los cambios que se dan en su entorno y la simulación es una estrategia didáctica que ayuda en su formación (Ayón-Parrales y Victores-Pérez, 2020). Asimismo, Gaba (2004) sostiene que la simulación es "una técnica, no una tecnología, para reemplazar o amplificar experiencias reales con experiencias guiadas que evocan o replican aspectos sustanciales del mundo real de una manera totalmente interactiva" (p. i2).

Al respecto, de acuerdo con Garzón Romero (2011) al identificar los procesos a simular previa la modelización, hará posible detectar posibles inconvenientes y los aprendizajes que necesitan la participación del aspirante para mejor comprensión y asimilación de los conocimientos y destrezas asociados a tales eventos. Además, Costas Santos (2013) encontró que el uso de simuladores favorece la práctica en condiciones más próximas a la realidad. Asimismo, Zurita López (2015) planteó la hipótesis si emplear simuladores virtuales como recursos didácticos influirá en el fortalecimiento del aprovechamiento escolar de los alumnos, con el resultado que el uso de simuladores tiene incidencia en el fortalecimiento del rendimiento de los alumnos.

Aunado a esto, en el estudio "SIGEM - Simulación de Gestión de Empresas-, un Modelo de Juego de Negocios para el desarrollo de las competencias genéricas universales (CGU) en la Educación Superior en Chile", se demostró que el desarrollo significativo de las

competencias genéricas universales en estudiantes universitarios a través de la utilización del simulador de negocio específico (Bonacic Vargas, 2016). En el mismo sentido, la investigación "Aplicación del simulador de negocios SIMDEF para el fortalecimiento del aprendizaje de contabilidad en los estudiantes universitarios", encontró que el empleo de un simulador SIMDEF ayuda a los alumnos a proceder de acuerdo al caso hipotético planteado. En consecuencia, los estudiantes al enfrentar casos reales serán capaces de resolver problemas o desarrollar un producto (Guerrero Caballero, 2020). Finalmente, Tortarolo (2018) encontró que el Simulador de Navegación es una herramienta esencial en la formación de los estudiantes de la academia naval, lo que hace posible la agilización de los procesos de aprendizaje y un real entrenamiento en ambientes controlados.

6.2. Accesibilidad Web

Para que un sitio web o aplicación sea accesible se debe caracterizar por ser fácil de usar por cualquier usuario independientemente si tiene algún tipo de discapacidad física, neurológica o cognitiva, ya sean estas discapacidades permanentes, temporales o situacionales. Para lograr accesibilidad en el diseño y desarrollo de sitios web o aplicaciones, se debe respetar ciertos parámetros. Las Pautas de accesibilidad al contenido web (WCAG) fueron creadas por la Iniciativa de Accesibilidad Web del Consorcio World Wide Web con el fin de establecer un modelo que pueda ser usado por todos con relación a la accesibilidad del contenido web y de las aplicaciones que beneficie no solo a los individuos, sino también a las organizaciones en general. Al hablar de discapacidad, se hace referencia a cualquier circunstancia que pueda obstaculizar la interacción de los usuarios con cualquier producto digital. En ese sentido, se presentan discapacidades visuales, de la voz, cognitivas, neurológicas y físicas (Evvano, 2021).

La accesibilidad web consiste en que en el diseño y desarrollo de aplicaciones o páginas web se contemplaran los obstáculos que puedan tener los posibles usuarios, con el fin que cualquier persona pueda hacer uso de un producto sin importar sus características y circunstancias particulares. Es decir, que las características de accesibilidad irán en beneficio de todos los usuarios y no solo de personas con discapacidad, como en el caso de adultos mayores que con el transcurso del tiempo se han enfrentado con capacidades disminuidas (Campoverde Molina et al., 2020).

Tim Berners-Lee (1996) sostiene que “El poder de la Web está en su universalidad. El acceso de todos, independientemente de su discapacidad es un aspecto esencial” (Web Accessibility, s.f., párrafo 1).

Antes de la creación del Internet, las personas con discapacidad tenían que enfrentar muchos obstáculos en su peregrinaje en el acceso a la información, por ejemplo los ciegos tenían que obtener una cinta de audio o una versión del libro o artículo deseado en Braille o pedir a alguien que les lea, lo que hacía que dependan de los demás. Actualmente, existen programas que leen textos en voz alta, dejando de lado la dependencia de los ciegos hacia otras personas. Además, las personas con discapacidad motora acceden a la web por medio de programas de asistencia que asignan sus destrezas a su hardware. Asimismo, las personas con problemas auditivos recurren al uso de aplicaciones que proporcionan subtítulos al contenido multimedia (Introduction to Web Accessibility, 2020).

La web está diseñada para que funcione para todas las personas, sin importar su procedencia o sus características físicas. Por lo tanto, la Web cumplirá con su objetivo cuando sea asequible a personas con discapacidad. El impacto de la discapacidad ha sido cambiado en la web al eliminar los obstáculos de comunicación e interacción que las personas hacen frente día a día. Sin embargo, aplicaciones o herramientas mal diseñadas pondrán obstáculos que dejarán por fuera a las personas que estén haciendo uso de la web. Por consiguiente, al momento de diseñar o desarrollar programas o aplicaciones no se debe dejar de lado las necesidades de las personas con discapacidad. Además, la accesibilidad hace posible la inclusión social, puesto que beneficia no sólo a las personas con discapacidad, sino también a otro tipo de personas como adultos mayores y personas que provienen de áreas rurales o de países en vías de desarrollo (Keio, 2018).

Sam Anlas y Stable Rodríguez (2016) sostienen que la accesibilidad web es una métrica de la Ingeniería Web, que trata de elementos relacionados con la codificación y la presentación de información en el diseño y funcionalidad de un sitio web. Por lo que, va a permitir que las personas con algún tipo de discapacidad puedan percibir, entender, navegar e interactuar de forma efectiva en la web, así como crear y aportar contenido. Por lo tanto, su incumplimiento ocasiona discriminación, debido a que no pueden utilizar la información disponible en la web de manera normal.

La accesibilidad web abarca los elementos relacionados con la codificación y la muestra de contenido en el diseño y funcionalidad de un sitio web o aplicación, que hará posible que las personas con discapacidad sean capaces de hacer uso de aplicaciones web de forma idónea, desarrollar y contribuir con contenido. En ese sentido, dejar de lado las características de accesibilidad a la hora de diseñar aplicaciones dejaría de lado a los usuarios que padezcan algún tipo de discapacidad (Sam Anlas y Stable Rodríguez, 2016).

Según Chisholm et al., (2021) al momento de diseñar una aplicación o página web, es importante tener en cuenta que no todas las personas son iguales y que el contexto de cada una puede responder a diferentes realidades. Así por ejemplo, algunos tendrán problemas visuales o auditivos, problemas en el procesamiento de la información, dificultades para leer o comprender lo que leen, limitaciones en cuanto al uso del teclado o mouse, contar con dispositivos con pantallas que permiten solo texto, dispositivos con pantallas pequeñas o tener mala conexión de internet, no hablar fluidamente el idioma del documento, estar en una situación en que hay interferencia de factores externos como ruido del ambiente o exceso de luz, tener versiones anteriores o diferentes del navegador o un sistema operativo desactualizado. Por consiguiente, con el fin que una aplicación o una página web sea accesible es necesario tomar en consideración las necesidades de todos los usuarios potenciales.

El Art. 9 de la "Convención sobre los derechos de las personas con discapacidad" establece que, para que las personas con discapacidad puedan vivir en forma independiente y participar plenamente en todos los aspectos de la vida, deben tener acceso, en igualdad de condiciones, a la información y las comunicaciones (Organización de las Naciones Unidas [ONU], 2007). Asimismo, el Art. 35 de la Ley Orgánica de Comunicación garantiza el derecho que tienen todas las personas al acceso a las tecnologías de la información y comunicación (Asamblea Nacional, 2019).

En Ecuador, la accesibilidad web está regulada en la Norma Técnica Ecuatoriana INEN-ISO/IEC 40500. En este documento, con relación a la accesibilidad web encontramos que las Pautas de Accesibilidad para el Contenido Web (WCAG) 2.0 desarrollan una serie de sugerencias para desarrollar contenido web accesible para más usuarios con discapacidad como por ejemplo problemas en la visión, problemas auditivos, dificultades de aprendizaje, movilidad reducida, mudez, fotosensitividad y variaciones de las anteriores (Instituto Ecuatoriano de Normalización, 2014).

6.2.1. Principios de Accesibilidad Web

La accesibilidad web se apoya en cuatro principios esenciales: 1) Perceptible: que se pueda percibir por medio del tacto, vista u oído. 2) Operable: compatible con dispositivos periféricos de entrada. 3) Comprensible: de fácil comprensión. 4) Robusto: compatible con cualquier clase de navegador tecnologías de asistencia, dispositivos móviles, etc. (Web Accessibility Principles, 2020).

6.2.1.1. Información perceptible e interfaz de usuario

De acuerdo con Abou Zahra (2019) las alternativas de texto son equivalentes para el contenido que no es de texto, expresan la finalidad de una imagen o función para ofrecer una experiencia de usuario equivalente y se pueden exhibir de diferentes formas. Por ejemplo, leer en voz alta a usuarios con limitaciones visuales y para usuarios con problemas de lectura incrementar el tamaño del texto o presentarse en dispositivos braille. Las alternativas de texto facilitan la navegación, ya sea con el teclado o a través del reconocimiento de voz, ya que son como etiquetas para los controles y la funcionalidad. Además, favorecen el reconocimiento de audio, video, archivos en diferentes formatos y aplicaciones intercaladas como parte de un sitio web.

Los usuarios con limitaciones auditivas o visuales requieren opciones, por ejemplo, transcripciones de texto y subtítulos para contenido de audio o video, descripciones de audio para describir detalles visuales significativos de un video, interpretación en lenguaje de señas de contenido de audio. Las transcripciones de texto que contienen información precisa de cualquier contenido auditivo o visual, dotan de accesibilidad a las aplicaciones y facilitan la producción de subtítulos y descripciones de audio (Abou Zahra, 2019).

Según Abou Zahra (2019) la presentación del contenido puede ser cambiada por los usuarios, para ello es necesario que los encabezados, listas, tablas, campos de entrada y estructuras de contenido se encuentren señalados acertadamente, que las instrucciones guarden autonomía con respecto a la presentación y que la configuración de los navegadores y las tecnologías de asistencia ofrezcan opciones que permitan a los usuarios personalizarlos de acuerdo a sus preferencias. Por ejemplo, el contenido puede ser presentado generando resúmenes con el fin de ofrecer a los usuarios la obtención de una descripción general, para que se concentren en ciertas partes con mayor facilidad.

El contenido distinguible es más fácil de ver y escuchar, para ello es necesario que las combinaciones predeterminadas de color frontal y de fondo den el contraste apropiado, que al cambiar el tamaño del texto del espaciado no se pierda información, que el texto se reparta en ventanas y cuando se aumente el tamaño del texto las imágenes de texto cambien de tamaño o puedan ser reemplazadas con texto real o eliminadas, que el volumen del audio que se reproduce en un sitio web pueda ser pausado, detenido o ajustado y que en caso de contener audio de fondo este sea bajo o pueda ser silenciado para que no obstaculice el trabajo de los usuarios. En consecuencia, la información importante se distingue con facilidad atendiendo a las necesidades de los usuarios independientemente a si usan o no tecnologías de asistencia (Abou Zahra, 2019).

6.2.1.2. Navegación e interfaz de usuario operable

Una gran cantidad de usuarios usan exclusivamente el teclado para navegar por la Web, para que el teclado sea accesible debe tener acceso a todas las funciones de la interfaz de usuario, es decir, que toda la funcionalidad del mouse la debe tener el teclado y los navegadores web, las herramientas de creación y otras herramientas deben ser compatibles con el teclado (Abou Zahra, 2019).

Para Abou Zahra (2019) los usuarios leen y utilizan el contenido a su propio ritmo, es decir, unos necesitarán más tiempo que otros. Entre los mecanismos para proporcionar tiempo adecuado tenemos: ajustar los límites de tiempo, detener u ocultar contenido en movimiento, suprimir las interrupciones y posibilidad de autenticarse nuevamente sin perder datos cuando una sesión expire.

Para garantizar accesibilidad a los usuarios, el contenido no debe causar reacciones físicas. La presentación del contenido que contenga parpadeos a ciertas velocidades, animaciones o contenido en movimiento puede causar reacciones molestias y reacciones físicas. En la medida de lo posible, hay que evitar contenido intermitente o usarlo de forma que no represente un riesgo para el usuario, advertir a los usuarios antes de la presentación de contenido intermitente y ofrecer alternativas y mecanismos para detener las animaciones y los usuarios deben tener la posibilidad de pausar o detener la presentación de contenido no esencial que contenga parpadeos a ciertas velocidades (Abou Zahra, 2019).

Abou Zahra (2019) sostiene que para que los usuarios sean capaces de navegar de manera sencilla y encontrar contenido de manera rápida, es necesario que el contenido esté bien

organizado, para ello las páginas deben contar con títulos claros, estar organizadas usando encabezados de sección descriptivos, informar a los usuarios sobre la ubicación de las páginas relevantes dentro de un grupo de páginas relacionadas, que exista opciones para evitar la repetición de bloques de contenido en varias páginas, el teclado debe estar enfocado de forma evidente y el fin de un enlace debe ser obvio. Lo que permitirá, que los usuarios naveguen por las páginas web de acuerdo a sus necesidades y preferencias.

La accesibilidad del contenido se puede lograr a través del uso de diferentes modalidades de entrada como la activación táctil, el reconocimiento de voz y los gestos. A la hora de diseñar, se debe tener en cuenta ciertas particularidades entre la cuales tenemos que los usuarios tengan opciones cuando hagan frente a actividades que requieran movimientos finos, que los componentes estén diseñados para evitar la activación accidental, que las etiquetas admitan la activación por voz, que la funcionalidad que se activa con el movimiento también se active por medio de los componentes de la interfaz de usuario, que los botones, enlaces y otros componentes activos tengan el tamaño adecuado para que sean fácilmente activados al tocarlos (Abou Zahra, 2019).

6.2.1.3. Información comprensible e interfaz de usuario

Abou Zahra (2019) manifiesta que al momento de desarrollar el contenido, es necesario asegurarse que el texto sea legible y claro para la mayor cantidad de usuarios. Para cumplir con este objetivo, el contenido debe identificar el idioma principal de una página web, proporcionar definiciones para palabras inusuales y usar el lenguaje más claro posible, facilitando que el software procese apropiadamente el contenido del texto, lo que beneficiará especialmente a usuarios con discapacidad cognitiva.

El contenido será accesible si aparece y funciona de manera predecible, puesto que una gran cantidad de usuarios se fían de interfaces de usuario de las que ya conocen su funcionamiento y se ofuscan ante un comportamiento inconsistente. Por lo tanto, es necesario que los mecanismos de navegación y los componentes de la interfaz de usuario que se repiten en diferentes páginas siempre estén en el mismo lugar y tengan las mismas etiquetas y que previo a dar lugar a cambios importantes en una página web, se requiera el consentimiento del usuario (Abou Zahra, 2019).

Según Abou Zahra (2019) ciertos formularios e interacciones pueden ser difíciles de usar para muchos usuarios, lo que puede llevar a que cometan errores. Para evitar estos inconvenientes y ayudar a corregirlos, es necesario incluir instrucciones descriptivas, mensajes de error y sugerencias de corrección, suministrar explicaciones de contexto y proveer de opciones de revisar, corregir o deshacer, lo que facilitará que usuarios con limitaciones visuales y auditivas puedan acceder al contenido. Además, las personas que no entiendan, estén confundidas o que por cualquier otro motivo cometieron errores serán capaces de acceder e interactuar adecuadamente al contenido y a los formularios.

6.2.1.4. Contenido robusto e interpretación confiable

Para que el contenido robusto sea compatible con diferentes navegadores, tecnologías de asistencia y otros agentes de usuario, es necesario cerciorarse que el marcado pueda ser interpretado de forma fiable, asignar nombre, función y valor a componentes de interfaz de usuario no comunes. Por consiguiente, facilitará la compatibilidad con los agentes de usuario y en especial que las tecnologías de asistencia procesen el contenido de manera fidedigna y lo exhiban de diferentes maneras (Abou Zahra, 2019).

6.2.2. Pautas de accesibilidad Web

De acuerdo con Caldwell et al., (2008) los principios de accesibilidad web se dividen en 12 pautas que contienen los objetivos principales, que los desarrolladores deben lograr con el fin que sus productos sean lo más accesiblemente posible para usuarios con discapacidad. Pautas que, aunque no son comprobables, ofrecen la base y los objetivos generales para ayudar a entender los criterios de éxito e implementar mejor las técnicas.

En el primer principio tenemos 4 pautas: 1) Alternativas textuales, todo contenido no textual debe ofrecer alternativas de texto que pueda ser cambiado a cualquier formato que se acomode a las necesidades del usuario, como por ejemplo voz o braille. 2) Hay que brindar alternativas para los medios que dependen del tiempo tales como los medios que contengan exclusivamente audio o video pregrabado, subtítulos pregrabados, descripción de audio pregrabado, subtítulos en directo, lenguaje de señas pregrabado, descripción de audio extendida pregrabada, medios alternativos pregrabados y contenido solo audio en directo. 3) El contenido creado debe ser adaptable, es decir, que debe ofrecer al usuario la posibilidad de presentarse de varias formas, por lo que el usuario podrá personalizar el diseño o estructura de una aplicación

o producto sin que eso represente una alteración en su estructura o pierda información. 4) Que el contenido sea distinguible implica que los usuarios podrán con mayor facilidad ver y oír la información presentada, lo que incluye la separación existente entre el primer plano y el plano de fondo. En esta pauta se deben considerar aspectos tales como el uso del color, control del audio, el contraste, tamaño de texto, imágenes de texto, audio de fondo bajo o nulo y presentación visual (Caldwell et al., 2008).

En el segundo principio tenemos 4 pautas: 1) Contenido accesible mediante teclado, implica que todas las funciones podrán ser accedidas al usar el teclado sin que sea necesario medidas de tiempo específicas para pulsaciones de teclas individuales y que el enfoque del teclado se maneje con una sola interfaz de teclado. 2) Proveer a los usuarios suficiente tiempo para interactuar con el contenido, para lo que será necesario ofrecer opciones para ajustar el tiempo a las preferencias de los usuarios, para pausar, ocultar o detener información en movimiento, parpadeante o que se actualiza de manera automática, el tiempo no tendrá un papel preponderante en las actividades a desarrollarse, excepto en actividades sincronizadas o en directo, las interrupciones podrán ser suspendidas por el usuario y no será necesario volver a autenticarse cuando expire la sesión, es decir, el usuario podrá proseguir con el desarrollo de las actividades sin el riesgo de pérdida de datos. 3) El contenido debe ser presentado de una forma que no cause convulsiones, para lo cual es necesario tener en cuenta que las páginas web deben evitar cualquier cosa que destelle más de 3 veces por segundo. 4) El contenido que se presenta debe ser navegable, es decir, debe facilitar al usuario la navegación y la localización del contenido, para lo cual es necesario contar con opciones para omitir bloques que se repiten en varias páginas, el título de las páginas deben indicar su contenido, la navegación de la página web se debe realizar de manera secuencial, lo que preserva el significado y la operatividad, el propósito del enlace se debe determinar por su contexto, debe haber varias opciones para ubicar una página web, los encabezados y las etiquetas deben indicar su propósito, la interfaz de usuario dirigida mediante teclado tiene un enfoque visible, disponibilidad de información sobre ubicación de usuario entre páginas web, identificación del propósito del enlace a partir del texto, para estructurar el contenido se utiliza los títulos de las secciones (Caldwell et al., 2008).

En el tercer principio tenemos 3 pautas. 1) El contenido del texto debe ser legible y comprensible, para lo cual es necesario que el idioma de la página web y del contenido pueda ser determinada mediante programación, que disponga de un mecanismo que permita

identificar palabras inusuales y abreviaturas, disponibilidad de texto alternativo cuando se requiera un nivel de lectura más avanzado que el mínimo requerido para secundaria, disponibilidad de un mecanismo para reconocer palabras con pronunciación ambigua. 2) Las páginas web deben presentarse y funcionar de manera predecible, cuando un componente recibe enfoque no iniciará un cambio de contexto, el cambio de configuración de cualquier componente no cambia el contexto, los mecanismos de navegación que se repiten en varias páginas siguen un orden establecido a menos que el usuario inicie un cambio, componentes con la misma funcionalidad en varias páginas web son identificados de forma consistente, los cambios son activados exclusivamente por el usuario. 3) La asistencia de entrada favorece la identificación de errores, instrucciones son proporcionadas cuando el contenido necesita que el usuario intervenga, al detectarse un error de entrada se proporcionarán opciones de corrección, la prevención de errores en páginas que impliquen compromisos legales o financieros para el usuario, que alteren o eliminen información del controlada por el usuario en bases de datos o que envíen respuestas a pruebas rendidas por el usuario debe ser posible la reversión, la opción de revisión con el fin de corregir errores y la confirmación que permite revisar, corregir errores y confirmar antes de finalizar un proceso de envío de información; disponibilidad de ayuda sensible al contexto y prevención de errores en general que permitirá la reversión, la revisión y confirmación que dará a los usuarios la oportunidad de revisar, corregir y confirmar la información que se proporciona en las páginas web antes de ser enviada (Caldwell et al., 2008).

El cuarto principio comprende una pauta: 1) " Compatible, Maximizar la compatibilidad con las aplicaciones de usuario actuales y futuras, incluyendo las ayudas técnicas" (Caldwell et al., 2008).

6.2.3. ¿Cómo hacer que los juegos sean más accesibles?

Para lograr accesibilidad es necesario diseñar juegos desde otra perspectiva. En el desarrollo de los productos es necesario tener en cuenta las necesidades y particularidades de los posibles usuarios, tratando de eliminar todos los obstáculos que un videojuego pueda presentar en cuanto a su accesibilidad, sin dejar de lado el desafío que es la esencia misma de los videojuegos. Para lograrlo, no es necesario realizar cambios radicales en los productos ya desarrollados, lo indicado sería brindar a los usuarios opciones para que puedan personalizar el juego, incluyendo en los productos características como el texto predictivo o el reconocimiento de voz, productos que comuniquen información en más de una forma, ya sea con texto,

imágenes, colores, sonidos o símbolos. El desarrollo de características para garantizar la accesibilidad en el desarrollo de los videojuegos no significa que se dejará de lado la calidad del mismo. Es importante que, mientras se desarrolle el juego se lo evalúe de manera constante con la ayuda de una lista de comprobación y características de accesibilidad. En el proceso de evaluación del producto, es necesario incluir usuarios con discapacidad que contribuirán desde su experiencia, lo que ayudará a detectar todas las barreras que contiene el producto con el fin de mejorarlo. A la hora de promocionar el producto, es necesario incluir una descripción de las características de accesibilidad del videojuego para que los usuarios sepan lo que están comprando (Hacer que los juegos sean accesibles, 2022).

6.2.4. Principios del Diseño Universal o Diseño para Todos

Los principios de diseño universal fueron desarrollados por Bettye Rose Connell, Mike Jones, Ron Mace, Jim Mueller, Abir Mullick, Elaine Ostroff, Jon Sanford, Ed Steinfeld, Molly Story, y Gregg Vanderheiden (Jofre y Fernández Zalazar, 2019).

Primer principio. Uso equiparable. El diseño debe ser útil para usuarios con diferentes capacidades, proporcionando las mismas formas de uso a los usuarios o equivalentes, sin segregar a los usuarios, brindando garantía y seguridad y siendo llamativo para todos los usuarios (Jofre y Fernández Zalazar, 2019).

Segundo principio. Uso flexible. El diseño debe ser capaz de acomodarse a las preferencias y habilidades de cada usuario, proporcionando opciones en las formas de uso, que su acceso pueda hacerse por cualquiera de las dos manos, facilitando precisión y adaptándose al ritmo del usuario (Jofre y Fernández Zalazar, 2019).

Tercer principio. Simple e intuitivo. El diseño debe ser de fácil comprensión y debe acomodarse a las características individuales del usuario, evitando en lo posible la complejidad, debe responder a las expectativas del usuario, debe adaptarse a una amplia variedad de niveles de alfabetización y destrezas lingüísticas, ofreciendo información de acuerdo con su importancia y suministrando advertencias eficientes y formas de réplica durante y luego de terminada la actividad (Jofre y Fernández Zalazar, 2019).

Cuarto principio. Información perceptible. El diseño presenta de manera eficiente la información que necesita el usuario, teniendo en cuenta sus condiciones ambientales o

habilidades sensoriales. El diseño debe usar diversas formas para exhibir la información importante, ya sea de forma gráfica, verbal o táctil, que proporcione el contraste necesario entre la información fundamental y lo que le rodea, que incremente la comprensibilidad del contenido relevante, distinguiendo los elementos en formatos que puedan ser explicados y facilite la compatibilidad con diferentes dispositivos empleados por usuarios con deficiencias sensoriales (Jofre y Fernández Zalazar, 2019).

Quinto principio. Con tolerancia al error. El diseño disminuye el peligro de acciones involuntarias y sus efectos desfavorables. Por consiguiente, debe disponer de componentes para disminuir los riesgos y errores, presentar alertas sobre riesgos y errores, proveer propiedades inequívocas de suspensión, que desaconsejen maniobras involuntarias en labores que necesiten supervisión (Jofre y Fernández Zalazar, 2019).

Sexto principio. Que exija poco esfuerzo físico. El uso del diseño de manera eficiente y cómoda se debe realizar con el mínimo esfuerzo físico, que posibilite que el cuerpo del usuario esté en una posición neutral, con empleo razonable de energía en la ejecución de acciones que disminuya movimientos repetitivos y esfuerzo físico prolongado (Jofre y Fernández Zalazar, 2019).

Séptimo principio. Tamaño y espacio para el acceso y uso. El diseño debe facilitar tamaño y espacio adecuados para el acceso, alcance, manipulación y uso, considerando el volumen del cuerpo y la posición del usuario, que provea visibilidad clara de los componentes esenciales, adaptándose a cambios de tamaño de la mano o del agarre y suministrando espacio suficiente para el empleo de cualquier tipo de asistencia (Jofre y Fernández Zalazar, 2019).

6.2.4.1. Discapacidad

La discapacidad es un término general que abarca las deficiencias, las limitaciones de la actividad y las restricciones de la participación. Las deficiencias son problemas que afectan a una estructura o función corporal; las limitaciones de la actividad son dificultades para ejecutar acciones o tareas; y las restricciones de la participación son problemas para participar en situaciones vitales. Por consiguiente, la discapacidad es un fenómeno complejo que refleja una interacción entre las características del organismo humano y las características de la sociedad en la que vive (Organización Mundial de la Salud (OMS) y Discapacidad, 2000).

Según la Organización Mundial de la Salud (OMS) el término discapacidad hace referencia a las deficiencias, limitaciones o restricciones a las que se ven enfrentadas las personas al momento de realizar cualquier tipo de actividad, enfrentamiento que se evidenciará en la interacción de las características de una persona con las particularidades de la sociedad en la que se desenvuelve. La discapacidad es una restricción de la capacidad de realizar una actividad de acuerdo a la definición de normalidad establecida para los seres humanos. Por consiguiente, la discapacidad se manifiesta como deficiencia o exceso en el desempeño de una actividad diaria normal. Además, la deficiencia puede ser la perdida de una función psicológica, fisiológica o anatómica como la existencia o aparición de una anomalía, defecto o pérdida producida por un miembro, entre otros. Esta deficiencia, limitación o restricción puede ser temporal o permanente (Qué es Discapacidad, 2018).

Según la OMS la discapacidad se clasifica en: la discapacidad física hace referencia a las alteraciones como efectos secundarios de accidentes o enfermedades, tales como lesión medular o poliomielitis. La discapacidad sensorial, comprende a personas con deficiencias en el sentido de la vista y del oído y personas que presentan dificultades para comunicarse o con el lenguaje. La discapacidad intelectual, comprende la disminución de las funciones mentales y de las funciones motoras. La discapacidad psíquica, se manifiesta en personas con alteraciones neurológicas y trastornos mentales (Qué es Discapacidad, 2018).

6.2.4.1.1. Discapacidad física

La discapacidad física se da cuando el estado físico de una persona le impide de forma permanente e irreversible moverse con toda la funcionalidad de su sistema motriz. Incide en el aparato locomotor, de manera especial en las extremidades puede presentarse como una deficiencia en la movilidad de la musculatura esquelética. Más del 80% de las condiciones de la discapacidad física sobrevienen después del nacimiento debido a accidentes. No obstante, las condiciones de discapacidad pueden desarrollarse durante la etapa de gestación, debido a problemas genéticos o durante el alumbramiento. Con relación a los tipos de discapacidad física, podemos mencionar las que afectan a las extremidades inferiores y superiores, afectación de órganos y vísceras, déficits de las estructuras musculares (La discapacidad física: ¿qué es y qué tipos hay?, 2016).

La discapacidad física puede responder a condiciones genéticas como fibrosis quística, a causas externas como accidentes o puede ser el síntoma de condiciones que afectan al cerebro,

médula espinal o musculatura. Las condiciones más comunes que afectan al cerebro son daño Cerebral Adquirido (DCA) y parálisis cerebral. Los daños en la médula espinal pueden ser lesión en la médula espinal, espina bífida y esclerosis múltiple. En cuanto a las afectaciones en la musculatura está la distrofia muscular (Observatorio de la Discapacidad Física, 2016).

6.2.4.1.2. Discapacidad sensorial

La discapacidad sensorial hace referencia a cualquier condición que afecte a los sentidos tales como ceguera, pérdida de visión, sordera, pérdida de audición, discapacidades olfativas y gustativas, discapacidad somatosensorial, trastornos del equilibrio, entre otros (¿Cuáles son las discapacidades sensoriales?, s/f).

Las discapacidades sensoriales afectan los sentidos como el oído, la vista, el tacto, el olfato y el gusto. La discapacidad sensorial puede responder a accidentes, factores genéticos o ambientales o enfermedades. Ciertas condiciones que causan discapacidad sensorial son trastornos irreversibles, mientras que otras condiciones pueden ser corregidas recurriendo a cirugías. Los tipos de discapacidades sensoriales más comunes e importantes son ceguera y disminución en la visión, sordera y disminución en la audición, sordoceguera y trastorno del procesamiento sensorial (4 Most Common Types of Sensory Disabilities, s/f).

6.2.4.1.3. Discapacidad intelectual

La discapacidad intelectual consiste en un funcionamiento intelectual inferior al promedio y el déficit de habilidades que un individuo necesita en su diario vivir. La discapacidad intelectual puede responder a diversas causas que pueden incluir infecciones en el nacimiento, anomalías cromosómicas, ambientales, metabólicas, nutricionales, tóxicas, traumatismos e inexplicables (Discapacidad intelectual, s/f).

6.2.4.1.4. Discapacidad psíquica

Según Castillero Mimenza (2018) “Hablamos de discapacidad psíquica cuando estamos ante una situación en que se presentan alteraciones de tipo conductual y del comportamiento adaptativo, generalmente derivadas del padecimiento de algún tipo de trastorno mental” "(“Los diferentes tipos de discapacidad”, párrafo 4).

En esa misma línea, discapacidad psíquica es “Aquella que sufre una persona cuando presenta trastornos en su comportamiento de adaptación. Puede ser provocada por diversos

trastornos mentales, como la depresión mayor, la esquizofrenia, el trastorno bipolar, los trastornos de pánico, el trastorno esquizomorfo o el síndrome orgánico" (Discapacidad psíquica, s.f., párrafo 1).

6.3. Principios de recursos educativos

Azcuy y Llull Céspedes (2021) sostienen que el material didáctico es un objeto que asiste en el proceso de enseñanza-aprendizaje. Es un instrumento que transmite contenido con el fin de facilitar el aprendizaje, es una herramienta interactiva y dinámica que presenta elementos multimedia como imágenes, sonidos, videos, animaciones, entre otros. El avance de la tecnología ha permitido que se pueda disponer de una infinidad de materiales digitales educativos, que son utilizados por profesores y alumnos para facilitar, poner en contexto y reforzar los procesos de enseñanza-aprendizaje.

Los recursos educativos son instrumentos que facilitan el proceso de enseñanza-aprendizaje, pueden ser físicos o virtuales y su objetivo es llamar la atención de los estudiantes, adaptarse a sus particularidades, servir de guía al docente y adaptarse a los contenidos. Su trascendencia descansa en la influencia que ejercen sobre los estudiantes al ponerlos en contacto con el objeto de aprendizaje. Con el fin que cumplan su función y sean de utilidad en el proceso de selección de recursos, es necesario tomar en cuenta las necesidades y preferencias de los alumnos. Los recursos didácticos proporcionan información, cumplen objetivos, guían el proceso de enseñanza-aprendizaje, dan contexto a los estudiantes, hacen posible la efectiva comunicación e interacción entre docentes y estudiantes y despiertan el interés de los estudiantes en el proceso de enseñanza aprendizaje. Se clasifican en textos impresos, material audiovisual, tableros didácticos y nuevas tecnologías de información y comunicación (NTIC) (Vargas Murillo, 2017).

Matosas López y Romero Luis (2019) manifiestan que Los avances tecnológicos han tenido un gran impacto en los recursos didácticos y han contribuido a su desarrollo, lo que ha contribuido a la flexibilización de la educación gracias a la gran variedad de opciones en la elaboración de materiales didácticos.

6.4. Simuladores 3D

Según Bonilla Trujillo et al., (2019) los simuladores permiten modelar la realidad en ambientes controlados. Los usuarios pueden explorar e interactuar gradualmente con diferentes escenarios y recibir retroalimentación, para que puedan inferir y deducir, con el fin de lograr un aprendizaje significativo. Las características de los escenarios modelados responderán a las necesidades de formación de los usuarios, con el fin de que tengan vivencias lo más cercanas posible a la realidad que apreciarán a través de sus sentidos. La realidad 3D permite diseñar escenarios modelados lo más parecidos a la realidad.

"La simulación es una técnica numérica que, mediante la modelación de los sistemas reales, permite imitar el comportamiento de las variables y sus interrelaciones, para comprender los procesos internos y modificarlos si es necesario" (Aguilar Juárez y Heredia Alonso, 2013, p. 1).

Un simulador reproduce el comportamiento de un sistema en ciertas condiciones. Para simular la realidad combinan partes mecánicas y virtuales. En el ámbito profesional, los simuladores contribuyen en el proceso de formación de personas que en su vida profesional desempeñarán cargos de responsabilidad y que en caso de cometer errores representarían un riesgo para la vida o la seguridad de otras o la integridad de costosos equipos. Estos programas brindan a los alumnos la oportunidad de experimentar y fortalecer las destrezas que necesitarán para desenvolverse en el ámbito profesional, ya que el cometer errores usando el simulador no pondrá en riesgo nada, ni a nadie (Aguilar Juárez y Heredia Alonso, 2013).

Neri Vela (2017) manifiesta que "la simulación es una técnica que reemplaza y amplifica las experiencias reales, evocando y replicando aspectos sustanciales del mundo real de manera interactiva" (p. 22).

Simular significa representar algo, la simulación surge como producto de la necesidad de reproducir una situación determinada y ver las alternativas que se presenten. Lo que se constituye en recurso didáctico tecnológico con el fin de agrandar el cúmulo de experiencias a través de vivencias controladas, lo cual permitirá a los estudiantes interactuar con situaciones que se viven en la realidad, mediante las cuales podrán aprender a tomar decisiones y manejar imprevistos contribuyendo a reforzar aprendizajes (Vidal Ledo et al., 2019).

De acuerdo con Chaile Alfaro (2015) la función de las aplicaciones de simulación es imitar la ejecución de un proceso del mundo real a través del tiempo y realizar experimentos, para entender su comportamiento y evaluar estrategias de funcionamiento. Para simular, previamente se debe aplicar un modelo que contenga las características más importantes del proceso. El modelo representa al proceso y la simulación es la puesta en marcha del modelo, para apreciar el funcionamiento del sistema de acuerdo al tiempo. En la mayoría de los casos, la simulación se emplea con el fin de explorar diferentes escenarios y mejorar la toma de decisiones. Además, cuenta con herramientas que permiten observar cómo se ejecutan los modelos por medio de las simulaciones y controlar la duración de la simulación. Los simuladores pueden ser de propósito general o específico y basarse en diferentes paradigmas.

La simulación consiste en la utilización de software y hardware, para diseñar aplicaciones que hagan posible imitar procesos similares a la realidad, con los que se pueda experimentar con el fin de observar su funcionamiento o evaluar estrategias de operación. Entre las características de los simuladores tenemos que son herramientas confiables, proporciona un marco para analizar modelos en diferentes tipos de programas, permite la experimentación en un contexto libre de riesgos y la toma de decisiones, hace posible llevar a cabo estudios en diferentes áreas para establecer sus ventajas, desventajas y limitaciones, establece modelos simplificados de la realidad para su manipulación y estudio. Asimismo, hace posible acelerar el proceso de aprendizaje del usuario, su interfaz facilita la relación del usuario con la máquina, se puede conectar en el exterior, es compatible con otras aplicaciones, la biblioteca de objetos puede ser ampliada, elimina riesgos relacionados con el funcionamiento de equipos costosos o con accidentes del usuario mientras interactúa con la realidad y permite la retroalimentación inmediata (Cabero Almenara y Costas, 2016).

El enfoque de aprender haciendo es la razón de ser de la simulación. La interacción de teoría y práctica hace posible aplicar y obtener conocimientos y comprobar la validez de los conceptos. Este enfoque, se basa en que el estudiante experimente de manera directa con la aplicación de una situación de la vida real que despierte su interés y adquiera los conocimientos necesarios para hacerle frente. Los estudiantes tienen que involucrarse de manera personal con la problemática planteada, que contendrá características de escenarios reales que se relacionen directamente con lo que sucede en la vida real. En el campo educativo la simulación reemplaza los contextos reales y al contener las mismas características que contendrían los escenarios

reales y permite la creación de entornos propicios para el aprendizaje situado (Gonzalez y Vallejo, 2021).

En el ámbito educativo, los simuladores son herramientas confiables mediante las cuales los alumnos pueden experimentar sin los riesgos presentes en escenarios reales, son fáciles de manejar al ser versiones simplificadas de la realidad, evita los riesgos relacionados con el manejo de equipos costosos y brindan retroalimentación constante (Gonzalez y Vallejo, 2021).

Entre las ventajas del uso de simuladores en el campo educativo tenemos: permite comprender el funcionamiento y reacción del fenómeno representado, fomenta el aprendizaje por descubrimiento, hace posible comprender conceptos abstractos del problema estudiado, el estudiante puede practicar las veces que considere necesarias puesto que estará en un medio seguro y controlado, promueve el autoaprendizaje, posibilita comprobar ideas previas y construir hipótesis respecto a la situación representada. Los alumnos reaccionarán como si estuvieran ante situaciones reales, entrenándose en la toma de decisiones y formulación de conclusiones, el estudiante tendrá que demostrar sus conocimientos, fomenta el aprendizaje personalizado y la autoevaluación (Gonzalez y Vallejo, 2021), "fomentar la creatividad, ahorra tiempo y dinero." (Cabero Almenara y Costas, 2016)

De acuerdo con Jonassen (1996) los simuladores usados con propósitos educativos son "como “herramientas cognitivas”, ya que aprovechan la capacidad de control del ordenador para amplificar, extender o enriquecer la cognición humana" (Cabero Almenara y Costas, 2016).

Simulación es la representación de algún proceso real a través del tiempo, ya sea manualmente o por computadora. Para simular, es necesario crear un historial artificial de un proceso y con observación obtener inferencias de las características de un proceso real. La simulación por computadora consiste en estudiar una gran gama de modelos de procesos del mundo real por medio de un software diseñado para simular características de un proceso real con frecuencia y a través del tiempo (Gaviño Ortiz et al., 2021).

Según Sandí Delgado y Sanz (2020) la simulación posibilita el diseño de ambientes simulados de aprendizaje, tiene la capacidad para fortalecer habilidades y competencias en diferentes áreas del conocimiento, sin que los estudiantes se vean expuestos a riesgos en su

integridad física o emocional. Posibilita la adquisición de diferentes conocimientos en contextos reales, en los que los estudiantes fortalecen habilidades, se relacionan con otros más fácilmente, refuerzan sus niveles de tolerancia a la frustración, aumentando su motivación, fomenta el logro de objetivos y la resolución de problemas.

La simulación es la representación gráfica de algún proceso a través del tiempo, para lo cual es necesario recopilar los elementos más importantes de un sistema y mediante la observación obtener conclusiones relativas a las características operacionales del sistema real. La simulación por computadora, se realiza por medio del uso de un software capaz de reproducir las características del sistema en el transcurso del tiempo (Díaz Martínez et al., 2018).

6.5. Modelado

Un modelo es una representación conceptual de un proceso, que contiene las características más importantes del fenómeno y se utiliza para analizar sus relaciones e interacciones con el fin de predecir posibles escenarios para dicho fenómeno. Es una representación simplificada de un proceso real, una descripción de entidades y la relación entre ellas. Es un método para reducir y entender la complejidad de los procesos. El modelado consiste en que los procesos y variables de un fenómeno real son representados mediante un conjunto de ecuaciones, con el fin de obtener indicios de su comportamiento de acuerdo a diferentes variables. Los modelos son utilizados para procesar información, mejorar el entendimiento de un proceso real, probar teorías, predecir resultados ante diferentes situaciones, controlar el proceso y obtener resultados anticipados. Los modelos de simulación facilitan el estudio de fenómenos reales (Candelaria Martínez et al., 2011).

Para Herrera y Becerra (2014) en el proceso de simulación, la recolección de la información debe hacerse lo más precisa posible para que los datos obtenidos sean verídicos y no se altere el resultado de la simulación. La etapa inicial es la formulación del problema, el proceso de simulación inicia con la elaboración del plan de estudio. Además, se debe definir los objetivos de la simulación, establecer cómo se va a evaluar su desempeño y los resultados que se esperan obtener. La recolección de información comprende el análisis preliminar del sistema, poniendo atención en sus restricciones, su efectividad, resultados y la determinación de variables y diagramas de flujo que describen el modelo. La siguiente etapa es establecer objetivos, definir el plan general del proyecto y fijar la conceptualización del modelo que será

modificada progresivamente con el fin de introducir mejoras. Luego tenemos la recolección de la información para la construcción del modelo, la implementación del modelo en la computadora, teniendo relevancia el determinar el lenguaje que será usado en la computadora, la etapa de validación en la que se realizan pruebas para comparar el modelo con la realidad, el diseño piloto en el que se establece los posibles escenarios a imitar y con los resultados de los escenarios planteados hay que realizar un análisis con el fin de determinar los escenarios que más se asemejen a la realidad. Así mismo, la etapa de documentación del proyecto debe contener recomendaciones, un informe y los resultados logrados y la puesta en práctica del proyecto.

Modelado es " un método para organizar el conocimiento acumulado a través de la observación o deducido de principios subyacentes. " (Etter, 2018, p. 1) Además, Etter sostiene que “un modelo pretende generalizar y abstraer. Un modelo perfecto es aquel que representa perfectamente la realidad.” (Etter, 2018, p.8)

> Un modelo 3D es una representación matemática de cualquier objeto tridimensional (real o imaginario) en un entorno de software 3D. Para el proceso de modelado 3D es necesario tener en cuenta las propiedades físicas, como texturas, tamaños, triángulos, vértices, etc. y así poder construir cualquier elemento que se desee, desde personajes a objetos, que pueden o no, tener una animación (Vanegas Prieto, 2020, párrafo 1).

Díaz Martínez et al., (2018) sostienen que el primer paso de la simulación es establecer un modelo. Un modelo contiene una descripción física o matemática de un sistema que generalmente hace referencia a un punto determinado de acción en el tiempo. Los modelos sirven para estudiar y establecer la representación de un sistema real de manera abstracta, con el fin de revelar su comportamiento. El modelo generalmente consiste en un conjunto de presunciones relacionadas con la ejecución del proceso, presunciones que se manifiestan como relaciones matemáticas y lógicas entre los elementos principales del proceso. Para modelar el proceso, es necesario determinar sus elementos y presentar una serie de prácticas que contengan un historial del proceso objeto de estudio.

6.6. Unity 3D

Unity 3D es una herramienta creada por Unity Technologies diseñada para la creación de videojuegos. Su desarrollo multiplataforma lo hace compatible con Android, IOS, PC, PlayStation4, entre otros. Unity admite la integración de redes sociales y soporte gráfico. Además, su tienda de activos dispone de modelos incorporados 3D y 2D que contribuirán en el proceso de diseño del producto y de implementos usados para dar nuevas características al motor (Deshpande et al., 2020).

Es una herramienta utilizada para desarrollar videojuegos propiedad de Unity Technologies. Unity es compatible con Windows, Linux y OS X y con plataformas como Playstation3, Playstation4, XBOX 360, XBOX ONE, WII, Android, Windows Phone, Smart TV y con la realidad virtual. Este motor gráfico cuenta con cuatro versiones: Personal, Plus, Pro, Empresa, diferenciándose en los ingresos que generen y las características ofrecidas, todas las versiones cuentan con la capacidad de crear tu videojuego que puede ser exportado a cualquier plataforma. Entre sus características de edición tenemos los GameObjects que son los objetos que interactúan en el juego, la escena son las distintas pantallas que se ven y contiene los objetos del nivel. Unity cuenta con varias ventanas que pueden ser agrupadas como pestañas, entre las que tenemos la cámara de escena, en la que se pueden ver todos los GameObjects, la cámara del juego donde se reproduce lo que se vea en la cámara principal del editor, la ventana de animación que permite crear y modificar las animaciones, la ventana de jerarquía muestra los GameObjects de la escena que se está editando, la ventana del inspector muestra las características y propiedades del objeto seleccionado, la ventana del proyecto muestra la disposición jerárquica de las carpetas y archivos que lo componen y la consola que muestra los posibles errores y mensajes de debug (Cumbreño Juan, 2017).

6.6.1. Interfaz de usuario de Unity

Tiene 5 componentes principales que son: 1. Vista de la escena, que permite navegar y editar visualmente tu escena, puede ser de perspectiva 2D o 3D y tiene herramientas para visualizar que GameObjects tienen un determinado script, quitar y añadir sonidos, etc. 2. Ventana de jerarquía, representa jerárquicamente en forma de texto los GameObjects de la escena, como se relacionan entre si los objetos de la escena. 3. Inspector, hace posible ver y editar las propiedades del objeto seleccionado. Su contenido varía dependiendo de las

propiedades de los objetos. Este componente hace posible agregar y quitar scripts que determinan las características de un GameObject. 4. Ventana del proyecto, contiene los recursos disponibles para usar en un proyecto, permite añadir carpetas, imágenes, sonidos, entre otros. Cualquier recurso importado aparecerá en esta ventana. 5. Barra de herramientas, da acceso a las principales herramientas de trabajo. Las herramientas de la izquierda permiten manipular la escena y los objetos que contiene. Las del centro contienen los controles de play, pausa y stop. En la derecha encuentras la opción para acceder a la cuenta de Unity y servicios de la nube. En un lado encuentras los controles del menú de visibilidad y por último encontrarás los controles de edición del diseño, usado para construir el proyecto y permite generar tu propia plantilla (Interfaz de usuario de Unity, 2018).

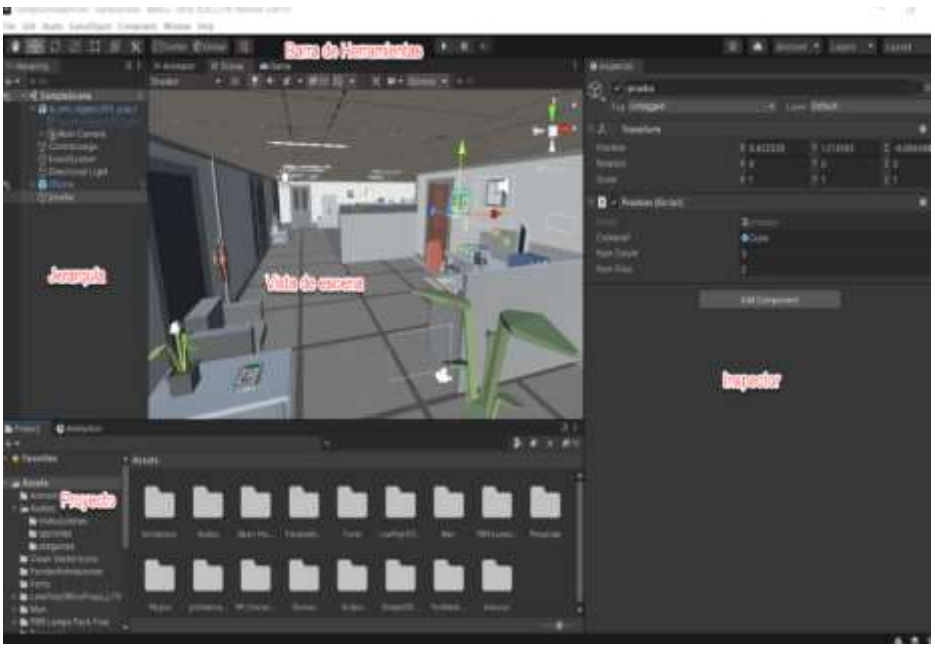


Figura 1. Interfaz de Usuario Unity.

6.6.2. Scripts en Unity 3D

Los componentes adjuntos controlan el comportamiento de los GameObjects. Con Unity es posible la creación de componentes con el uso de scripts, lo que hará posible activar o desactivar eventos en el juego, ajustar en el tiempo las propiedades del componente y dar respuesta a los datos introducidos por el usuario. Unity admite el lenguaje de programación C# que es un lenguaje de programación estándar parecido a Java o C^{++}. Además, lenguajes de programación .Net que puedan agrupar una DLL compatible pueden ser usados con Unity (Creating and Using Scripts, 2018).

Generalmente los scripts son creados dentro de Unity. Para crear un script hay que ir al menú de creación que está en la parte superior derecha del panel del proyecto o dando click en "Assets > Create > C# Script" en el menú principal. El nuevo script se guardará en la carpeta que seleccione y se requerirá dar un nombre a la carpeta, nombre que se utilizará en la creación del texto inicial dentro del archivo (Creating and Using Scripts, 2018).

6.6.3. Forma de un Script

Al dar doble click en recursos de script de Unity se abre una herramienta de edición de texto. El editor predeterminado de Unity es Visual Studio, pero a elección del usuario se puede seleccionar otro editor de texto en el botón de preferencias en la opción herramientas externas (Creating and Using Scripts, 2018).

Un script se conecta con el funcionamiento interno de Unity por medio de una clase derivada de MonoBehaviour. Una clase es como un tipo de plano desarrollado para crear un nuevo tipo de componente que se agregará a un GameObject. Un nuevo script dará una nueva característica a un GameObject. La clase tiene dos funciones definidas que son Update y Start (Creating and Using Scripts, 2018).

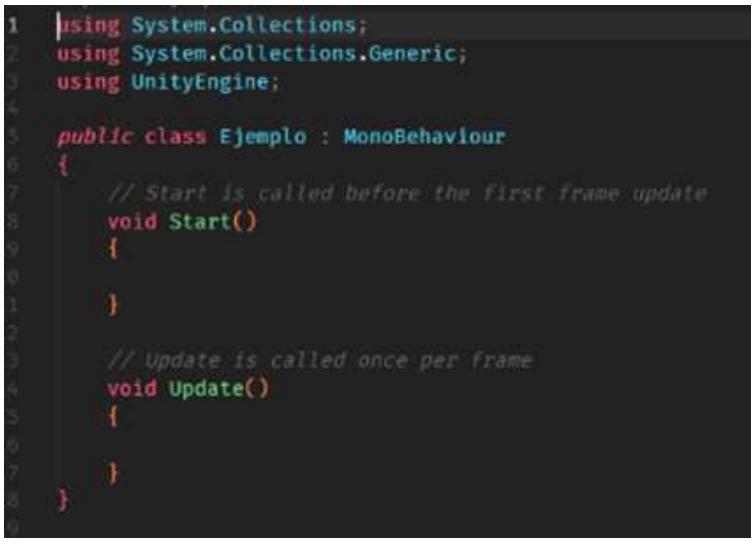


Figura 2. Forma de Script en Unity.

6.6.4. Controlando un GameObject"

Un script define un plano para un componente y el código será activado cuando una instancia del script se agregue a un GameObject. Para añadir un script se puede arrastrar el script a un GameObject ubicado en el panel de jerarquía o al inspector del GameObject seleccionado. El menú del componente contiene un sub menú de scripts donde se encuentran

todos los scripts que tiene el proyecto. Un script luego de ser adjuntado a un GameObject comenzará a funcionar al presionar play e inicie el juego, hecho que se evidencia al agregar un código en la función Start. Debug.Log es un comando que imprime un mensaje en la consola de salida de Unity. Al presionar play aparecerá el mensaje al fondo de la ventana principal del editor de Unity y en la ventana de la consola (Creating and Using Scripts, 2018).

6.6.5. Las Variables y el Inspector

Crear un script es como crear un nuevo tipo de componente que puede ser adjuntado a un GameObject y que tiene características y valores que se pueden editar en el inspector. Con Unity es posible modificar las características de las variables del script, mientras el juego esté corriendo y cuando el modo juego finalice los valores de las variables volverán a como estaban antes del juego (Variables and the Inspector, 2019).

6.6.6. Controlar GameObjects utilizando componentes

El editor de Unity permite realizar cambios en las características del componente por medio del inspector. Los cambios realizados en los componentes modificarán los valores de los GameObjects. Los scripts modifican de manera gradual o en respuesta al input del usuario los valores de los componentes que tienen influencia en el comportamiento de los GameObjects, cambios que al ser llevados a cabo de manera adecuada, permiten la implementación de cualquier tipo de juego (Controlar GameObjects utilizando componentes, 2019).

El acceso a los componentes se lo realiza utilizando la función GetComponent, en la que cogerá una referencia a la instancia del componente con el que se va a trabajar. Es posible tener más de un script personalizado adjuntado a un objeto. Un script mantiene registro de otros objetos, por ello para encontrar un GameObject hay que agregar una variable public GameObject a un script (Controlar GameObjects utilizando componentes, 2019).

Una escena de juego puede utilizar varios GameObjects del mismo tipo, por ello es posible que se necesite encontrarlos mediante un script que los supervise, para lo cual lo mejor es administrar un conjunto de GameObjects haciéndolos todos hijos de un GameObject principal. Además, los GameObjects individuales pueden ser ubicados utilizando la función GameObject.Find. Los objetos pueden ser localizados por su etiqueta utilizando las funciones GameObject.FindWithTag y GameObject.FindGameObjectsWithTag (Controlar GameObjects utilizando componentes, 2019).

6.6.7. Funciones de Evento

Un script tiene el control cuando es necesario que se ejecuten ciertas funciones, cuando la función termina de ejecutarse el control vuelve a Unity. Estas funciones son las funciones de eventos que Unity las activa para dar respuesta a eventos que ocurren mientras se desarrolla el juego. Entre los eventos importantes tenemos: Eventos Actualizados Regularmente, Eventos de Inicialización, Eventos GUI y Eventos de Física (Event Functions (Funciones de Evento), 2019).

6.6.8. Clases importantes: Administrador del Tiempo y Framerate

La clase de tiempo de Unity contiene características básicas importantes que posibilitan el trabajo con valores que involucren el tiempo en un proyecto. La función Update permite monitorear inputs y eventos regulados desde un script. Al manejar acciones basadas en tiempo hay que tener presente que el Framerate del juego y el periodo de tiempo entre llamados de funciones Update no es constante (Important Classes - Time and Framerate Management, 2019).

6.6.9. Namespaces

Dependiendo de la complejidad del proyecto y el número de scripts puede presentarse un conflicto entre los nombres de las clases de los scripts. Este inconveniente puede ser evitado con el uso de una convención de nombre o renombrando los scripts cada vez que un conflicto se presente. No obstante, en ocasiones habrá varias clases de scripts con nombres en conflicto. La propiedad namespaces del lenguaje C# resuelve el problema de los nombres en conflicto. Un namespace es una colección de clases a las que se hace referencia usando un prefijo determinado en el nombre de la clase (Namespaces, 2019).

6.6.10. Atributos

Los atributos permiten señalar un comportamiento especial a una clase, propiedad o función de un script al colocarlos sobre ellos, es decir, funcionan como marcadores. Los atributos que ofrece Unity los podemos encontrar en AddComponentMenu, CallbackOrderAttribute y en las bibliotecas (Attributes, 2019).

6.6.11. Excepción con referencia nula

Una excepción de referencia nula ocurre cuando se intenta acceder a una variable de referencia que no hace referencia a ningún objeto. El tiempo de ejecución emitirá una excepción de referencia nula cuando el usuario intente acceder a un objeto con variable nula. Las variables que han sido referenciadas en C# y JavaScript en concepto se asemejan a apuntadores en C y C++. Cuando no se configura una variable antes de usarla el usuario obtendrá una excepción de referencia nula en el código. Otra razón para la aparición de excepción de referencia nula es usar una variable que debía ser inicializada en el inspector. Una manera para solucionar este error es usar los bloques try/catch. Por consiguiente, una excepción de referencia nula se presenta cuando el código del script intenta usar una variable que no está referenciada a ningún objeto. El mensaje de error le indicará donde se suscita el problema. Para evitar este tipo de errores, es necesario escribir un código que se encargue de identificar estos problemas antes de acceder a un objeto o también se puede usar los bloques try/catch (Null Reference Exceptions, 2019).

6.6.12. Eventos de Unity

La función eventos de Unity permite que la devolución de llamada del usuario una vez activada continúe desde la edición hasta la ejecución, sin que sea necesaria una programación adicional o configurar una secuencia de comandos. Los eventos de Unity permiten devoluciones de llamada persistentes y que responden a contenido, sistemas de desacoplamiento y eventos de llamada preconfigurados. Los eventos de Unity se ejecutan desde el código y se pueden adjuntar a cualquier MonoBehaviour, luego de lo cual aparecerá en el inspector y permite adjuntar llamadas persistentes. Los eventos de Unity contienen limitaciones, entre las cuales tenemos que contienen referencias al elemento objetivo, para evitar que el objetivo sea identificado como basura. En lo relacionado a los eventos genéricos de Unity, un evento de Unity contenido en un MonoBehaviour a una función nula de forma dinámica (UnityEvents, 2019).

6.7. Lenguajes de programación para Unity 3D

Unity admite el lenguaje de programación C# que es un lenguaje de programación estándar parecido a Java o C^{++}. Además, lenguajes de programación .Net que puedan agrupar una DLL compatible pueden ser usados con Unity (Creating and Using Scripts, 2018).

6.7.1. Lenguaje de programación C#

C# es un lenguaje orientado a objetos y con seguridad de tipos que permite crear aplicaciones seguras y sólidas que se ejecutan en .NET. C# tiene sus orígenes en la familia de lenguajes C. C# proporciona construcciones de lenguaje, se trata de un lenguaje natural que permite crear y usar componentes de software. C# incluye características que admiten nuevas cargas de trabajo y prácticas de diseño de software emergentes (Paseo por el lenguaje C#, 2022).

C# es un lenguaje de programación de alto nivel, orientado a objetos, de propósito general, similar a Java y C++ y en alguna medida a Delphi, VB.NET y C. Los programas de C# constan de un conjunto de definiciones en clases que contienen métodos y los métodos contienen la lógica del programa. C# es desarrollado por Microsoft. C# es distribuido junto con el entorno Common Language Runtime (CLR) en el que se ejecuta. CLR forma parte de la plataforma .NET Framework. Los programas C# generalmente se ejecutan en MS Windows, pero .NET Framework y CLR también son compatibles con teléfonos móviles y otros dispositivos portátiles basados en Windows. No obstante, la implementación gratuita de .NET Framework Mono permite que programas C# puedan ejecutarse en Linux, FreeBSD, iOS, Android, MacOS X y otros sistemas operativos (Nakov y Kolev, 2013).

- **Principales Características**

Sencillez: C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Modernidad: C# incorpora elementos que son muy útiles para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular. Orientado a objetos: C# es un lenguaje orientado a objetos, puesto que no admite ni funciones ni variables globales, ya que todo el código y datos han de definirse dentro de definiciones de tipos de datos. Gestión automática de memoria: Tiene a su disposición el recolector de basura del CLR, por lo que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción using. Instrucciones seguras: Para evitar errores muy comunes, en C# se han impuesto una serie de restricciones en el uso de las instrucciones de control más comunes. Extensibilidad de operadores: C# permite redefinir el significado de la mayoría de los operadores cuando se apliquen a diferentes tipos de objetos, lo que facilita la legibilidad del código y conseguir que los nuevos tipos de datos básicos que se definan a través de las estructuras estén al mismo nivel

que los básicos predefinidos en el lenguaje. Eficiente: C# permite saltarse las restricciones del código manipulando objetos a través de punteros (Briones Quiroz et al., 2016).

6.7.2. NET Framework

Es un entorno de ejecución administrado para Windows. Los componentes principales de .NET Framework son: Common Language Runtime (CLR) consistente en un motor de ejecución cuya función es controlar las aplicaciones en ejecución y la biblioteca de clases de .NET Framework que ofrece una biblioteca de código probado y reutilizable. .NET Framework ofrece a las aplicaciones servicios de administración de la memoria, sistema de tipos comunes, biblioteca de clases extensa, marcos y tecnologías de desarrollo, interoperabilidad de lenguajes, compatibilidad de versiones, ejecución en paralelo y compatibilidad con múltiples versiones (Introducción a .NET Framework, 2022).

6.7.3. Visual Studio Code IDE

Un *entorno de desarrollo integrado* (IDE) es una aplicación con características que favorecen a los elementos del desarrollo de software. El IDE de Visual Studio es un panel de inicio que permite editar, depurar y compilar código y publicar una aplicación. Visual Studio incluye editor, depurador estándar, compiladores, herramientas de finalización de código, diseñadores gráficos, entre otros. Las ventanas principales de Visual Studio son Explorador de soluciones, en la que se puede ver y administrar los archivos de código, la ventana del editor que muestra el contenido del archivo y permite editar código o diseñar una interfaz de usuario y Git Changes que permite hacer un seguimiento de los elementos de trabajo y compartir código. Visual Studio está disponible para Windows y Mac y existen 3 ediciones que son: Community, Professional y Enterprise. (Le damos la bienvenida al IDE de Visual Studio, 2022)

6.8. Arquitectura REST

"Transferencia de estado representacional (REST) es un estilo arquitectónico basado en el protocolo HTTP que describe un estilo de diseño de navegación orientado a los recursos" (Dodero y Ghiglione 2008, pp. 91), "para cada recurso, una URI debe ser proporcionado para permitir que las aplicaciones del cliente naveguen desde él a otros recursos " (Dodero y Ghiglione 2008, pp. 91).

Los estilos arquitectónicos de servicios que siguen los principios REST favorecen un acoplamiento flojo de componentes. Según el estilo ReST todo en la Web es un recurso. Un recurso tiene diferentes representaciones y se direcciona por un nombre o URI. Los recursos respaldan verbos simples que se envían al URI como un método HTTP para realizar la función CRUD solicitada en el recurso identificado por id. Con excepción de la solicitud POST, el verbo se omite, ya que está implícito en el método HTTP. El formato de representación es parte del encabezado HTTP y se puede especificar en el URI. Los principios del diseño arquitectónico ReST permiten una integración abierta de API para acceder a recursos web. Un servicio web ReSTful acepta solicitudes en diferentes URI, cada una asignada a un recurso determinado. El estilo de acceso a los servicios de aprendizaje basado en REST proporciona una integración débilmente acoplada entre las actividades de LD y los proveedores de servicios de aprendizaje, favoreciendo el requisito de desacoplamiento de la interfaz de usuario (Dodero y Ghiglione 2008).

REST es un estilo arquitectónico para desarrollar servicios web que se caracteriza por su simplicidad, ya que para alcanzar sus metas se basa en los sistemas y características existentes del Protocolo de transferencia de hipertexto (HTTP) de Internet (¿Qué es REST (Transferencia de Estado REpresentacional)?, 2020).

6.8.1. Ventajas

Entre las ventajas de utilizar REST tenemos: Basado en recursos, las interacciones REST se basan en construcciones que son familiares para usuarios acostumbrados a usar HTTP. Comunicación, las interacciones basadas en REST comunican su estado a través de códigos de estado HTTP numéricos. Familiaridad, la mayoría de los desarrolladores ya están familiarizados con los elementos clave de la arquitectura REST. Independiente del idioma, los desarrolladores pueden emplear cualquier lenguaje que utilice HTTP para realizar solicitudes basadas en web. Penetrante, la popularidad de REST se debe a su uso generalizado en implementaciones del lado del servidor y del cliente. API web, cuando se trata de almacenamiento en caché los servicios RESTful emplean mecanismos HTTP efectivos. (¿Qué es REST (Transferencia de Estado REpresentacional)?, 2020).

6.8.2. Desventajas

Entre las desventajas tenemos: Arquitectura, limitaciones debido al diseño de su arquitectura. Apátrida aplicaciones, HTTP no almacena información basada en el estado entre los ciclos de solicitud y respuesta, por lo que el usuario tiene que llevar a cabo tareas de administración del estado dificultando así la implementación de actualizaciones del servidor. Definición, REST carece de una implementación de referencia clara que indique si un diseño específico se puede definir como RESTful, lo que provoca incertidumbre con relación a si determinada API web se ajusta a los principios basados en REST. Captación excesiva o insuficiente de datos, los servicios RESTful mezclan la información importante conjuntamente con datos inutilizables, dando como resultado un incremento del tiempo que el usuario se tarda a devolver datos útiles (¿Qué es REST (Transferencia de Estado REpresentacional)?, 2020).

La arquitectura REST define principios a seguir para implementar aplicaciones web. Para su implementación REST se basa en estándares como HTTP y XML. Los servicios REST caracterizan por lo siguiente: Las operaciones más importantes son GET, POST, PUT y DELETE. La utilización de hipermedias hace posible que el usuario explore los recursos de una API REST mediante enlaces HTML. La respuesta del API REST son JSON sin importar el lenguaje empleado (Alamilla Hernández et al., 2021).

6.8.3. Restricciones que definen a las API RESTful

La arquitectura REST está conformada por clientes, servidores y recursos que administran las peticiones o solicitudes con HTTP. El contenido de los clientes nunca se almacena en el servidor. Debido a que las interacciones son cliente-servidor, el almacenamiento en el caché elimina la necesidad y consumo de memoria. Las interacciones cliente-servidor logran mediaciones por capas adicionales. Al transferir código ejecutable los servidores pueden extender las funciones de un cliente. Define una interfaz genérica para administrar las interacciones que se dan entre el cliente y el servidor de manera uniforme, lo cual separa y simplifica la arquitectura. Para el diseño de las API RESTful, es necesario incluir los siguientes aspectos: identificación de recursos en las solicitudes, administración de recursos mediante representaciones, mensajes autodescriptivos e hipermedios que es el motor del estado de la aplicación (Alamilla Hernández et al., 2021).

6.9. WebGL

WebGL es una interfaz de programación para crear gráficos en el navegador que ofrece una serie de llamadas a una librería especializada en renderizado 3D y basada en OpenGL ES 2.0. WebGL puede ser accedido con lenguajes compatibles con DOM, Javascript básicamente, ya que trabaja bajo el elemento Canvas de HTML5 (Crespo Amigo, 2012).

WebGL es una interfaz de programación que hace posible desarrollar aplicaciones interactivas que generan imágenes por ordenador en páginas web. Al ser independiente del sistema operativo y del sistema gráfico de ventanas, necesita únicamente de un ordenador que le soporte (Ribelles y López, 2019).

WebGL permite que el contenido web utilice una API basada en OpenGL ES 2.0 para llevar a cabo la representación 2D y 3D en un elemento Canvas HTML en los navegadores que lo soporten sin el uso de plug-ins. WebGL consiste en código de control escrito en JavaScript y código de efectos especiales (código shader) que se ejecuta en la unidad de procesamiento gráfico de una computadora (GPU). Los elementos WebGL se pueden mezclar con otros elementos HTML y componerse con otras partes de la página o el fondo de la misma (Primeros Pasos con WebGL, 2022).

6.9.1. Unity WebGL

Unity publica contenido semejante a programas JavaScript que utiliza tecnologías HTML5 por medio del uso de la función de compilación WebGL y ejecuta contenido en la red mediante el empleo de la API de renderización WebGL. Para iniciar con el desarrollo y evaluación del contenido para WebGL hay que ir a la ventana de Build Player, seleccionar la opción de construcción WebGL y luego Build & Run (Getting started with WebGL development, 2021).

El código del contenido debe estar en JavaScript para poderlo ejecutar en WebGL. La cadena de herramientas del compilador emscripten se usa para hacer una compilación cruzada del código de ejecución de Unity, es decir, cambiar de C and C++ a asm.js JavaScript que permite a los motores de JavaScript llevar a cabo una compilación anticipada de código asm.js

en un eficiente código nativo. IL2CPP es empleado para cambiar los scripts que están en lenguaje C# a JavaScript (Getting started with WebGL development, 2021).

Unity WebGL es compatible con los principales navegadores en los ordenadores, pero no en dispositivos móviles. Las restricciones en la plataforma dan lugar a que no todas las funciones de Unity estén disponibles en las compilaciones de WebGL. JavaScript no es compatible con los subprocesos, no es posible depurar las compilaciones de WebGL en Visual Studio, por seguridad los navegadores no permiten acceder directamente a las tomas de IP para redes, la API de gráficos WebGL es semejante a OpenGL ES 2.0 y 3.0, las compilaciones de WebGL utilizan un backend personalizado para Audio que es compatible únicamente con la funcionalidad básica de audio, WebGL no permite la generación dinámica de código usando System.Reflection.Emit (Getting started with WebGL development, 2021).

6.10. Diseño Centrado en el Usuario

6.10.1. Principios

1- **Diseño para los usuarios y sus tareas**: Los sistemas informáticos existen para ayudar a que los usuarios realicen sus actividades. Es un sistema cuyo objetivo es brindar apoyo al usuario en el desenvolvimiento de sus actividades. Por consiguiente, las necesidades, el contexto, las características y preferencias de los usuarios deben ser tomadas en cuenta en el desarrollo y diseño de aplicaciones informáticas (Galeano, 2008).

2- **Consistencia:** Con el objetivo de facilitar el uso del sistema por parte de los usuarios, es necesario que el comportamiento y apariencia de los elementos comunes de la interfaz y de cajas de diálogo sean en la medida de lo posible constantes. Es decir, que los diseños se deben basar en los elementos ya existentes del sistema operativo, ya que una interfaz cuyo diseño diste mucho del resto del sistema traerá como consecuencia que el usuario se vea en la necesidad de invertir tiempo y esfuerzo para saber cómo usarlo (Galeano, 2008).

3- **Diálogo simple y natural:** La comunicación entre el usuario y el sistema debe desarrollarse sujetándose al orden que la actividad a desarrollarse requiera. La información proporcionada al usuario debe ser la estrictamente necesaria para el desenvolvimiento de la tarea, ya que información innecesaria sólo complicaría la interacción del usuario con el programa o aplicación. Por consiguiente, las notificaciones e instrucciones deben contener

un idioma neutral y un vocabulario de acuerdo a los posibles usuarios, con términos que tengan siempre el mismo significado (Galeano, 2008).

4- **Reducción del esfuerzo mental del usuario**: La interacción del usuario con la computadora debe desarrollarse de la manera más simple posible, es decir, debe tener la posibilidad de centrar completamente su atención en la actividad que está desarrollando, para lo cual es necesario simplificar las tareas que se desarrollan con cierta frecuencia, que sea capaz de acceder con facilidad a las instrucciones sobre el uso del sistema, sin verse forzado a tener que recordar información sobre el uso del sistema (Galeano, 2008).

5- **Proporcionar realimentación adecuada:** Es necesario poner a disposición del usuario información de si están o no desarrollando las actividades de manera correcta, lo cual se logra con indicadores de progreso, que advierten al usuario que la actividad requiere cierto tiempo para completarse (Galeano, 2008).

6- **Proporcionar mecanismos de navegación adecuados:** Los usuarios deben contar con información que le sirva como rutas claras para las secciones a la que tiene que acceder con el fin de llevar a cabo una tarea. Así tenemos por ejemplo: títulos en las ventanas, indicadores de localización, numeración en las páginas, mapa de navegación, una descripción, una historia de las secciones visitadas, etc. Cuando el usuario acceda a determinadas secciones por error se le debe proporcionar opciones para salir de ahí, como por ejemplo un botón de cancelar o deshacer, que le permita volver a la actividad que estaba desarrollando sin tener que pasar por cuadros de diálogo largos que le quitarán tiempo (Galeano, 2008).

7- **Dejar que el usuario dirija la navegación**: El usuario debe tener la posibilidad de elegir la información que requiere para la consecución de una determinada tarea, para lo cual es necesario que el sistema proporcione información clara que facilite el desempeño de las actividades, más cuando son tareas que se realizan de manera cotidiana (Galeano, 2008).

8- **Presentar información clara:** La información que se proporciona al usuario debe ser presentada en un esquema que le permita distinguir los diferentes contenidos, teniendo precaución de incluir sólo la información necesaria, colocando los contenidos por medio de ventanas con el fin de facilitar la comprensión de la interfaz (Galeano, 2008).

9- **El sistema debe ser amigable:** En la utilización del sistema el usuario no debe tener que verse en la necesidad de recurrir a manuales de usuario. Por lo tanto, la información que se brinde al usuario debe ser coherente con la tarea que se encuentra desarrollando, la

información que se brinde en línea debe responder a las necesidades del usuario, de acuerdo a la tarea que desarrolle, con el contexto de la ventana en la que está interactuando y enlistando el procedimiento a seguir con el fin de desarrollar con éxito su tarea (Galeano, 2008).

10- **Reducir el número de errores:** Con el fin de reducir los errores, es necesario que el usuario tome el camino que le ayudará a conseguir su objetivo. Si la actividad lo permite se puede poner a disposición del usuario opciones a las que tenga que dar respuesta con el fin de evitar errores, opciones que no deben entorpecer el normal desenvolvimiento de sus actividades. En ese sentido, al ingresar información la misma debe ser validada de forma inmediata, las notificaciones de errores deben ser presentadas en segundo plano y en lenguaje claro, usando tecnicismos solo en caso que con ellos se resuelva el problema o sugieran la solución (Galeano, 2008).

En el proceso de desarrollo de un producto el número de fases de DCU estará subordinado a la complejidad y tipo de aplicación o página web a desarrollar (Martín Fernández y Hassan Montero, 2004).

6.10.2. Fases de Diseño Centrado en el Usuario

6.10.2.1. Estudio y Modelado del Usuario

En esta fase lo primero es determinar y especificar los usuarios a los que estará dirigido el producto, ya que se debe tener en cuenta que no se puede incluir a cualquier tipo de usuario. Es decir, que determinados usuarios deben ser excluidos, ya que las características propias del producto así lo requieren. Por consiguiente, es necesario establecer las características y necesidades del potencial usuario, sin dejar de lado en razón de su condición física, su edad o que tienen limitaciones por el contexto en el que se encuentran. La razón para incluir usuarios con discapacidad o con limitaciones es que aunque sus necesidades de acceso difieren del usuario promedio, tienen las mismas metas que un usuario promedio (Martín Fernández y Hassan Montero, 2004).

La información del usuario se puede obtener mediante estudios directos, ya sea encuestas, estudios de campo, entrevistas y por medio del estudio de artículos científicos de las necesidades, obstáculos que enfrentan las personas con discapacidad. Los datos recogidos en la investigación al potencial usuario nos servirán como guía para iniciar el desarrollo del producto,

proceso que debe enfocarse en las necesidades y preferencias del potencial usuario (Martín Fernández y Hassan Montero, 2004).

La información debe ser sintetizada y organizada con el fin que resulte beneficiosa y pueda ser utilizada de manera apropiada por el diseñador. Por consiguiente, se debe determinar perfiles de usuarios, para los cuales se diseñará un producto encaminado a la satisfacción de las necesidades específicas de los grupos de usuarios (Martín Fernández y Hassan Montero, 2004).

Los perfiles de usuarios se determinan de acuerdo a las características que los usuarios tengan en común, como por ejemplo personas con limitaciones de acceso a la información. En este sentido, el modelado cumplirá con los requisitos de accesibilidad si se definen clases de usuario atendiendo a sus limitaciones de acceso, quienes serán agrupados entre personas que tengan necesidades parecidas. En esta técnica de modelado de usuario hay que tener en cuenta que la clasificación puede no ser viable, cuando la población a la que va dirigida es muy grande y diversa; al presentarse esta dificultad se aconseja emplear el enfoque "persona" de Alan Cooper. Esta técnica se basa en determinar modelos de usuarios que simbolizan formas de comportamiento, carencias y propósitos; modelos a los que se les denomina personas son definiciones descriptivas de usuarios que tienen un nombre, edad y sexo imaginarios, a quienes se les asigna un modelo con las características de los usuarios a quienes está dirigida la aplicación. A más de los prototipos de usuarios, se deben establecer escenarios en los que se detalle la circunstancia en la que el programa será utilizado, para así poner en contexto la interacción de la "persona" con el producto (Martín Fernández y Hassan Montero, 2004).

Estos modelos de usuarios deben ser máximo de 3 o 4 clases y no representan a la totalidad de usuarios del producto. Esta técnica tiene como propósito permitir que el diseñador diseñe centrándose en el usuario que puede ser considerado real, ya que aunque es ficticio sus características son tomadas de usuarios reales (Martín Fernández y Hassan Montero, 2004).

De estos modelos de usuarios ficticios hay que distinguir entre usuarios primarios y secundarios, los primarios no tienen la capacidad de utilizar una interfaz diseñada para otra persona a la que se le llama de tipo secundario. En ese sentido, hay que establecer modelos de usuario con algún tipo de discapacidad o situaciones en las que el producto no sea del todo accesible. Lo que ayudará a que el diseñador desarrolle su producto teniendo en mente un

modelo de usuario real, siendo capaz de discernir qué tipo de diseño será accesible para los potenciales usuarios y cuál no (Martín Fernández y Hassan Montero, 2004).

El hecho que un único interfaz atienda las necesidades de acceso de todos los usuarios o sea adaptable a cada tipo de usuario dependerá de si para el diseño del producto se empleó el enfoque "persona-scenario" o un modelado de usuario íntegro, con los usuarios potenciales totalmente clasificados (Martín Fernández y Hassan Montero, 2004).

El interfaz adaptable al usuario necesita representar electrónicamente la información completa y detallada del usuario, para lo cual se necesita establecer y dar a conocer un modelado que encasille y especifique de manera integral a los potenciales usuarios. Al contrario, el enfoque "persona-scenario" es el indicado cuando se tiene como propósito desarrollar un único interfaz web. Por consiguiente, el modelado total de la audiencia será un medio beneficioso en la toma de decisiones al momento de diseñar un producto y para la definición del conocimiento del usuario que tiene el sistema (Martín Fernández y Hassan Montero, 2004).

6.10.2.2. Diseño y Prototipado

Teniendo en cuenta que un único interfaz no es capaz de satisfacer las necesidades y preferencias de todos los usuarios de la manera que lo harían varios diseños de acuerdo al tipo de usuario, algunos sitios web proporcionan versiones alternativas como la versión solo texto o distinto idioma, lo que aumenta la cobertura de usuarios, pero presenta inconvenientes con la gestión y actualización de contenidos. En ese sentido, aunque es más complicado diseñar interfaces que se adapten a las necesidades del usuario es más efectivo que diseñar múltiples interfaces. Para lo cual, hay que representar el conocimiento que el sistema tiene sobre las diversas clases de usuario, las reglas de inferencia y las circunstancias que determinen cómo y cuándo será conveniente adaptar la estructura y presentación de los contenidos de forma dinámica (Martín Fernández y Hassan Montero, 2004).

Para asegurar que una aplicación sea adaptable se necesita separar el contenido, la presentación, la estructura lógica y la interacción. Ante la existencia de diferentes tipos de usuarios, se necesita que la información se presente de diferentes maneras y de la adaptación de los elementos de interacción. En una página web el enlace constituye el elemento básico para la interacción y como su comportamiento denota simplicidad no necesita de dicha abstracción. Por otro lado, elementos complejos como las listas de selección se presentan y

responden de manera diferente a la interacción dependiendo de las características, capacidades necesidades y preferencias de los usuarios. Implementar una interfaz adaptable dinámicamente u ofrecer múltiples interfaces no resulta apropiado para todos los casos, ya que para proyectos pequeños o que estén dirigidos a un público excesivamente heterogéneo, lo adecuado es desarrollar un único diseño que procure atender las necesidades de interacción de todos los usuarios (Martín Fernández y Hassan Montero, 2004).

En el Diseño Centrado en el Usuario la fase de diseño consiste en las decisiones predictivas e interpretativas, que el diseñador toma basándose en la información obtenida del usuario en la fase de observación y a los resultados logrados en la etapa de evaluación, con relación a cuál será la opción más apropiada de diseño que garantice al usuario la usabilidad y accesibilidad de una página o aplicación web (Martín Fernández y Hassan Montero, 2004).

La evaluación de las decisiones de diseño desde el inicio del desarrollo de una aplicación y el rediseño del producto de acuerdo a los resultados obtenidos en la evaluación, se realiza mediante la aplicación de la técnica conocida como prototipado. El prototipado consiste en la elaboración de modelos de interfaz de una aplicación o página web. Los prototipos son reproducciones, no son fases iniciales de una aplicación y una vez usados dejan de ser útiles, es decir, no son parte de la aplicación (Martín Fernández y Hassan Montero, 2004).

Las diferentes clases de prototipado se clasifican en base al nivel de funcionalidad reproducida. El prototipado horizontal reproduce una parte significativa del aspecto de la aplicación, pero estos modelos no tienen el respaldo de la funcionalidad que tendrá el producto final. El prototipado vertical reproduce el aspecto visual únicamente de una parte de la aplicación, pero la parte reproducida tiene la misma funcionalidad que tendrá la aplicación terminada (Martín Fernández y Hassan Montero, 2004).

Según el grado de calidad del prototipo se clasifican en: El modelo será de alta fidelidad si su aspecto guarda gran similitud con la interfaz del producto en su estado final. Los prototipos serán de baja fidelidad cuando su aspecto guarde una gran diferencia con la interfaz de la aplicación finalizada, con la ventaja que su desarrollo no implica costes elevados (Martín Fernández y Hassan Montero, 2004).

Roger R. Hall citado por Hassan Montero y Martín Fernández (2004) establece que para elegir el tipo de prototipado más conveniente hay que considerar los siguientes factores: el tipo de producto a diseñar, la información que se necesita, la etapa del diseño en que se empleará el modelo, los recursos con los que se cuenta y el probable coste de realizar un mal diseño.

Según la perspectiva del diseño inclusivo el tipo de prototipado será elegido de acuerdo a las necesidades de acceso que se procura encontrar. Así, por ejemplo para detectar errores relacionados con aspectos cognitivos como comprensión, lenguaje y contexto cultural, es viable utilizar prototipados de baja calidad, ya que su bajo coste permite analizar el nivel de trascendencia del rotulado y la comprensibilidad de la interrelación de los contenidos determinados en la distribución de los elementos en la interfaz. Por otro lado, en el caso de problemas relacionados con aspectos físicos, ya sean visuales, auditivos o motrices es imperativo el empleo del prototipado de alta fidelidad (Martín Fernández y Hassan Montero, 2004).

6.10.2.3. Evaluación de la Accesibilidad

En la etapa de evaluación se validan las decisiones de diseño y se descubren errores metodológicos, de ahí su trascendencia en el desarrollo de productos. La evaluación debe ser implementada desde las fases iniciales del desarrollo de la aplicación, es decir, que se debe realizar la evaluación de los prototipos y del producto final, ya que si no se evalúa desde el inicio, es decir, desde las fases iniciales del desarrollo del producto, los errores que pudieron ser detectados en etapas tempranas tendrán un efecto negativo en el costo del proyecto. La evaluación de la accesibilidad puede ser realizada mediante una serie de técnicas y métodos. La selección de las técnicas y métodos a emplear en la evaluación y el número de evaluaciones a efectuarse se lleva a cabo en base al nivel de profundidad solicitado y del presupuesto con el que se cuente (Martín Fernández y Hassan Montero, 2004).

- **Evaluación Heurística**

La evaluación heurística es un método en el que un grupo reducido de expertos, en base a su experiencia, en reconocidos principios de usabilidad y en guías evalúan de forma independiente el producto contrastando finalmente los resultados con el resto de evaluadores. Este método permite descubrir problemas de uso y de acceso (Martín Fernández y Hassan Montero, 2004).

- **Evaluación con Usuarios**

Esta técnica de evaluación de usabilidad consiste en pruebas llevadas a cabo en "laboratorio", con el fin de observar a un grupo de usuarios ejecutar varias actividades que se les ha solicitado, para luego analizar si en el desarrollo de la evaluación se presentaron problemas de usabilidad. Las pruebas de usabilidad con usuarios con determinadas variaciones, se pueden aplicar en la evaluación de la accesibilidad (Martín Fernández y Hassan Montero, 2004).

- **Evaluación con Usuarios Discapacitados**

Cuando las evaluaciones involucran usuarios con discapacidad se presentan ciertas trabas como por ejemplo, al momento de reclutar participantes o el coste económico de la evaluación. Asimismo, considerando varios tipos de discapacidad, no es factible cubrir en la prueba cada una de las clases de discapacidad. No obstante, este tipo de evaluaciones ayudan a encontrar dificultades de usabilidad-accesibilidad que no son halladas con el uso de otras técnicas de evaluación (Martín Fernández y Hassan Montero, 2004).

VII. Marco metodológico

El siguiente proyecto está enfocado en desarrollar simuladores laborales 3D orientados en accesibilidad, los cuales serán utilizados por estudiantes universitarios con discapacidad de las universidades socias del proyecto Edutech en coordinación con la Cátedra UNESCO de la UPS y la UDA. Los simuladores tienen como objetivo fortalecer competencias de los estudiantes con la finalidad de facilitar su inserción laboral. Para el levantamiento de los requerimientos del proyecto, se llevó a cabo una reunión con los colaboradores del proyecto Edutech (*Mónica Rodas Catedrática de la UDA y el grupo de investigación de la Cátedra UNESCO de la UPS*) y se realizó un análisis de los ejercitarios(ejercicios) propuestos, los simuladores tienen como objetivo ayudar a fortalecer competencias laborales, de esta manera facilitar la inclusión laboral y educativa de estudiantes universitarios con discapacidad.

Para el desarrollo del proyecto se utilizó elementos teóricos y técnicos en los que se plantea una arquitectura, implementación de los simuladores laborales, además de las evaluaciones respectivas para medir accesibilidad, usabilidad y navegabilidad. Además, se trabajó con la herramienta Unity debido a que su plataforma es intuitiva, permite diseñar

entornos de calidad, permite acceder a versiones gratuitas, brinda una documentación completa. Asset Store (tienda de Unity) es una tienda de objetos, los cuales ayudaron en el diseño del proyecto, es una multiplataforma que permite almacenar los simuladores laborales en la nube; gracias a los múltiples beneficios que tiene Unity para el desarrollo de videojuegos, se diseña una barra de preferencias en la que el estudiante podrá adaptar el simulador laboral a sus capacidades y necesidades. Por lo que, se implementa un módulo de accesibilidad, el cual se podrá integrar en futuros simuladores laborales. Por consiguiente, el desarrollo de del módulo se basó en las pautas de accesibilidad del contenido web para lograr una interacción y una interfaz de usuario accesible.

La importancia de desarrollar simuladores laborales 3D se justifica en que la simulación posibilita el diseño de ambientes simulados de aprendizaje, además de la capacidad de fortalecer habilidades y competencias en diferentes áreas del conocimiento sin que los estudiantes se vean expuestos a riesgos en su integridad física o emocional. Por otro lado, la simulación posibilita la adquisición de diferentes conocimientos en contextos reales, en los que los estudiantes fortalecen habilidades, se relacionan con otros más fácilmente, refuerzan sus niveles de tolerancia a la frustración frente a situaciones reales, aumentando su motivación, fomenta el logro de objetivos y la resolución de problemas (Sandí Delgado y Sanz, 2020) en un proceso real, los simuladores laborales se desarrollaron en Unity gracias a que es una plataforma intuitiva, la cual permite diseñar entornos de calidad.

A partir de los objetivos indicados y teniendo en cuenta que los simuladores están orientados al usuario, se toma como base los principios de la metodología, la cual nos dice que una interfaz debe ser consistente con el objetivo de facilitar al usuario el uso del sistema. Por lo que, es necesario que la apariencia y funcionalidad de los componentes de la interfaz y componentes de texto sean en lo posible constantes, de esta manera evitar que el usuario tenga que invertir tiempo y esfuerzo para entender cómo usarlo. Por consiguiente, el desarrollo de este proyecto se divide en cuatro fases basándose en la metodología DCU: fase 1 (Estudio y modelado del usuario), fase 2 (Diseño), fase 3 (Desarrollo o prototipado) y fase 4 (evaluación) esta última fase se detalla en el apartado de resultados.

7.1. Fases

7.1.1. Fase 1: Estudio y modelado del usuario

En esta fase se realizó una investigación con el objetivo de comprender el estado del arte de los simuladores en el ámbito educativo, implantación y la importancia. Por otro lado, se realizó un análisis de los usuarios en el que se identificaron los requerimientos y funcionalidades que se espera de un simulador laboral 3D.

A partir de los ejercitarios (ejercicios) planteados, se identificaron los respectivos requerimientos funcionales y no funcionales que deben alcanzar los simuladores laborales 3D.

7.1.1.1. Ejercitario El Tiempo

Ejercitario(ejercicio) propuesto por colaboradores del proyecto Edutech.

Nombre de Ejercitario	El tiempo
Instrucción principal	Planificar y jerarquizar las actividades descritas
Principales Competencias medidas	Planificación y manejo del tiempo
Categoría del ejercicio	Interacciones uno a uno
Duración	
Variaciones	Presentaciones con uso de material audiovisual
Instrucciones para el participante	Cualquier actividad que realicemos necesita tiempo. Aprender también requiere de un periodo planeado y organizado con ese fin. Los ejercicios que te proponemos tienen el objetivo que desarrolles tu habilidad para administrar tu tiempo y puedas con ello lograr tus propósitos personales, así como aprender. Pasos de la administración del tiempo Administrar el tiempo es realizar las actividades en el momento adecuado para alcanzar nuestros propósitos. Para administrar tu tiempo puedes seguir los siguientes pasos:

	1. Definir las actividades que vas a realizar. 2. Jerarquizar esas actividades. 3. Delimitar el tiempo de cada actividad 4. Organizar un horario. Usted es jefe de un departamento de talento humano de la empresa XYZ, y le han delegado realizar las siguientes actividades.

Actividades	**Jerarquizar**	**Delimitar tiempo**	**Jerarquizar**	**Delimitar tiempo**
Implementar los procesos de selección, contratación e inducción de personal.			2	4 días
Planificar y ejecutar los procesos de evaluación de desempeño, elaborar, presupuestar y ejecutar el plan anual de capacitación y desarrollo.			3	7 días
Realizar la actualización del manual de perfiles de cargo.			1	5 días
Ejecutar planes de comunicación interna.			5	1 día
Mantener actualizadas las bases de datos del área: selección, información de la capacitación y la formación de cada uno del personal de la empresa.			4	2 días
Coordinar externamente con diferentes instituciones programas que aporten a la gestión del talento humano.			6	10 días

Rúbrica de calificación

Competencia	**Definición**	**Indicadores de desempeño**	**Calificación**
Capacidad de priorización y toma de decisiones	Determinar eficazmente metas y especificar las etapas, acciones, plazos y recursos requeridos para el logro de los objetivos.	a.- Jerarquiza y asigna tiempo a las actividades de manera coherente y organiza sin	**a. 100%**

	Ordenar y sistematizar los períodos de tiempo destinados a la realización de actividades de modo que permitan el logro de sus metas, objetivos y proyectos. Implica cumplir el mayor número de responsabilidades con calidad respetando el propio tiempo y el de los demás.	improvisaciones de última hora. b.- Jerarquiza y asigna tiempo a las actividades de manera coherente y organiza con pocos errores. c.- Jerarquiza y asigna tiempo a las actividades de manera coherente y organizadamente, cometiendo la mitad de errores. d.- Jerarquiza y asigna tiempo a las actividades solo a pocas actividades.	b.- 75% c.-50% d.- 25%

Tabla 1. Ejercitario El Tiempo.

7.1.1.2. Ejercitario Información Difícil

Ejercitario(ejercicio) propuesto por colaboradores del proyecto Edutech.

Nombre de Ejercitario	Información difícil
Instrucción principal	Usted es el rector de una institución educativa privada, que por la crisis ocasionada por la Pandemia Covid 19, se ha visto muy afectada por la reducción de alumnos. Para mantenerse en pie se ha decidido realizar un recorte de personal.
Principales Competencias medidas	Toma de decisiones y comunicación efectiva
Categoría del ejercicio	Interacción individual
Duración	Usualmente corta, no más de 10 minutos
Variaciones	Presentaciones con uso de material audiovisual
Instrucciones al Participante:	Usted es el rector de una institución educativa privada, por la crisis ocasionada con la Pandemia Covid 19, se ha visto muy afectada por la reducción de alumnos ya que los padres de familia no pueden seguir pagando las pensiones. Con el fin de mantenerse en pie, se

	ha decidido realizar un recorte de personal y prescindir de algunos colaboradores.
Actividades	**Opciones**
Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.	a.- Análisis de evaluaciones del desempeño. (los que tengan menos de la media en las evaluaciones del desempeño). **Respuesta correcta** b.- Cargas familiares (los que tengan menos cargas familiares) c.- Por antigüedad (los que estén menos años en la organización) d.- Aleatoriamente (los nombres que salgan sorteados)
Ahora elija el discurso que usted utilizaría para comunicar a su compañero:	a.-Esta es una situación difícil, debo comunicarle que por la situación económica que estamos atravesando hemos decidido prescindir de sus servicios, esperamos comprenda nuestra decisión. **Respuesta correcta** b.-Como parte del respeto que le tenemos a nuestros colaboradores, hemos analizado varios criterios que nos permitan seleccionar de manera objetiva a las personas que deben retirase de la institución, lamentamos decirles que por la situación económica que atravesamos debemos prescindir de sus servicios. d.- De manera comedida debo indicarle que cuando mejore nuestra situación financiera buscaremos su reincorporación a la institución, por ahora espero entienda que usted está despedido. e.- Los procesos de separación son difíciles de asumir, por ello espero que pronto entienda que su salida de la institución solo se debe a un recorte de personal por una situación difícil que pasa nuestra organización.

Competencia	**Definición**	**Indicadores de desempeño**	**Calificación**
Toma de decisiones		Cumple con elegir la opción a 100% de la competencia al elegir la opción.	**a. 100%**
Comunicación efectiva		Cumple con elegir la opción a 100% de la competencia al elegir la opción.	**a.- 100%**

Tabla 2. Ejercitario Información Difícil.

7.1.1.3. Requerimientos Funcionales de los simuladores laborales

A partir de los ejercitarios planteados, se define una lista de requerimientos funcionales los cuales están orientados a accesibilidad y usabilidad que se desea presentar a los estudiantes

con discapacidad. En la siguiente tabla se lista los requerimientos funcionales del proyecto. Junto al grupo de investigación de la Cátedra UNESCO se establecieron los requerimientos funcionales de los simuladores laborales. Después de un análisis del problema del proyecto, se inició con la indagación de los principales objetivos que debían cumplir los simuladores laborales, en los que al final se definió los requerimientos funcionales que cumplan los aspectos principales de los simuladores.

Requerimientos	**Descripción**	**Observación**
Estructura del simulador laboral	Cada simulador laboral se estructura de la siguiente manera: ✓ Escenario ambientado ✓ Personaje ✓ Interfaz de Inicio ✓ Interfaz introductoria ✓ Interfaz de actividades ✓ Interfaz de calificación obtenida ✓ Interfaz de retroalimentación ✓ Interfaz de barra de preferencias Dicha estructura se define para el desarrollo de futuros simuladores.	**Prioridad Alta**
Escenario	El escenario debe estar ambientado hacia el tipo de ejercitario.	**Prioridad Alta**
Animación Personaje y cámara	El personaje camina por el escenario hacia su puesto de trabajo, una vez llegue visualizará un botón de inicio, una vez se visualice dicho botón de inicio, el personaje podrá mover la cámara y así podrá mover la cámara.	**Prioridad Media**
Introducción al ejercitario	La información presentada debe ser amigable al participante, debido a que dicha información es fundamental para la interacción con las actividades que se le presentará.	**Prioridad Alta**
Opciones de botones	El participante debe poder interactuar con botones que le permita: ***salir*** *(salir del simulador),* ***preferencias*** *(accedo al panel de preferencias),* ***atrás*** *(retroceder entre los botones) y* ***siguiente*** *(avanzar hacia adelante).*	**Prioridad Alta**

Descripción de audios	Se deberá reproducir audios de todos los textos del simulador laboral.	**Prioridad Alta**
Navegación	Se iluminará o cambiará de color el botón cuando se posicione en dicho botón, el puntero del mouse o al presionar el teclado.	**Prioridad Alta**
Barra de Preferencias	El estudiante tendrá la opción de cambiar las preferencias del simulador laboral con el que esté interactuando en cualquier momento, tales como: *tipos de fuente, contraste de pantalla, tamaño de fuente, activar o desactivar audios, etc.*	**Prioridad Alta**
Interacción normal	El estudiante podrá interactuar con el simulador laboral mediante el mouse.	**Prioridad Alta**
Interacción accesible	El estudiante podrá interactuar con el simulador laboral mediante el teclado.	**Prioridad Alta**
Actividades de tipo numérico	En este tipo de actividades se requiere que el estudiante ingrese un numero como respuesta. En donde se le indica el rango de valores a ingresar permitidos. De cada una de las actividades: de debe jerarquizar ingresando valores de 1 a 6 en donde dichos valores no deben repetirse, para el ingreso de valores que corresponden a delimitar tiempos se debe ingresar valores de 1 a 11.	**Prioridad Alta**
Actividades de tipo Selección	En este tipo de actividades se plantea un concepto o una descripción al estudiante y este a su vez deberá seleccionar de varias opciones la opción correcta.	**Prioridad Alta**
Calificación obtenida	El participante podrá visualizar la calificación obtenida con respecto a las actividades del simulador laboral con las que haya interactuado.	**Prioridad Alta**
Retroalimentación final	Cuando el estudiante finalice la interacción con las actividades presentadas recibirá una retroalimentación con respecto al simulador laboral con el que ha interactuado.	**Prioridad Alta**

Tabla 3. Requerimientos Funcionales de Simuladores Laborales.

7.1.1.4. Requerimientos no funcionales de los simuladores laborales

Requerimientos	**Descripción**	**Observación**

Estructura del simulador laboral	Cada simulador laboral se estructura de la siguiente manera: ✓ Escenario ambientado ✓ Personaje ✓ Interfaz de Inicio ✓ Interfaz introductoria ✓ Interfaz de actividades ✓ Interfaz de calificación obtenida ✓ Interfaz de retroalimentación ✓ Interfaz de barra de preferencias Dicha estructura se define para el desarrollo de futuros simuladores.	**Prioridad Alta**
Escenario	El escenario debe estar ambientado hacia el tipo de ejercitario.	**Prioridad Alta**
Personaje	El personaje ha utilizar para los simuladores deberá ser el mismo que se está utilizando con el resto de los simuladores del proyecto Edutech. Dicho personaje se muestra al inicio del simulador laboral, quien simula las tareas de un rector de una institución	**Prioridad Media**
Cámara	El simulador cuenta con una vista 3D, la cámara inicia en tercera persona hasta que aparezca en pantalla el botón para empezar la interacción, una vez que aparece el botón de empezar, la cámara se mueve en primera persona.	**Prioridad Media**
Audios	Salida de audios correspondientes al texto que se muestre en pantalla, es decir, cuando el puntero del mouse o al presionar el teclado se posicione en alguno de los botones se reproducirá el audio correspondiente a ese texto.	**Prioridad Alta**
Interfaz	La interfaz debe ocupar la mayor parte de la pantalla.	**Prioridad Alta**
Botones de accionamiento	Los botones principales con los que se estructura cada simulador laboral son 4: ***salir*** *(salir del simulador),* ***preferencias*** *(accedo al panel de preferencias),* ***atrás*** *(retroceder entre los botones) y* ***siguiente*** *(avanzar hacia adelante).*	
Actividades	Cada vez que el participante acceda a las actividades del simulador laboral deberán mostrarse de manera aleatoria dichas actividades. Se debe medir el tiempo que le toma resolver al participante cada actividad y guardarlos en la base de datos.	**Prioridad Alta**

Tiempo empleado	- Se debe medir el tiempo de inicio y tiempo de finalización del simulador laboral. - Se deberá medir el tiempo que le tome al participante interactuar con cada una de las actividades planteadas, en el que los tiempos deberán ser registrados en la base de datos del servidor web.	**Prioridad Alta**
Consumo de servicios	Se debe recibir como parámetro (la URL del simulador laboral alojado en la web, ID del usuario quien va a interactuar con el simulador laboral y el número de ejercitario al que corresponde), se debe hacer una petición al servidor mediante el método GET, con el cual se obtiene el ID de participante.	**Prioridad Alta**
Registro de datos	Se debe guardar en la base de datos mediante un archivo Json (respuesta ingresada, tiempo de respuesta, numero de pregunta, correo, número de ejercitario, tiempo de inicio, tiempo que finaliza y la fecha que realiza la actividad)	**Prioridad Alta**
Ejecución en WebGL	Los simuladores laborales deberán ser construidos a WebGL, los cuales serán alojados en el servidor web del grupo de investigación Edutech, para posteriormente ser ejecutados desde la web.	**Prioridad Alta**

Tabla 4. Requerimientos no funcionales de simuladores laborales.

7.1.1.5. Requerimientos de Herramientas

Para iniciar en base a los objetivos y funcionalidades que presentan los simuladores se define las herramientas a utilizar para el diseño e implementación, en los que una vez estén en su versión final deberán ser alojados en el servidor web del grupo de investigación Edutech.

Herramienta	**Descripción**
Unity	Motor de videojuegos multiplataforma, en su versión 2020.3.21f1
Asset Store	Tienda de recursos propia de Unity, para el modelado de los escenarios, personaje e interfaz de usuario.
C#	Lenguaje de programación, en su versión v1.25.0
Visual Studio Code	Editor de código, en su versión 1.70.0

Tabla 5. Herramientas para el desarrollo de los simuladores laborales.

7.1.2. Fase 2: Diseño

En esta segunda fase se realiza el diseño de escenarios de los simuladores laborales, el diseño de interfaces mostrará el contenido referente al simulador, distribución de botones en cada pantalla (interfaz). Esta fase es muy importante, ya que se define los elementos que contiene cada simulador laboral: los objetos, la ambientación, se desarrolla la historia de la situación real a simular, se define el personaje, el contexto, el modelo de interfaz de usuario, etc.

Para alcanzar un diseño accesible se debe tener en cuenta los principios de diseño universal, accesibilidad y usabilidad que al cumplirlos hacen de un sitio web accesible. Para la integración de un módulo de accesibilidad en los simuladores laborales, se desarrollaron en base a las pautas de accesibilidad del contenido web WCAG 2.0.

7.1.2.1. Modelado de Simuladores Laborales

Puesto que, la simulación consiste en el uso de hardware y software para diseñar aplicaciones para recrear los procesos reales y experimentar con ellos para entender su comportamiento y funcionamiento. En el ámbito de la educación, la simulación se enfoca en representar la realidad, replicar una situación real lo más cercana a la realidad con el fin de permitir al usuario practicar, aprender, evaluar y adquirir conocimientos de actuaciones humanas. Los simuladores son herramientas confiables que permiten al usuario experimentar una situación real sin los riesgos que se presentan en escenarios reales, siendo versiones simplificadas de situaciones reales, los hacen fáciles de manejar, evitan el riesgo de manejo de equipos costosos y permite retroalimentarse de manera constante, ya que el usuario puede practicar las veces que le sean necesarias. Al interactuar con los simuladores, el usuario reaccionará como si fueran situaciones reales, lo cual le permitirá fortalecer habilidades y competencias en distintas áreas del conocimiento, sin que se vean expuestos a riesgos tales como su integridad física o emocional. La interacción con escenarios de situaciones recreadas de contextos de la vida real permitirá al usuario fortalecer sus habilidades, reforzar sus niveles de tolerancia a la frustración, elevará su motivación, apoyará al logro de objetivos y a resolver problemas que se le presenten.

El proceso de diseño de ambientación de escenarios virtuales se logra por medio de bocetos que se usan como elementos de apoyo (sillas, escritorios, etc.), por lo que, para el

diseño de estos escenarios se tomó como base los recursos del *Asset Store* de Unity, el cual nos brinda elementos como la estructura de la oficina, escritorios, sillas, etc.

7.1.2.2. Escenario Simulador Laboral "El Tiempo"

Para el modelado de este escenario se utilizó el recurso *Simple Office Interiors* del *Asset Store* de Unity, el cual nos brinda elementos como la estructura de la oficina, escritorios, sillas, etc.

Desde Unity accedemos *al Asset Store* e importamos el paquete de Assets (recursos) para el diseño del escenario, e iniciamos con nuestro diseño agregando los elementos necesarios, a continuación, se presenta el diseño de escenario realizado para el simulador laboral El Tiempo.

A continuación, se plantea un escenario ambientado en una oficina de un jefe de talento humano, en el que el personaje se dirige a su lugar de trabajo e inicia sus actividades. Por lo que, el objetivo de este escenario es permitir al participante experimentar con un escenario lo más fiel posible a una situación real, en el que pueda poner en práctica sus conocimientos teóricos sin miedo a equivocarse, aprendiendo de sus propios errores a tomar mejores decisiones y reducir niveles de frustración. De esta manera, tendrá la oportunidad de prepararse en cómo actuar o responder en una situación real.

7.1.2.3. Escenario Simulador Laboral "Información Difícil"

Para el modelado de este escenario se utilizó el recurso *School Scene* del *Asset Store* de Unity, el cual nos brinda elementos como la estructura de la oficina, escritorios, sillas, etc.

Desde Unity accedemos *al Asset Store* e importamos el paquete de Assets (recursos) para el diseño del escenario e iniciamos con nuestro diseño agregando los elementos necesarios. A continuación, se presenta el diseño de escenario realizado para el simulador laboral Información Difícil.

A continuación, se plantea un escenario ambientado en una oficina de un rector de una institución educativa privada, en el que el personaje se dirige a su lugar de trabajo y al llegar se le informa que deberá realizar ciertas actividades, de las cuales se ve obligado a realizar un recorte de personal y prescindir de algunos colaboradores. Por las decisiones que deberá tomar, el personaje muestra una expresión preocupante en su rostro, por lo tanto el objetivo de este escenario es permitir al participante experimentar con un escenario lo más fiel posible a una situación real como esta, en la que pueda poner en práctica sus conocimientos

teóricos sin miedo a equivocarse, aprendiendo de sus propios errores, así tomar mejores decisiones y reducir niveles de frustración, de esta manera tendrá la oportunidad de prepararse en cómo actuar o responder en una situación real.

Figura 3. Vista escritorio de la oficina.

7.1.2.4. Modelado de Personaje

A la hora de seleccionar o modelar el personaje es importante lograr que el jugador o estudiante se vea reflejado en el personaje y sienta la experiencia del mismo. Junto al grupo de investigación de la Cátedra UNESCO se define que se utilice el mismo personaje para los dos simuladores laborales, para lo cual descargamos el personaje *Eric Man* de la página ***Cgtrader*** (mercado de modelos 3D) de manera gratuita.

Al finalizar la descarga tendremos un archivo .zip, el cual descomprimimos e importamos la carpeta a nuestro proyecto en Unity en la carpeta Assets y finalmente añadimos el personaje a la escena.

7.1.2.5. Animaciones

Una vez diseñado los escenarios para los simuladores laborales y agregado el personaje en la escena, se procede a generar las respectivas animaciones que darán sentido a la experiencia con el simulador. Desde Unity creamos un componente *Animator* desde la ventana proyect damos clic derecho *Create/AnimatorController*, en este caso se utiliza las animaciones predeterminadas del portal web Míxamo, aquí cargamos el personaje que cargamos a la escena y lo subimos con el personaje con la extensión .fbx, gracias a la amplia gala de animaciones que tiene Míxamo, ahora procedemos a descargar las animaciones que sean necesarias.

Para la animación de nuestro personaje en la escena de cada simulador laboral, iniciamos posicionando al personaje desde el punto del cual va a iniciar. En el caso del simulador laboral, el tiempo trata de un jefe de talento humano, el cual se dirige a su oficina y lo mismo para el simulador laboral información difícil trata de un personaje, el cual es un director de una institución educativa privada que de igual manera se dirige a su oficina. Por ello, se integra un componente A*nimator* a nuestro personaje y dependiendo de la ruta que debe recorrer el personaje en cada simulador se agregan las animaciones necesarias, tales como caminar de frente, girar a la izquierda, girar a la derecha, sentarse, etc.

Figura 4. Elaboración de animación personaje simulador El Tiempo.

Figura 5. Elaboración de animación de personaje simulador Información Difícil.

7.1.2.6. Propuesta de interfaz de usuario para simuladores laborales

Una vez terminado el diseño de personaje y escenarios se procede a crear la interfaz de usuario para cada simulador laboral, para ello se diseña un prototipo que defina la estructura para cada interfaz, la cual se compone de cuatro botones base (*botón salir, preferencias, atrás y siguiente*).

Figura 6. Propuesta de interfaz de usuario integrando accesibilidad.

7.1.2.7. Interfaz de usuario accesible para simuladores laborales 3D

Iniciamos con el diseño de una interfaz de usuario accesible en el que el diseño va dirigido hacia la interacción *estudiante-simulador laboral*. Para el desarrollo de los simuladores laborales, se han tomado en cuenta los siguientes aspectos de accesibilidad: *ambientación de escenarios, tipos de fuente, contraste, tamaño de fuente, audio descripción, tamaño de botones e interacción teclado.*

La integración de un módulo de accesibilidad se enfoca en lograr que el estudiante pueda interactuar de una manera accesible con los simuladores laborales, permitiéndole al estudiante adecuar el simulador conforme a sus necesidades, ya sean visuales, auditivas o cognitivas. Para el desarrollo de este módulo de accesibilidad bajo el estándar WCAG 2.0, el cual contiene 12 pautas que se agrupan en cuatro principios: perceptible, operable, comprensible y robusto, los que tienen como objetivo alcanzar productos lo más accesibles para usuarios con discapacidad.

Principio 1: Perceptible: Este principio se fundamenta en que todo el contenido e información de la interfaz de usuario deben ser presentados de manera que sea fácil percibir para el usuario.

Para que el contenido de la interfaz de los simuladores laborales sea perceptible se implementa la descripción de audios, los cuales corresponden a todo el texto presente y demás componentes con texto alternativo (botones salir, atrás, siguiente, preferencias, etc.) a los cuales se les ha asignado imágenes decorativas.

Botones	**Imagen decorativa**	**Función**
Empezar		Al presionar este botón el participante podrá iniciar a la simulación del ejercitario (ejercicio) planteado.
Salir		Al presionar este botón, el participante saldrá del simulador laboral.
Atrás		Al presionar este botón el participante será dirigido al panel anterior.
Siguiente		Al presionar este botón el participante pasará al siguiente panel.
Fuentes		Se implementó tres botones en el que cada uno corresponde a un tipo de fuente diferente, de los cuales el participante podrá ir seleccionando.
Contrastes		Se implementó dos botones en los que cada uno corresponde a cada tipo de contraste, de los cuales el participante podrá ir seleccionando.
Tamaño Fuentes		Se implementó dos botones en los que cada uno corresponde a un distinto tamaño de fuente, con los cuales el participante podra ir interactuando.
Audio		El presionar este botón el participante podra desactivar o desactivar el audio de la escena, para ello se hace uso de dos imágenes, una que indica cuando el audio esta activado y otra que indica cuando el audio esta descativado.
Preferencias		Se implementa un botón con el cual el participante podrá acceder a la barra de preferencias.

Salir de Preferencias		Se implementa un botón con el cual el participante podrá salir de la barra de preferencias.
Reiniciar		Se implementa un botón con el cual el participante podrá retornar al panel inicial una vez que haya llegado al panel final del simulador laboral.

Tabla 6. Distribución de botones para mecánicas de juego.

Nota" El tamaño de los botones dependerá de la cantidad de contenido que se quiera mostrar"

Hacer que el contenido sea distinguible ayuda a que sea más fácil de ver y escuchar para el usuario, por ello es necesario combinar correctamente el color del contenido frontal con respecto al contenido del fondo para lograr un contraste adecuado de las interfaces de los simuladores, en este caso se maneja dos combinaciones de contraste (*fondo color negro + frente color amarillo*). Así como también, el modificar el tamaño de los textos sin que se pierda información, hacer uso de fuentes de texto accesibles, los cuales ayuden a una percepción que el audio pueda ser activado o desactivado.

Con respecto al contraste, se implementa dos opciones tomando como referencia las opciones de accesibilidad del sitio web de simuladores del grupo Edutech (*https:simulab.edutech-project.org/):* contraste1 (*combinación de contenido color azul con fondo blanco*) y contraste2 (*combinación de contenido color amarillo con fondo negro*) para ello verificamos que los colores elegidos sean accesibles mediante el sitio web *webaim.org.*

Figura 7. Interfaz de usuario usando el contraste 1.

Figura 8. Interfaz de usuario usando el contraste 2.

Principio 2: Operable: Este principio se basa en que todas las funciones deben estar disponibles a través del teclado, las funciones deberán ser las mismas que las del mouse, lo cual ayudaría no solo a usuarios con discapacidad motriz sino a todos los usuarios que lo deseen. Para la navegación por teclado se ha hecho uso de determinadas teclas (*teclas: flecha arriba, flecha izquierda y alt*), las cuales permitirán al usuario navegar por todo el contenido hacia

atrás, mientras que las teclas (*teclas: flecha abajo, flecha derecha y tab*) le permitirán al usuario navegar por todo el contenido hacia adelante; cada vez que se posicione en algún botón se activará un indicador de enfoque en el que para una mejor percepción se hace uso de imágenes adecuándoles colores distintivos, las imágenes se activan cuando el indicador se activa, de esta manera el usuario sabrá en que parte de la interfaz está ubicado. Esta estructura permitirá que el estudiante tenga la opción de elegir con cual de sus manos desea controlar el teclado.

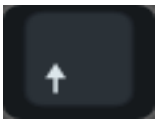
Distribución de teclas de navegación por teclado	
Navegación Hacia Adelante	**Navegación Hacia Atrás**
Tab 	Alt 
Flecha Derecha 	Flecha Izquierda 
Flecha Abajo 	Flecha Arriba 

Tabla 7. Distribución de teclas de navegación en interfaz de usuario de simuladores laborales 3D.

Principio 3: Comprensible: Este principio se basa en que la información y manejo de la interfaz de usuario debe ser de fácil comprensión. En el caso de información comprensible, se hace uso de tres fuentes gratuitas recomendadas: *Arial Bold, Lato y Open Dyslexic* para mejorar la lectura y accesibilidad en usuarios con discapacidad visual, la información debe ser lo más clara posible para facilitar a usuarios con discapacidad cognitiva, para alcanzar un contenido accesible se utiliza la misma estructura y mecanismo de navegabilidad en todas las interfaces de cada simulador laboral, de tal manera evitamos que el usuario se frustre ante algún comportamiento inconsistente.

Principio 4: Robusto: Este principio se basa en que la aplicación debe ser compatible con cualquier navegador, es decir, que no se pierdan las funcionalidades de la aplicación. Es este caso se cumple el principio, ya que al desplegar los simuladores laborales en el servidor

web se realizaron las respectivas pruebas de funcionamiento mediante la navegación por mouse y teclado.

A continuación, se detalla el proceso de diseño de la interfaz de usuario accesible realizado para cada simulador laboral 3D, en el que cada interfaz está compuesta de varios paneles (*información introductoria, actividades, calificación y retroalimentación*) manteniendo cada uno la misma estructura, a más de ello se diseña un panel o barra de preferencias, el cual cuenta con opciones de accesibilidad (*seleccionar tipo de fuente, contraste, tamaño de texto, activar o desactivar audio*).

Cuando la simulación inicie se aparecerá en pantalla un personaje con su respectiva animación, al finalizar la animación del personaje se presenta al participante una interfaz con un botón que al seleccionarlo le permitirá iniciar la interacción con el simulador laboral, para ello podrá acceder mediante el teclado (flecha arriba, flecha abajo, fecha izquierda o flecha derecha) + la tecla *intro, tecla espacio* o mediante el uso del mouse.

Figura 9. Diseño de panel inicial para simuladores laborales, el cual contiene el botón para empezar la interacción con el ejercitario (ejercicio).

Figura 10. Diseño arquitectura de paneles (pantallas) de simuladores laborales, para mostrar el instructivo de cada ejercitario (ejercicio).

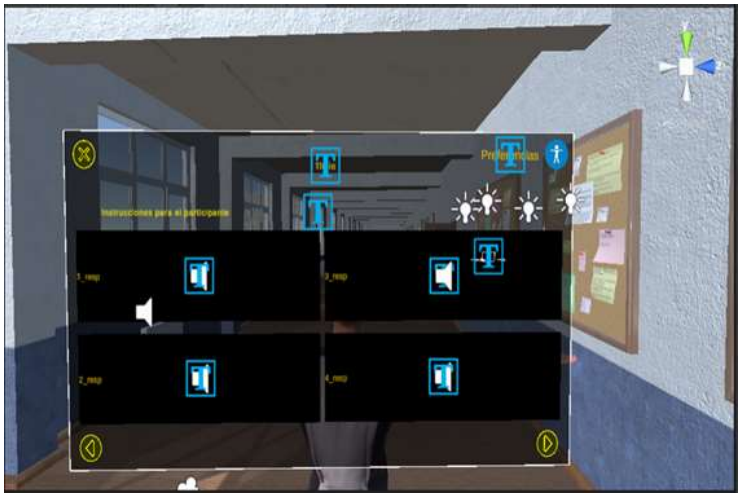


Figura 11. Diseño de arquitectura para actividades de tipo selección, en este caso para el Simulador Laboral Información Difícil.

Figura 12. Diseño de arquitectura para actividades de tipo numérico en donde el participante deba ingresar valores, en este caso para el Simulador Laboral El Tiempo.

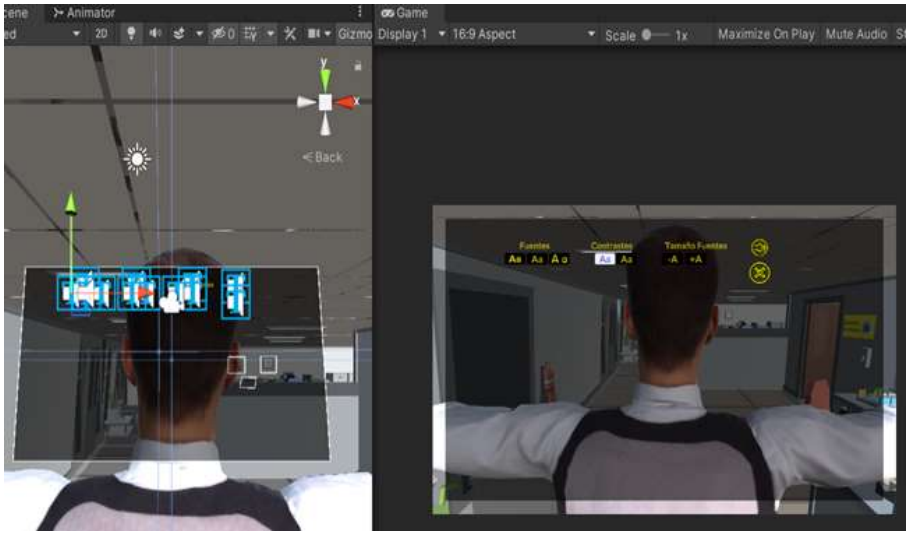


Figura 13. Diseño de barra de preferencias para simuladores laborales.

7.1.3. Fase 3 Desarrollo o prototipado

En la fase de diseño se crearon los escenarios para los simuladores laborales, los cuales se ambientaron cada uno según la situación real planteada, también se diseñó una interfaz de usuario accesible para cada simulador laboral, la interfaz contiene todo lo referente al simulador laboral con el que deberá interactuar el participante.

Por lo que, en esta fase se procede a darle sentido al diseño realizado, en el que se desarrollan las mecánicas de juego, lógica de los ejercitarios planteados, etc.

7.1.3.1. Base de datos simuladores laborales 3D

Para el desarrollo de este proyecto se utilizó parte de la base de datos del servidor web de Simuladores Laborales 3D del proyecto Edutech. A continuación, se indica las entidades utilizadas:

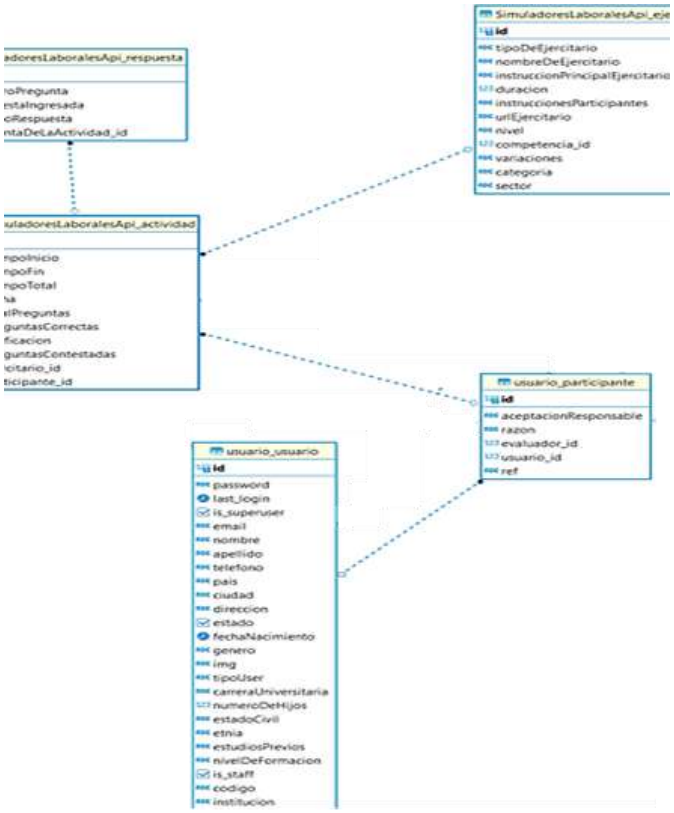


Figura 14. Entidades utilizadas de la base de datos del servidor web del proyecto Edutech.

7.1.3.1.1. Entidades utilizadas

SimuladoresLaboralesApi_respuesta: Mediante esta entidad se puede guardar(*numeroPregunta*, *respuestaIngresada*, *tiempoRespuesta*).

SimuladoresLaboralesApi_actividad: Mediante esta entidad se guarda el (*tiempoInicio, tiempoFin, fecha),* además se relaciona con la entidad usuario_participante y este a su vez se relaciona con la entidad *usuario_usuario* de esta manera se accede al *correo* del participante, también se relaciona con la entidad SimuladoresLaboralesApi_ejercitario en lo cual se accede a al *id* del ejercitario.

A continuación, se detalla la programación de las mecánicas o reglas de los simuladores laborales, lógica de calificación, implementación de barra de preferencias, etc.

7.1.3.2. Estructura de simuladores laborales

Para cada simulador se implementó las mismas mecánicas de juego, lo único que en algún momento puede variar es el tipo de actividades para el usuario, es decir, en el simulador laboral Información difícil se presentan actividades o preguntas de tipo selección, pero en el

caso del simulador laboral El Tiempo se presentan actividades en las cuales el participante debe ingresar valores por teclado, pero en caso de integrar estos scripts en nuevos proyectos, solo se tendría que acoplar el mismo código según sea el caso. La estructura que se hace es por el tipo de actividades, en este caso se tiene actividades de tipo selección, es decir, al participante se le presenta un enunciado con varias alternativas de las cuales deber escoge una y por otro lado se tiene actividades de tipo numéricos, en las que el participante debe ingresar números. Es por ello que, el simulador laboral Información Difícil se compone de 1 script principal y 6 secundarios. En el caso del simulador laboral El Tiempo, se compone de 1 script principal y 5 secundarios, ésta implementación resulta óptima a la hora de desarrollar más simuladores laborales de este tipo.

Además de los scripts, es importante que en cada proyecto que se realice se agregue un plugin en el cual están definidas las funciones JavaScript que nos permitirá la comunicación con el servidor web, el plugin lo debemos colocar en la carpeta del proyecto *Assets/Plugins/Plugin.jslib* y uno de los scripts es el que tiene el código necesario para comunicarnos con el plugin y finalmente consumir el servicio web.

7.1.3.2.1. Scripts Simuladores Laborales

Se implementaron 7 scripts que fueron integrados en los dos simuladores laborales, lo cual permitirá que estos scripts sean integrados en futuros simuladores laborales.

Simulador Laboral Información Difícil conformado por 7 scripts (GeneralEmpezar.cs, MouseOver.cs, OverTextos.cs, OpcionesOver.cs, MouseColor.cs, PreferSctipt.cs y WebData.cs).

Simulador Laboral El Tiempo conformado por 6 scripts (GeneralEmpezar.cs, MouseOver.cs, OverTextos.cs, MouseColor.cs, PreferSctipt.cs y WebData.cs)

1 El Script GeneralEmpezar.cs controla las mecánicas de navegación, gestión de calificación, envío de datos, etc.

2 El Script MouseOver.cs permite reproducir el audio que tiene un botón(salir, siguiente,etc) cuando el mouse está sobre él.

3 El Script OverTextos.cs permite reproducir el audio de textos de un objeto (*títulos, subtítulos, descripciones*) cuando el mouse está posicionado sobre él.

4 El Script OpcionesOver.cs permite reproducir el audio de un botón (*preguntas*) cuando el mouse está posicionado sobre él .

5 El Script MouseColor.cs resalta de un color el objeto cuando el mouse está sobre él.

6 El Script PreferSctipt.cs contiene las opciones de accesibilidad de personalización de los simuladores laborales.

7 El Script WebData contiene el código de comunicación con el plugin (funciones JavaScript) que permitirá la comunicación con el servidor web.

7.1.3.2.1.1. GeneralEmpezar.cs

Se da inicio con la instancia de variables de los objetos que representan a los botones, campos de texto, objetos escenario, el texto correspondiente a la información a mostrar al participante del ejercitario planteado, listas de imágenes para cada panel presentado, variables que harán de contadores, etc., podemos encontrar las imágenes del código completo empleado junto a la documentación del script en los Anexos F.

```
public GameObject juego, botSalir, botPref, botSig, botAtras, tit, detall, conte
public TextMeshProUGUI textoEmpiezo,textoTit,textPreg, textodetall, textoconten,
public Image empiezo, nextB;
public GameObject prefer, fondoPC, papeles, papeles1, calcu, mouse;
public static bool pefBool = false;
string documento = "Por favor seleccione la opción que considere correcta para s
string line;
string[] tiemposResp;
int pos = 0, contRes = 0;
List<string> contenPreg, alternaPregs, contenPregFinal, alternaPregsFinal;
public GameObject[] altern;
public TextMeshProUGUI[] textalter;
public string[] rescor;
```

Figura 15. Declaración de variables Simulador Laboral Información Difícil.

Figura 16. Declaración de variables Simulador Laboral El Tiempo.

En este script se implementó la mayoría de las funcionalidades, entre ellas las mecánicas de juego, creando las funciones:

La función *Tabulando()* controla los eventos del teclado, es decir, la navegación en la interfaz al presionar ciertas teclas, para lograrlo primero se declaran arreglos de tipo Selectable, luego asociamos con cada GameObject de la escena, luego agregamos a nuestra sentencia de código el método *Select()* de Unity, el cual hace que se seleccione el componente requerido cada vez que se navegue con el teclado. Para controlar en que parte del simulador se encuentra el participante, se hace uso de la variable estática *menú*, esta variable va cambiando de nombre en cada panel que nos encontremos. Por ejemplo, para indicar que estamos en el panel principal decimos que menú es igual a *Main*, cuando estamos en el panel de instrucciones menú es igual a *Jugando*, cuando pasemos al panel de preguntas menú es igual a *Preguntas*, cuando pasemos al panel de calificación menú es igual a *FinA* y cuando pasemos al último panel menú es igual a *Fin*. podemos encontrar las imágenes del código completo empleado junto a la documentación del script en los Anexos F.

Figura 17. Fragmento de código para controlar que estamos en el panel principal, simulador laboral Información Difícil.

Figura 18. Fragmento de código para controlar que estamos en el panel principal, simulador laboral El Tiempo.

La función *Void Update()* controla los movimientos del mouse en el escenario, reiniciará el cronómetro cada vez que pase al panel de preguntas, desactivará al personaje cuando aparezca en pantalla el botón empezar y hará que siempre se ejecute la función de eventos del teclado. Podemos encontrar las imágenes del código completo empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

```
void Update()
{
    if (GeneralEmpezar.menu == "Preguntas")
    {
        correTiempo();
    }
    if (botEmpe.activeSelf == true && move == true)
    {
        float pointer_x = Input.GetAxis("Mouse X");
        float pointer_y = Input.GetAxis("Mouse Y");
        camara.transform.Rotate(0, pointer_x * 1f, 0);
    }
    if (botEmpe.activeSelf == true)
    {
        personaje.SetActive(false);
        personajeHuesos.SetActive(false);
    }
    Tabulando();
}
```

Figura 19. Fragmento de código función Update Simulador Laboral Información Difícil

Algo adicional que se agrega en la función Update() del Simulador Laboral El Tiempo, es que debido a que el participante debe ingresar valores se controla que mientras el participante no haya llenado todos los espacios de los campos de entrada, es decir, que aún no haya completado todas las respuestas o que haya valores nulos no se le activará el botón siguiente.

```
void Update()
{
    //si menu es igual a Preguntas
    if (GeneralEmpezar.menu == "Preguntas")
    {
        correTiempo();//empieza a tomar el tiempo
    }
    /*si el botón empezar esta activo y el movimiento de la cámara esta activo,
    entonces me permitirá mover la cámara con el mouse*/
    if (botEmpe.activeSelf == true && movoCamara == true)
    {
        float pointer_x = Input.GetAxis("Mouse X");
        float pointer_y = Input.GetAxis("Mouse Y");
        camara.transform.Rotate(0, pointer_x * 1f, 0);
    }
    if (botEmpe.activeSelf == true)//si el botón empezar esta activo
    {
        /*se desactiva el personaje*/
        personaje.SetActive(false);
    }
    //llamamos a la función tabulando, para que siempre este escuchando a los eventos del teclado
    Tabulando();

    //si el participante completa todos los campos de entrada de texto
    if (GeneralEmpezar.menu == "Preguntas" && resOp1.text != "" && resOp2.text != "" && resOp3.text != "" && resOp4.text != ""
    && resOp5.text != "" && resOp6.text != "" && diasOp1.text != "" && diasOp2.text != "" && diasOp3.text != "" &&
     diasOp4.text != "" && diasOp5.text != "" && diasOp6.text != "")
    {
        //Se activa el botón siguiente
        botSig.SetActive(true);
```

Figura 20. Fragmento de código función Update () Simulador El Tiempo.

La función *Empiezo ()* permite iniciar la interacción del participante con el simulador laboral, esta función es llamada al presionar el botón empezar del panel principal, podemos encontrar las imágenes del código empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

```
public void Empiezo()
{
    nuevoJuego();
    textoTit.text = "EJERCITARIO INFORMACIÓN DIFÍCIL";
    textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE";
    datetime = DateTime.Now.ToString("hh:mm:ss");
    sigoFin.image.sprite = ImagenSiguiente;
    GenerarAleatoriosSinRepetir();
    GeneralEmpezar.move = false;
    empiezo.enabled = false;
    textoEmpiezo.gameObject.SetActive(false);
    juego.SetActive(true);
    botSig.SetActive(true);
    botPref.SetActive(true);
    botSalir.SetActive(true);
    tit.SetActive(true);
    detall.SetActive(true);
    conten.SetActive(true);
    GeneralEmpezar.menu = "Jugando";
    contadorPR = 0;
}
```

Figura 21. Fragmento de código de la función Empiezo () del Simulador Información Difícil.

```
public void Empiezo()
{
    nuevoJuego();
    textoTit.text = "Ejercitario El Tiempo";
    textodetall.text = "Instrucciones para el participante";
    textoconten.text = instrucciones[conteoInstruc];
    datetime = DateTime.Now.ToString("hh:mm:ss");
    sigoFin.image.sprite = ImagenSiguiente;
    GenerarAleatoriosSinRepetir();
    GeneralEmpezar.muevoCamara = false;
    empiezo.enabled = false;
    textoEmpiezo.gameObject.SetActive(false);
    juego.SetActive(true);
    botSig.SetActive(true);
    botAtras.SetActive(true);
    botPref.SetActive(true);
    botSalir.SetActive(true);
    tit.SetActive(true);
    detall.SetActive(true);
    conten.SetActive(true);
    GeneralEmpezar.OpcMen = "Jugando";
}
```

Figura 22. Fragmento de código de la función Empiezo () Simulador Laboral El Tiempo.

La función *ActivoPrefer()* es llamada al presionar el botón de preferencias, lo que hace es habilitar la barra de preferencias. Para evitar errores de visualización se desactiva los objetos que puedan sobreponerse, podemos encontrar las imágenes del código empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

```
public void ActivoPrefer()
{
    if (pefBool == false)
    {
        prefer.SetActive(true);
        GeneralEmpezar.menu = "Preferencias";
        tit.SetActive(false);

        GeneralEmpezar.pefBool = true;
        if (pregu.activeSelf == true)
        {
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
        }
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            foreach (var item in dial)
            {
                item.enabled = false;
            }
        }
        else if (
            textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
        )
        {
            foreach (var item in finAI)
            {
                item.enabled = false;
            }
        }
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            foreach (var item in finI)
            {
                item.enabled = false;
            }
        }
    }
}
```

Figura 23. Fragmento de código de función que al llamarla activa la barra de preferencias del Simulador Laboral Información Difícil.

```
public void ActivoPrefer()
{
    if (prefMuestro == false)
    {
        prefer.SetActive(true);
        GeneralEmpezar.OpcMen = "Preferencias";
        tit.SetActive(false);
        GeneralEmpezar.prefMuestro = true;
        if (pregu.activeSelf == true)
        {
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
        }
        else if (textodetall.text == "Instrucciones para el participante")
        {
            foreach (var item in dial)
            {
                item.enabled = false;
            }
        }
        else if (textodetall.text == "Gracias por utilizar el simulador laboral")
        {
            foreach (var item in finAI)
            {
                item.enabled = false;
            }
        }
        else if (textodetall.text == "Retroalimentación Final")
        {
            foreach (var item in finI)
            {
                item.enabled = false;
            }
        }
    }
}
```

Figura 24. Fragmento de código de función que al llamarla activa la barra de preferencias Simulador Laboral El Tiempo.

La función *DesctivoPrefer()* permite ocultar la barra de preferencias cuando el participante haya seleccionado el botón salir de preferencias, solo entonces se activarán nuevamente los objetos que se había desactivado, podemos encontrar las imágenes del código empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

```
public void DesctivoPrefer()
{
    if (pefBool == true)
    {
        prefer.SetActive(false);
        tit.SetActive(true);
        GeneralEmpezar.pefBool = false;
        if (pregu.activeSelf == true)
        {
            GeneralEmpezar.menu = "Preguntas";
        }
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            GeneralEmpezar.menu = "Jugando";
        }
        else if (
            textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
        )
        {
            GeneralEmpezar.menu = "FinA";
        }
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            GeneralEmpezar.menu = "Fin";
        }
    }
}
```

Figura 25. Fragmento de código de función que al ser llamada permite desactivar la barra de preferencias del Simulador Laboral Información Difícil.

```
public void DesctivoPrefer()
{
    if (prefMuestro == true)
    {
        prefer.SetActive(false);
        tit.SetActive(true);
        GeneralEmpezar.prefMuestro = false;
        if (pregu.activeSelf == true)
        {
            GeneralEmpezar.OpcMen = "Preguntas";
        }
        else if (textodetall.text == "Instrucciones para el participante")
        {
            GeneralEmpezar.OpcMen = "Jugando";
        }
        else if (textodetall.text == "Gracias por utilizar el simulador laboral")
        {
            GeneralEmpezar.OpcMen = "FinA";
        }
        else if (textodetall.text == "Retroalimentación Final")
        {
            GeneralEmpezar.OpcMen = "Fin";
        }
    }
}
```

Figura 26. Fragmento de código de función que al ser llamada permite desactivar la barra de preferencias del Simulador Laboral El Tiempo.

Cada vez que el participante presiona el botón empezar del panel principal se presentan las preguntas de manera aleatoria, para ello se crea una función *GenerarAleatoriosSinRepetir*(), la cual permite generar números aleatorios. En el caso del simulador laboral Información Difícil se generan números randomicos de 0 a 1, ya que solo tiene dos preguntas. Para el simulador laboral El Tiempo se generan números randomicos de 0 a 5, ya que en este caso son 6 preguntas.

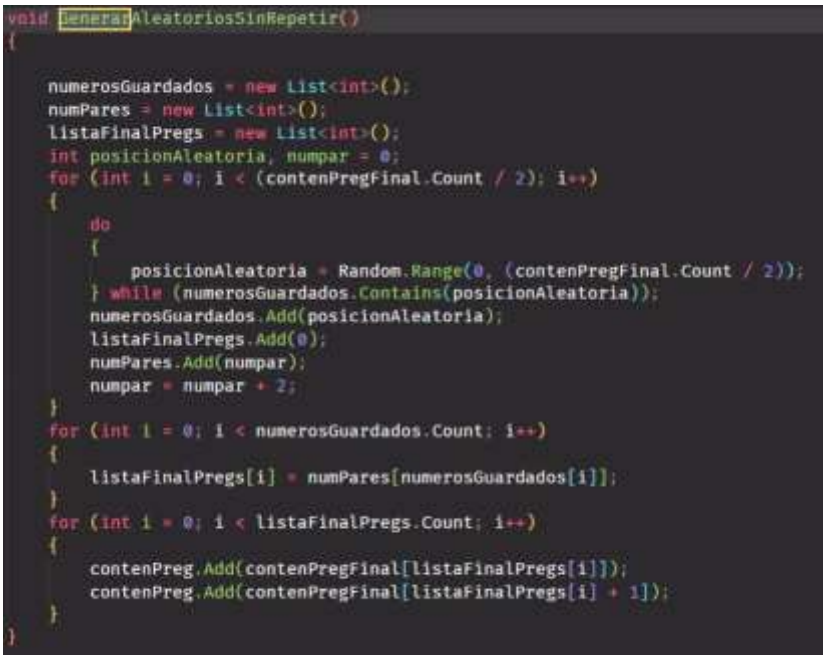


Figura 27. Fragmento de código de la función GenerarAleatoriosSinRepetir() Simulador Laboral Información Difícil.

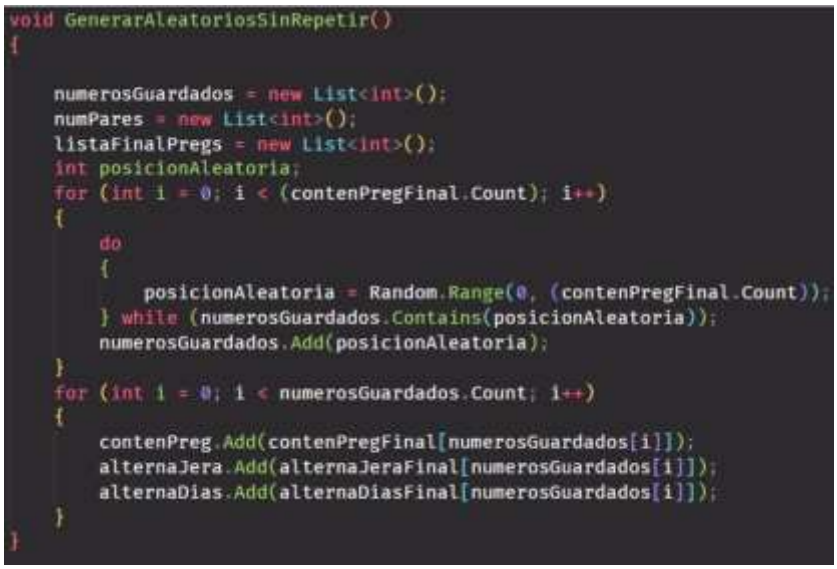


Figura 28. Fragmento de código de la función GenerarAleatoriosSinRepetir() Simulador Laboral El Tiempo.

En el caso del simulador laboral El Tiempo se crea la función *soloNúmeros*(), para validar el ingreso de campos de texto, en este caso se valida que solo se ingrese números, también se valida que el participante no ingrese valores repetidos a la hora de jerarquizar las actividades planteadas, para ello se crea la función *noRepetidosIngreso()*.

```
public void soloNumeros(TMP_InputField textBox1)
{
    try
    {
        int.Parse(textBox1.text);
    }
    catch (Exception e)
    {
        textBox1.text = "";
    }
}

public void noRepetidosIngreso(TMP_InputField textBox1)
{
    foreach (var item in audiosPregMouse)
    {
        item.Stop();
    }
    foreach (var item in jerarquiaInput)
    {
        if (item.name != textBox1.name)
        {
            if (item.text == textBox1.text && textBox1.text != "")
            {
                textBox1.text = "";
                sonidoAS.clip = avisoRepe;
                sonidoAS.Play();
            }
        }
    }
}
```

Figura 29. Validaciones de ingreso de datos del Simulador Laboral El Tiempo.

Al llamar a la función *Siguiente()* primeramente se llama a la función *ReinicioSeleccion(),* en el que se iniciará en cero todos los valores de las variables, por ejemplo, las variables de las respuestas seleccionadas. Creamos un condicional para verificar que la barra de preferencias esté desactivada, caso contrario el participante no podrá pasar al siguiente panel, se hace esto con el fin de facilitar la navegación a estudiantes con discapacidad.

En esta función también presentan las actividades a resolver para el participante, para ello primero consultamos si *menú* es igual a *Preguntas*, si es que si entonces se muestra al participante la actividad junto a las posibles respuestas, cuando seleccione el botón la opción que crea correcta internamente se estará llamando la función Respondo() que tomará las

respuestas seleccionadas por el participante y las comparará contra el arreglo de respuestas correctas, una vez que el participante conteste la última pregunta pasará al panel de calificación, eso lo controlamos por medio de un condicional en el que decimos si el contador de preguntas es igual al número de preguntas que tiene la lista de preguntas. Significará que ha terminado de resolver las preguntas, lo cual lo llevará al panel de calificación, una vez ahí se mostrará el puntaje obtenido. Para ello, se consulta si la variable *menú* es igual a *FinA*, si es que si, entonces dice, si el puntaje obtenido es 0 la calificación es 0%, si el puntaje obtenido es 50 entonces la calificación es 50% y si el puntaje obtenido es 100 entonces tendrá el 100%. Finalmente, cuando la variable *menú* sea igual a Fin entonces estaremos en el último panel, una vez que estemos internamente se llama a la función *Post()* para consumir el servicio web. Podemos encontrar las imágenes del código empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

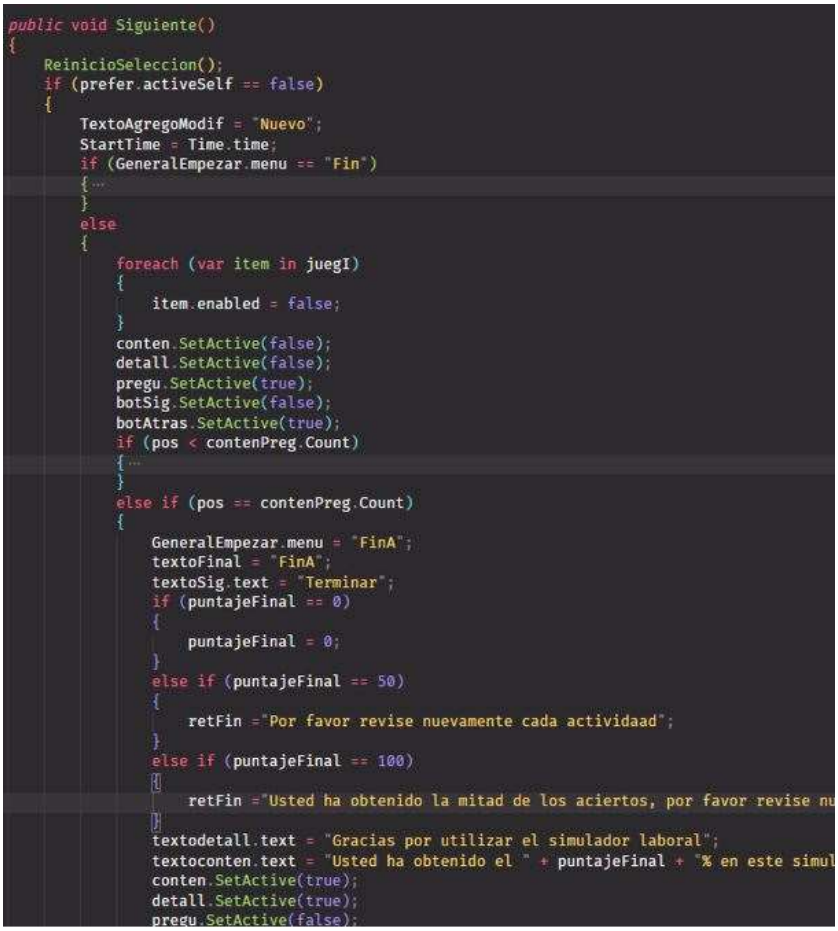
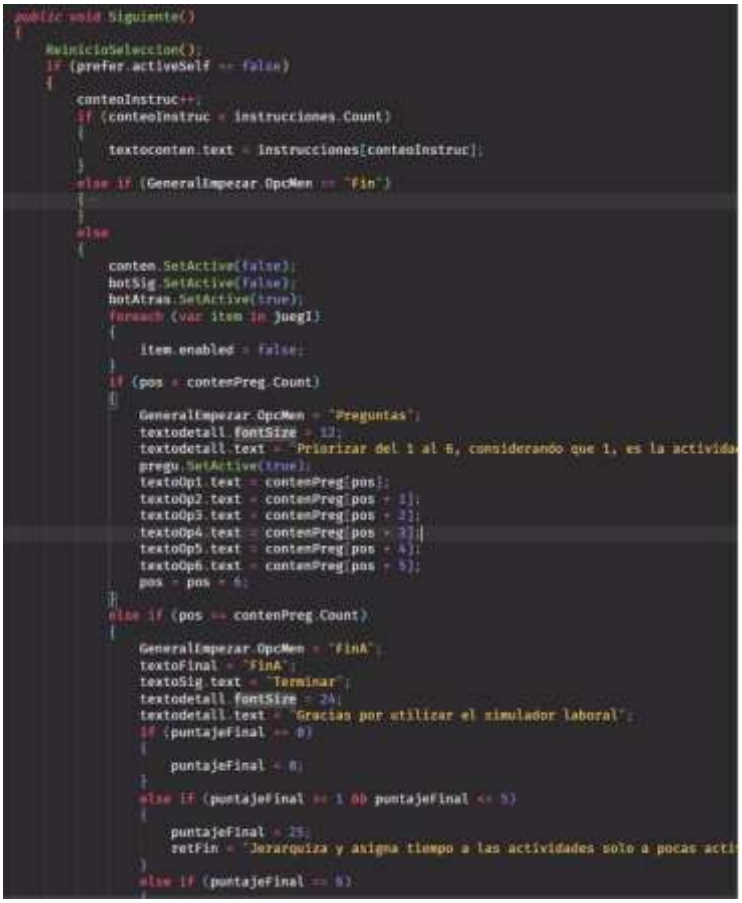


Figura 30. Fragmento de código de la función Siguiente () del Simulador Laboral Información Difícil.



```
public void Siguiente()
{
    ReinicioSeleccion();
    if (prefer.activeSelf == false)
    {
        conteoInstruc++;
        if (conteoInstruc < instrucciones.Count)
        {
            textoconten.text = instrucciones[conteoInstruc];
        }
        else if (GeneralEmpezar.OpcMen == "Fin")
        {
        }
        else
        {
            conten.SetActive(false);
            botSig.SetActive(false);
            botAtras.SetActive(true);
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
            if (pos < contenPreg.Count)
            {
                GeneralEmpezar.OpcMen = "Preguntas";
                textodetall.fontSize = 12;
                textodetall.text = "Priorizar del 1 al 6, considerando que 1, es la activida
                pregu.SetActive(true);
                textoOp1.text = contenPreg[pos];
                textoOp2.text = contenPreg[pos + 1];
                textoOp3.text = contenPreg[pos + 2];
                textoOp4.text = contenPreg[pos + 3];
                textoOp5.text = contenPreg[pos + 4];
                textoOp6.text = contenPreg[pos + 5];
                pos = pos + 6;
            }
            else if (pos >= contenPreg.Count)
            {
                GeneralEmpezar.OpcMen = "FinA";
                textoFinal = "FinA";
                textoSig.text = "Terminar";
                textodetall.fontSize = 24;
                textodetall.text = "Gracias por utilizar el simulador laboral";
                if (puntajeFinal == 0)
                {
                    puntajeFinal = 0;
                }
                else if (puntajeFinal >= 1 && puntajeFinal <= 5)
                {
                    puntajeFinal = 25;
                    retFin = "Jerarquiza y asigna tiempo a las actividades solo a pocas acti
                }
                else if (puntajeFinal == 6)
                {
```

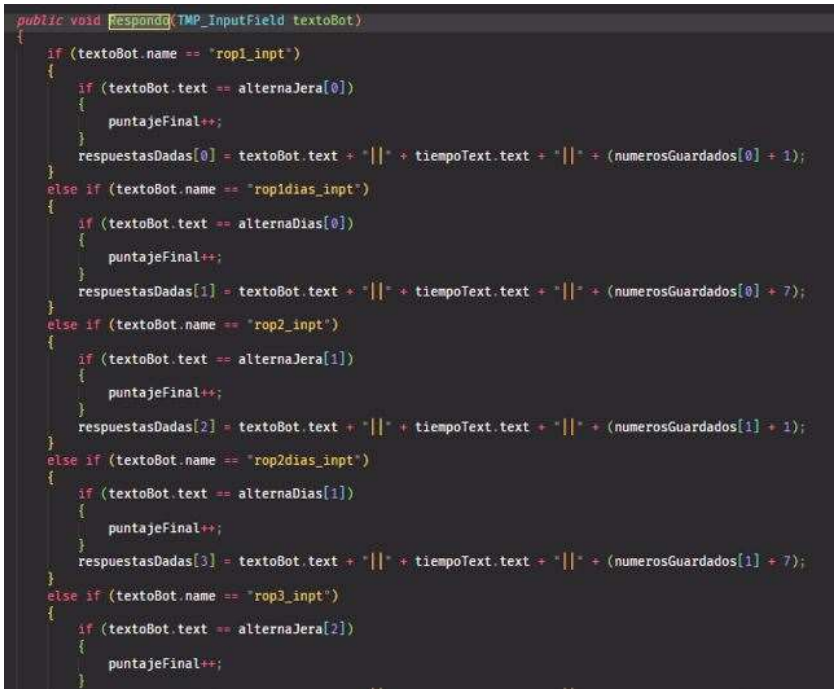
Figura 31. Fragmento de código de la función Siguiente () del Simulador Laboral El Tiempo.

Para la calificación, se toma como base la rúbrica de calificación de los ejercitarios propuestos. Como se mencionó en el paso anterior, la función *Respondo()* realiza el cálculo de calificación, entonces usamos un condicional en el que decimos que, si el objeto seleccionado por el participante es igual al nombre registrado como respuesta correcta, si es que si, se sumará 50 a esa pregunta, el valor lo guardamos en una variable de tipo entero *puntaje Final*, además captura el tiempo de respuesta empleado por pregunta. Si el participante ya respondió la primera pregunta y por algún motivo retrocede de panel y al hacerlo la o las respuestas seleccionadas se eliminan y empiezan sus valores en 0, en este caso para el control de tiempo de respuesta se está guardando el tiempo que le tomó responder la primera vez y se suma el tiempo que le tomó la segunda vez, eso se hace para que el participante pueda tener mejores resultados con respecto a su aprendizaje.

La función Respondo() también es la encargada de recuperar todos los valores ingresados por el participante referente a la pregunta, para ello se crea un objeto de la clase

Pregunta, en el que este objeto recuperará todos esos valores de la pregunta y la pregunta se agregará al objeto de la clase SimuladorData. Para controlar cuando el participante retroceda, se utiliza dos condicionales en el que se dice que si TextoAgregoModif es igual a Nuevo significa que ya respondió la pregunta, pero si TextoAgregoModif es igual Actu significa que el participante ha retrocedido de panel y lo que se hace es llamar al método updatePregunta de la clase SimuladorData. Podemos encontrar las imágenes del código empleado junto a la documentación de todo el funcionamiento del script en los Anexos F.

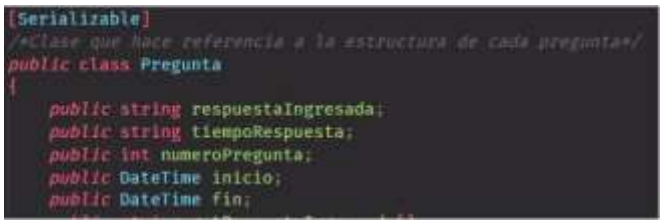
Figura 32. Fragmento de código de la función Respondo () del Simulador Laboral Información Difícil.



```
public void Respondo(TMP_InputField textoBot)
{
    if (textoBot.name == "rop1_inpt")
    {
        if (textoBot.text == alternaJera[0])
        {
            puntajeFinal++;
        }
        respuestasDadas[0] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0] + 1);
    }
    else if (textoBot.name == "rop1dias_inpt")
    {
        if (textoBot.text == alternaDias[0])
        {
            puntajeFinal++;
        }
        respuestasDadas[1] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0] + 7);
    }
    else if (textoBot.name == "rop2_inpt")
    {
        if (textoBot.text == alternaJera[1])
        {
            puntajeFinal++;
        }
        respuestasDadas[2] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1] + 1);
    }
    else if (textoBot.name == "rop2dias_inpt")
    {
        if (textoBot.text == alternaDias[1])
        {
            puntajeFinal++;
        }
        respuestasDadas[3] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1] + 7);
    }
    else if (textoBot.name == "rop3_inpt")
    {
        if (textoBot.text == alternaJera[2])
        {
            puntajeFinal++;
        }
```

Figura 33. Fragmento de código de la función Respondo () del Simulador Laboral El Tiempo.

A la hora de guardar datos y enviarlos a un servidor web es muy importante serializarlos primero, para ello debemos anteponer la palabra Serializable[] a la o las clases que deseamos que sus datos se guarden. En este caso, lo utilizamos para guardar los datos de la clase *Pregunta* y para guardar los datos de la clase *SimuladorData*, la cual hace referencia a los datos del participante junto a las preguntas resueltas, en el que los valores se guardaran en la base de datos del servidor web desde un archivo Json. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script en los Anexos F.



```
[Serializable]
/*Clase que hace referencia a la estructura de cada pregunta*/
public class Pregunta
{
    public string respuestaIngresada;
    public string tiempoRespuesta;
    public int numeroPregunta;
    public DateTime inicio;
    public DateTime fin;
```

Figura 34. Fragmento de código de serialización de la clase Pregunta para el Simulador Laboral Información Difícil y el Simulador Laboral el Tiempo.

Figura 35. Fragmento de código de serialización de la clase SimuladorData para el Simulador Laboral Información Difícil y el Simulador Laboral el Tiempo.

Se crea la función *Post ()* para consumir el servicio web, pasamos la url del servidor web indicando el tipo de método en este caso POST, ya que se va a insertar datos en el servidor web, generamos el archivo Json de mis preguntas, envío el Json y finalmente se realiza la petición al servidor web mediante SendWebRequest().

Figura 36. Función para consumir el servicio web.

7.1.3.2.1.2.MouseColor.cs

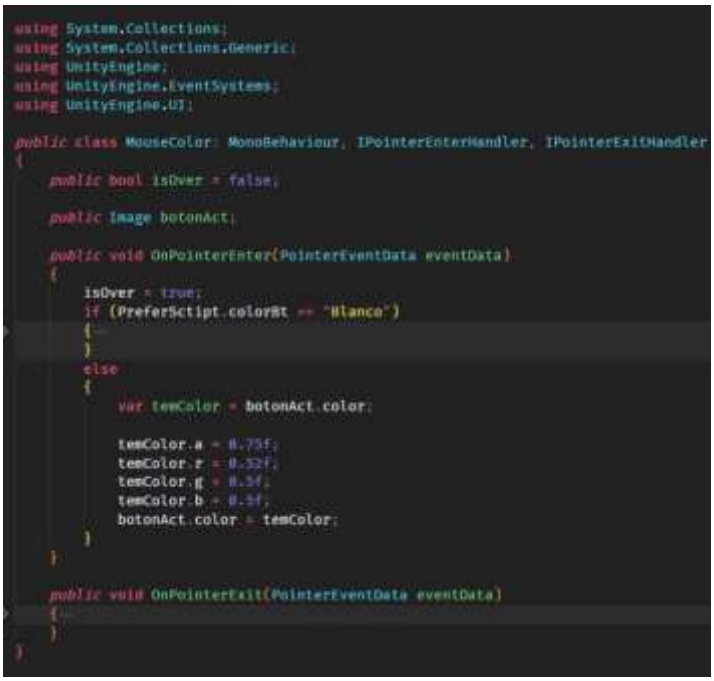


Figura 37. Script para Simulador Información Difícil y Simulador Laboral El Tiempo.

Este script permite resaltar los objetos cuando el mouse este sobre alguno de ellos, es decir, cuando el mouse esté sobre un botón éste cambiara de color. De esta manera, el estudiante sabrá exactamente en donde está posicionado, para ello se ha implementado dos funciones: cambiar de color cuando el mouse esté sobre algún botón (función *OnPointerEnter ()*) y cuando el mouse salga de dicho botón se activará la función *OnPointerExit ()* volviendo al color anterior del botón. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script en los Anexos F.

7.1.3.2.1.3.MouseOver.cs

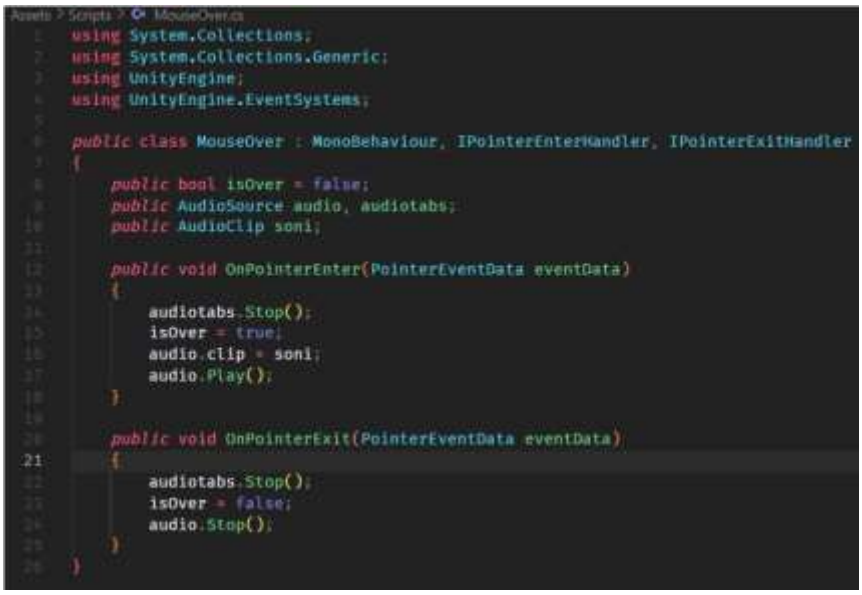


Figura 38. Script para Simulador Laboral Información Difícil y Simulador Laboral El Tiempo.

Este script permite reproducir los audios de los botones principales del simulador laboral (*empezar, salir, atrás, siguiente, preferenciasB_txt*), cuando el mouse esté sobre algún botón reproducirá el audio correspondiente y dejará de reproducir el audio cuando el mouse salga del botón. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script en los Anexos F.

7.1.3.2.1.4.OverTextos.cs

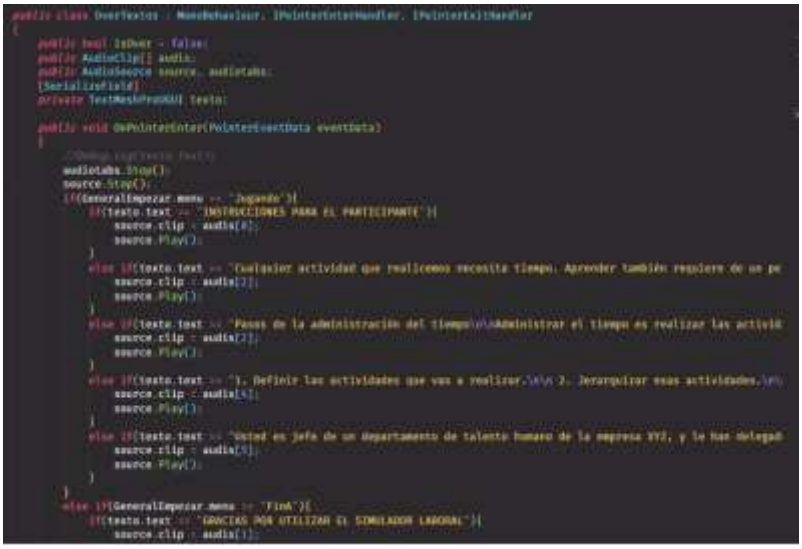
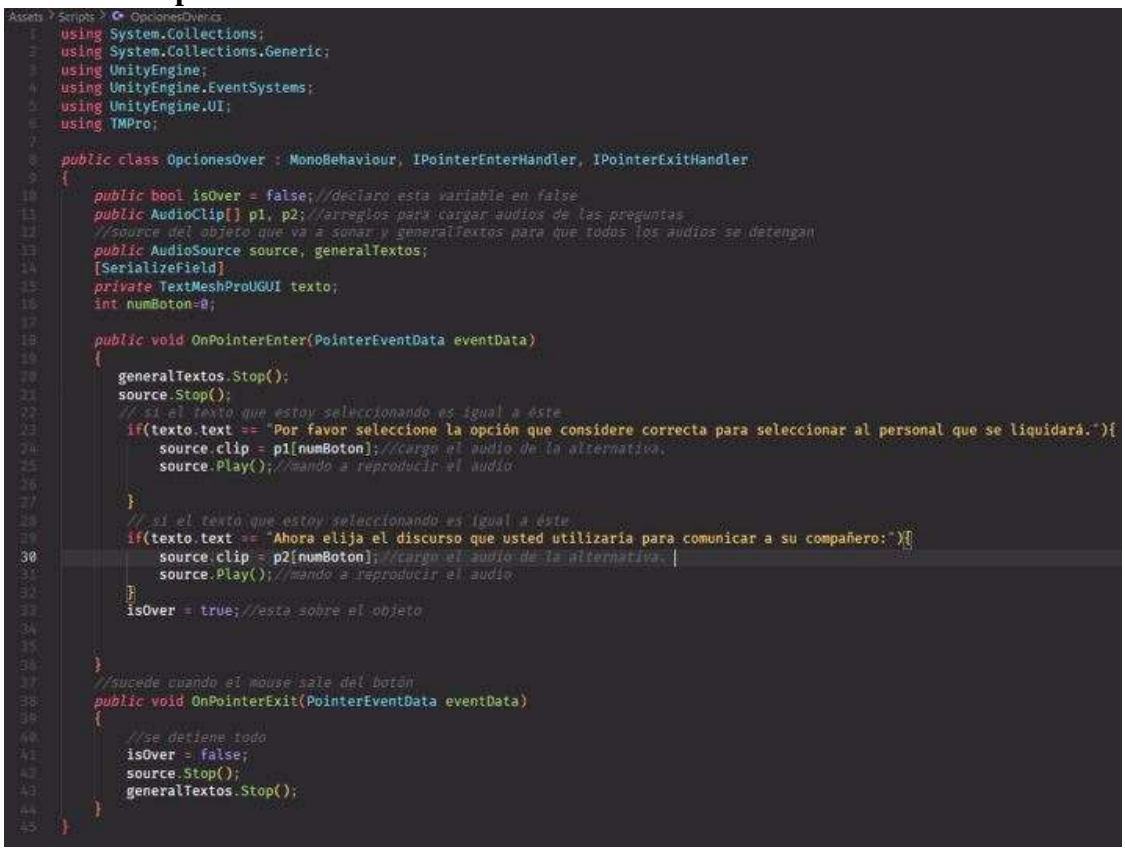


Figura 39. Script para Simulador Laboral Información Difícil y Simulador Laboral El Tiempo.

Este script permite reproducir el audio de un objeto de la escena cuando el mouse esté sobre él, dicho objeto recibe un texto del script GeneralEmpezar.cs. Entonces, para asegurar que se reproduzca el audio correcto vamos a la función OnPointerEnter(), dentro de esta definimos un condicional en el que decimos que reproduzca el audio, solo si el texto que contiene el objeto es igual al que se envió a comparar. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script en los Anexos F.

7.1.3.2.1.5.OpcionesOver.cs



```
Assets > Scripts > OpcionesOver.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;

public class OpcionesOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false;//declaro esta variable en false
    public AudioClip[] p1, p2;//arreglos para cargar audios de las preguntas
    //source del objeto que va a sonar y generalTextos para que todos los audios se detengan
    public AudioSource source, generalTextos;
    [SerializeField]
    private TextMeshProUGUI texto;
    int numBoton=0;

    public void OnPointerEnter(PointerEventData eventData)
    {
       generalTextos.Stop();
       source.Stop();
       // si el texto que estoy seleccionando es igual a éste
        if(texto.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará."){
            source.clip = p1[numBoton];//cargo el audio de la alternativa.
            source.Play();//mando a reproducir el audio

        }
        // si el texto que estoy seleccionando es igual a éste
        if(texto.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:"){
            source.clip = p2[numBoton];//cargo el audio de la alternativa.
            source.Play();//mando a reproducir el audio
        }
        isOver = true;//esta sobre el objeto


    }
    //sucede cuando el mouse sale del botón
    public void OnPointerExit(PointerEventData eventData)
    {
        //se detiene todo
        isOver = false;
        source.Stop();
        generalTextos.Stop();
    }
}
```

Figura 40. Fragmento de código del script OpcionesOver Simulador Laboral Información Difícil.

Este script permitirá que cuando el mouse esté sobre uno de los botones de las alternativas reproduzca un audio y cuando salga de él, detendrá todos los audios. Para ello, dentro de la función OnPointerEnter() usamos un condicional en el que se solicita que reproduzca el audio del botón solo si el texto del enunciado de la pregunta es igual al texto con el que se está comparando.

7.1.3.2.1.6.PreferSctipt.cs

```
Assets > Scripts > PreferSctipt.cs
public class PreferSctipt : MonoBehaviour
{
    [SerializeField]
    private Image[] botones, inputsImagen, activiImagen,imgAct,imgInp;
    [SerializeField]
    private TextMeshProUGUI[] objeto,objPrefer,botPref, pregs, inputs;
    [SerializeField]
    private TextMeshProUGUI[] tiposLetra;
    [SerializeField]
    private TextMeshProUGUI tem1, tem2;
    [SerializeField]
    private Image panel, muteBtn;
    public Button botMute;
    public AudioSource[] sonidoAS;
    public TextMeshProUGUI silbtn;
    float alpha = 215;
    public static string menu="Main";
    public static string colorBt="Negro";
    public static int diag=0, pre=0;
    public static bool isMute = false;
    [SerializeField]
    private Image[] botonesImagenes;
    public Sprite ImagenMute, ImagenMuteS;
    private bool Disle=false;

    public void Cambio(string colpan){...
    }
    public void MuteAll(){...
    }
    public void CambioColBot(string te){...
    }
    public void CambioColText(string te){...
    }
    public void cambioFuente(TMP_FontAsset fu){...
    }
    public void cambioTamano(int tam){...
    }
    public static void MenuSel(string tipoMen){
        Debug.Log("Eras "+PreferSctipt.menu+" ahora eres "+tipoMen);
        PreferSctipt.menu = tipoMen;
    }
    public static void ColB(string col){
        PreferSctipt.colorBt = col;
    }
}
```

Figura 41. Script correspondiente a opciones de accesibilidad del simulador laboral para Simulador Laboral Información Difícil y Simulador Laboral El Tiempo.

Se desarrolla este script con el objetivo de establecer un módulo de accesibilidad para los futuros simuladores laborales que planteen los colaboradores del proyecto Edutech. Para dicho módulo se implementa varias funciones, las cuales permitirán al estudiante personalizar el simulador laboral de acuerdo a sus necesidades y capacidades: para las opciones de contraste se emplea tres funciones las cuales trabajan en conjunto (*Cambio(), CambioColBot(), CambioColText()*), con las cuales podrá cambiar el color de fondo de la pantalla, el color de los botones y el color del texto. Para la opción de activar o desactivar la reproducción de audios se implementa la función *MuteAll().* Para la opción de tipos de fuentes se implementa la función *cambioFuente()*, para ello se tiene tres opciones de fuentes. Para la opción de aumentar o reducir el tamaño de texto se implementa la función *cambioTamaño()*. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script en los Anexos F.

7.1.3.2.1.7.WebData.cs

```
using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {

    [DllImport("__Internal")]
    private static extern int GetUrlVal();

    [DllImport("__Internal")]
    private static extern int GetEjer();

    [DllImport("__Internal")]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {

    }
    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text+="-- "+id;
        return  id;
    }

    public string Mail(){
    }

    public int Ejercitario(){
    }
}
```

Figura 42. Fragmento de código del script WebData.

En este script se definen las funciones get para consumir los servicios web, en el que se obtiene el ID del participante, el correo y el ID del ejercitario.

7.1.3.3. Sistema de información en el servidor web

A continuación, se muestra la arquitectura que se planteó para la comunicación del servicio web con la información de los simuladores laborales desarrollados en Unity.

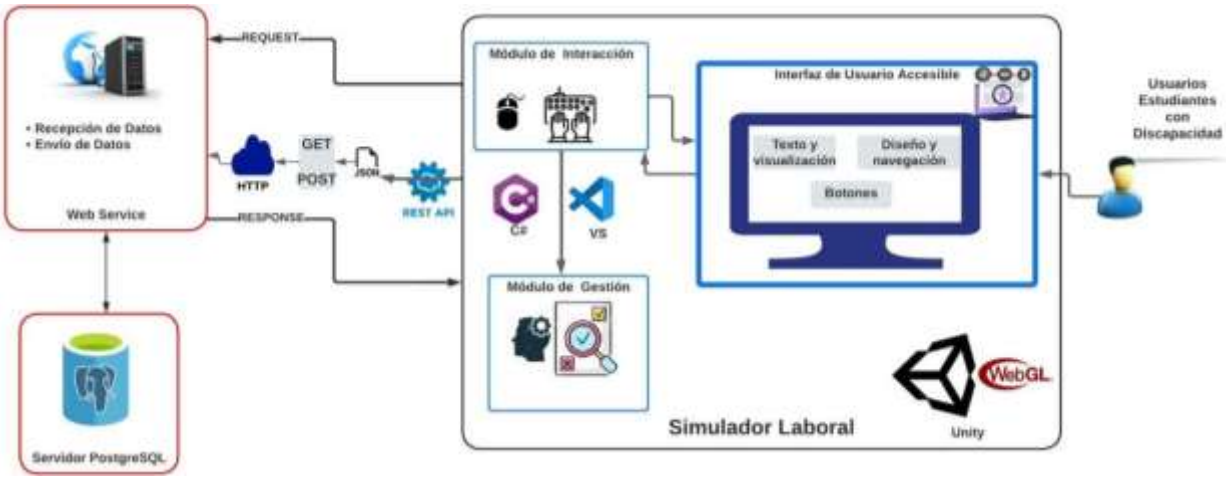
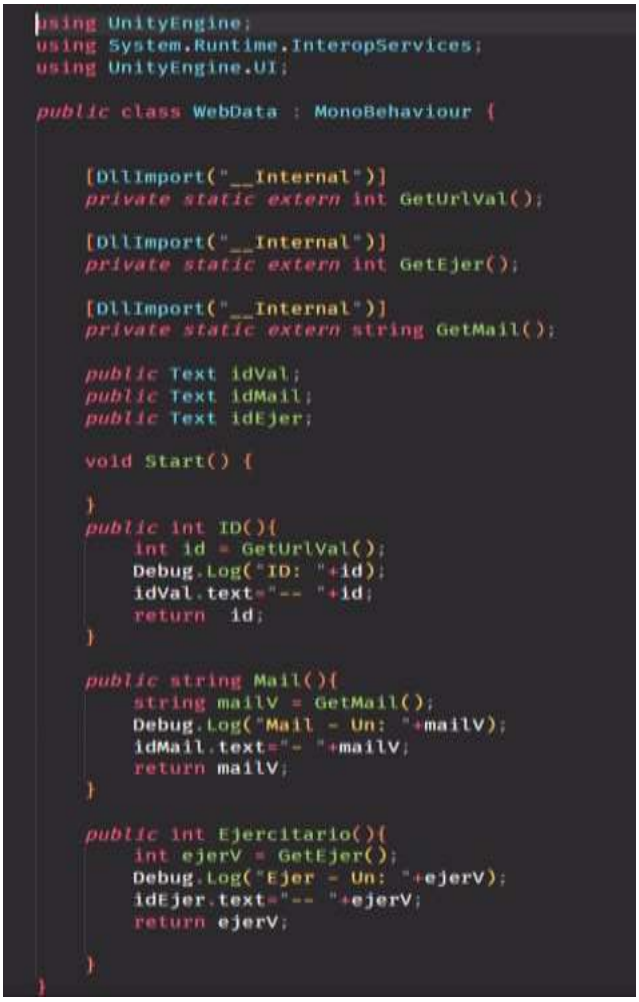


Figura 43. Arquitectura de funcionamiento del Servicio Web con los simuladores laborales.

Para el consumo de servicios de la API del servidor web de Simuladores Laborales 3D del proyecto Edutech, se utiliza un archivo que contiene código JavaScript, el código corresponde a las fuentes de JavaScript, las cuales permitirán la comunicación de Unity con el servidor web, al archivo lo llamamos *Plugin.jslib* y lo colocamos en una carpeta llamada *Plugins* dentro de la carpeta Assets del proyecto.

Figura 44. Plugin que tiene el código JavaScript para comunicación de Unity con el servidor web.

Luego se implementa el script *WebData.cs*, el cual contiene el código para llamar a dichas funciones. Podemos encontrar las imágenes del código empleado junto a la documentación completa del script y demás pasos empleados en los Anexos F.



```
using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {


    [DllImport("__Internal")]
    private static extern int GetUrlVal();

    [DllImport("__Internal")]
    private static extern int GetEjer();

    [DllImport("__Internal")]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {

    }
    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text="-- "+id;
        return  id;
    }

    public string Mail(){
        string mailV = GetMail();
        Debug.Log("Mail - Un: "+mailV);
        idMail.text="- "+mailV;
        return mailV;
    }

    public int Ejercitario(){
        int ejerV = GetEjer();
        Debug.Log("Ejer - Un: "+ejerV);
        idEjer.text="-- "+ejerV;
        return ejerV;

    }
}
```

Figura 45. Código para llamar a funciones JavaScript del archivo Plugin.jslib desde C#.

VIII. Resultados

En este apartado se describe la fase de evaluación de accesibilidad según la metodología DCU.

8.1. Pruebas y encuestas de simuladores laborales

Para la evaluación de los simuladores laborales 3D, se realizaron pruebas de experimentación con 7 participantes en las instalaciones de la cámara de Gesell de la Universidad Politécnica Salesiana bajo la supervisión de la Ing. Paola Ingavélez tutora del proyecto. En el que, la cámara de Gesell es una habitación conformada por dos ambientes separados por un vidrio de visión unilateral, la habitación cuenta con un equipo especializado de audio y video para la grabación de los diferentes experimentos, los equipos permiten grabar. De esta manera, el evaluador podrá revisar las grabaciones las veces que

desee hasta lograr definir las conclusiones de cada experimento, en el que finalmente se estableció varios escenarios, debido a que cada participante tuvo diferentes puntos de vista frente a cada reto planteado. Durante las pruebas, los participantes interactuaron con los simuladores laborales alojados en la nube del servidor web del proyecto Edutech, a cada participante se le presentó la respectiva introducción acerca del proyecto Edutech, el objetivo de las pruebas planteadas, el objetivo de los simuladores laborales, las mecánicas y dinámicas de los mismos.

Las encuestas se realizaron con el fin de determinar la facilidad de interacción, navegabilidad, usabilidad y accesibilidad del usuario final con los simuladores laborales a partir de los puntos de vista, reacciones relacionadas a la interacción con los simuladores laborales 3D, para ello se planteó algunos retos en preguntas. Con la intención de analizar los datos, se realizaron pruebas con un grupo de estudiantes universitarios y personal de la Universidad Politécnica Salesiana, con el fin de determinar la reacción de cada uno de ellos frente a los simuladores laborales. En la Tabla 8 se muestra información referente a los participantes involucrados en el desarrollo de las encuestas.

Nombre	**Apellido**	**Edad**	**Posee Discapacidad**	**Profesión/Carrera**	**Fecha**
Noé	Ayavaca	28	No	Estudiante de Computación	30/06/2022
Roberto	Pacho	28	No	Estudiante de Ing. Sistemas	30/06/2022
Fernando	Déleg	26	No	Estudiante de Ing. Sistemas	30/06/2022
Gabriel	Chuchuca	24	No	Estudiante de Ing. Sistemas	01/07/2022
Michelle	Peñalosa	25	No	Estudiante de Ing. Sistemas	01/07/2022
Fabián	Armijos	28	No	Estudiante de Ing. Sistemas	01/07/2022
José	Tenesaca	42	Discapacidad Visual	Auxiliar Biblioteca UPS	13/07/2022

Tabla 8. Información de participantes.

Figura 46. Cámara de Gesell UPS- Monitoreo de interacción de usuario con los simuladores laborales 3D.

El tiempo promedio empleado por los participantes en resolver los retos planteados fue de 30 minutos, teniendo en cuenta que ninguno de ellos estaba familiarizado con ese tipo de simuladores. Se puede revisar la información completa y fotos de las encuestas realizadas en el Anexo A.

Luego de haber procesado la información obtenida de las evaluaciones y encuestas a los participantes involucrados los resultados son los siguientes:

1. **Primera pregunta: Interacción con ratón para rotación de la cámara sobre la escena.**

Cada participante deberá interactuar con la rotación de la cámara sobre la escena a través del mouse.

Tiempo Empleado		
Participante	**Simulador Laboral El Tiempo**	**Simulador Laboral Información Difícil**
Participante 1	0:10	0:12
Participante 2	0:07	0:10
Participante 3	0:12	0:09
Participante 4	0:10	0:12
Participante 5	0:13	0:15
Participante 6	0:06	0:10
Participante 7	No aplica	No aplica

Tabla 9. Tiempo empleado en realizar la tarea 1.

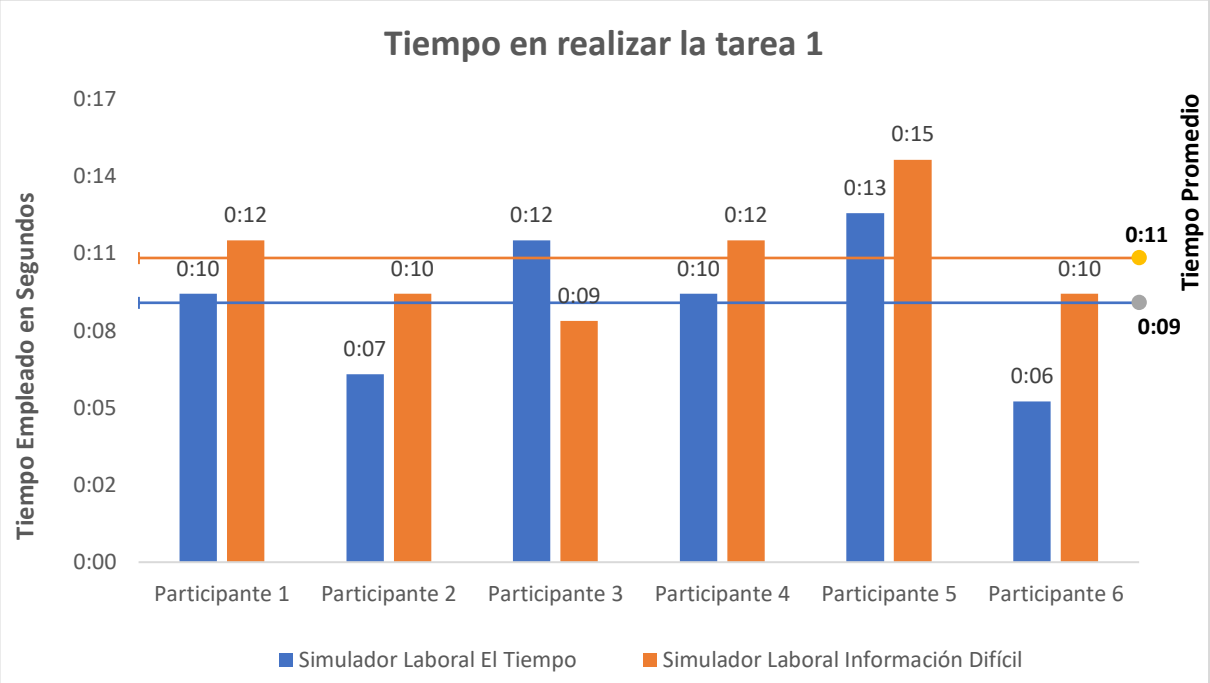


Figura 47. Gráfico de barras del tiempo que le tomó a cada participante realizar la tarea asignada.

Interpretación.

La evaluación de accesibilidad se realizó mientras los participantes interactuaban con los simuladores laborales, la evaluación constaba de 10 retos planteados para el Simulador Laboral Información Difícil y 12 retos planteados para el Simulador Laboral El Tiempo, en el que se llevó un registro del tiempo que le tomó a cada participante realizar cada uno de los retos planteados.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 9 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 11 segundos (Figura 47).

2. **Segunda pregunta: Acceder al simulador a través del botón empezar.**

Los participantes ingresaron al simulador laboral al presionar el botón empezar, el cual se muestra en pantalla, ya sea por teclado o por el mouse.

Participantes	Tiempo empleado por mouse		Tiempo empleado por teclado	
	El Tiempo	Información Difícil	El Tiempo	Información Difícil
Participante 1	0:02	0:03	0:02	0:02
Participante 2	0:03	0:02	0:04	0:03
Participante 3	0:02	0:02	0:03	0:03
Participante 4	0:03	0:02	0:03	0:02
Participante 5	0:02	0:02	0:03	0:02
Participante 6	0:02	0:02	0:03	0:03
Participante 7	No aplica	No aplica	0:04	0:03

Tabla 10. Tiempo empleado en realizar la tarea 2.

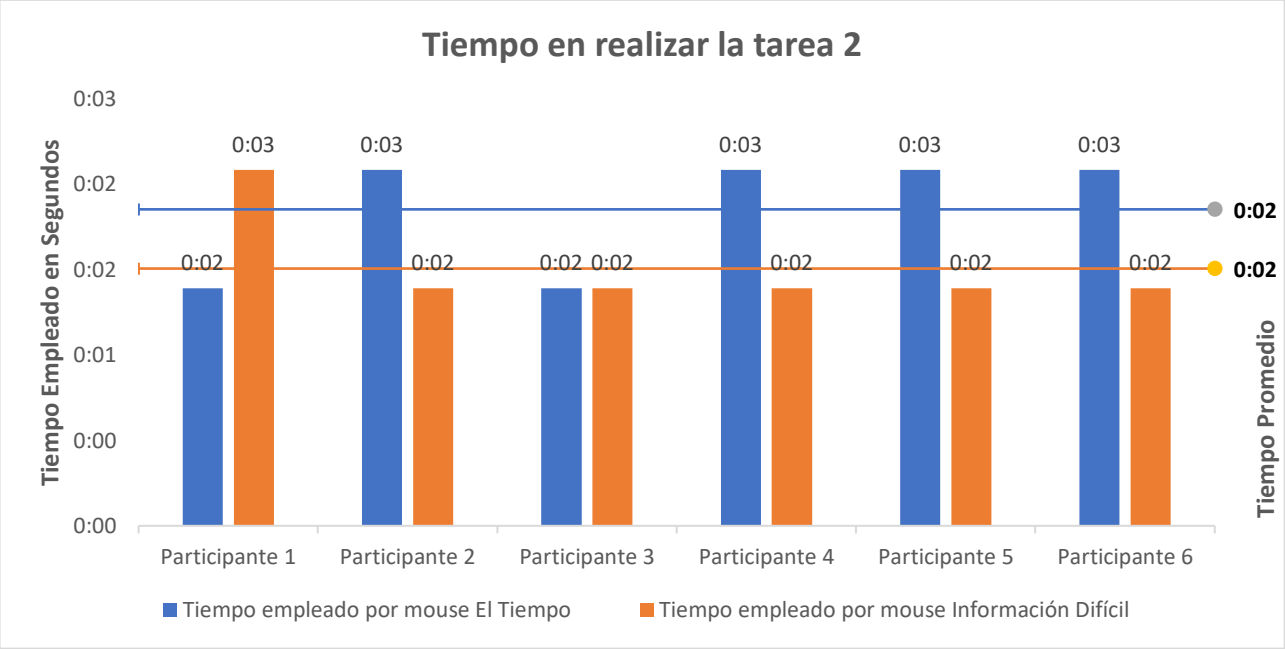


Figura 48. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 2 segundos para los dos simuladores laborales (Figura 48).

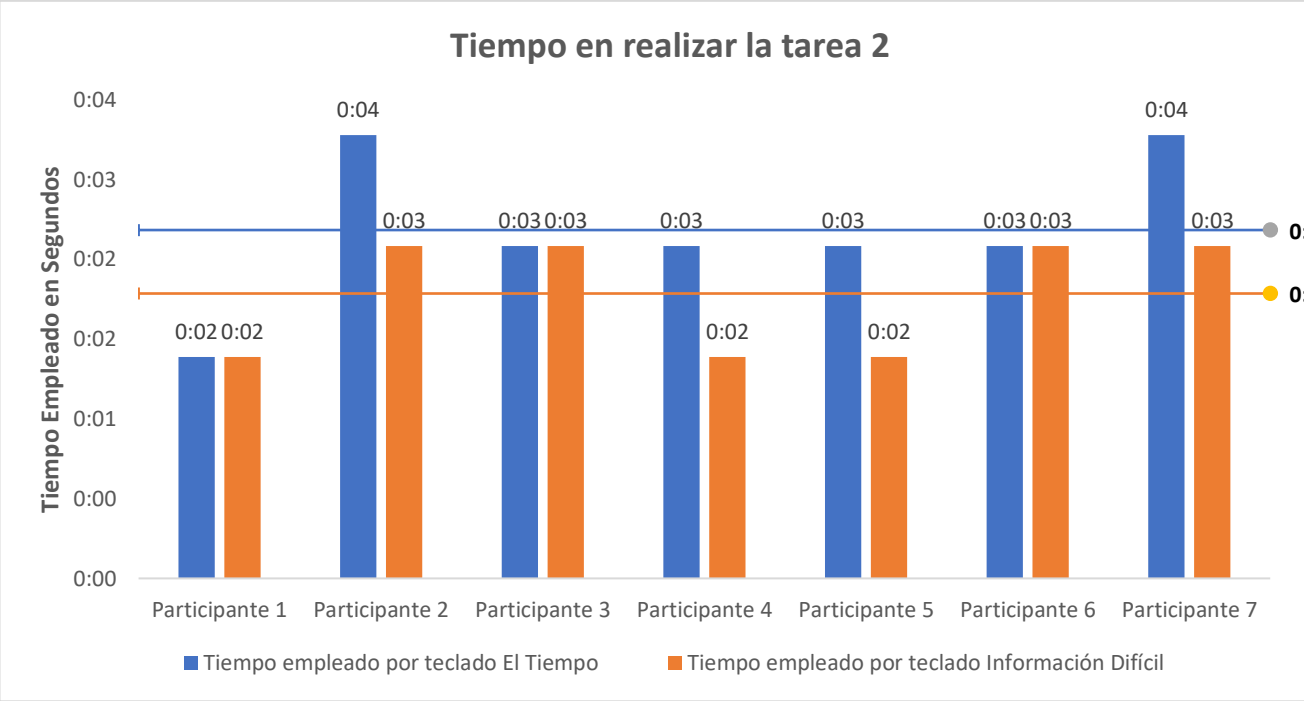


Figura 49. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 3 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 2 segundos (Figura 49).

3. Tercera pregunta: Acceder a la barra de preferencias.

Los participantes accederieron a la barra de preferencias del simulador laboral al presionar el botón preferencias, ya sea por teclado o por el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Dificil**
Participante 1	0:02	0:03	0:04	0:03
Participante 2	0:03	0:02	0:03	0:03
Participante 3	0:02	0:02	0:04	0:03
Participante 4	0:03	0:02	0:04	0:04
Participante 5	0:03	0:02	0:03	0:03
Participante 6	0:03	0:02	0:04	0:03
Participante 7	No aplica	No aplica	0:05	0:05

Tabla 11. Tiempo empleado en realizar la tarea 3.

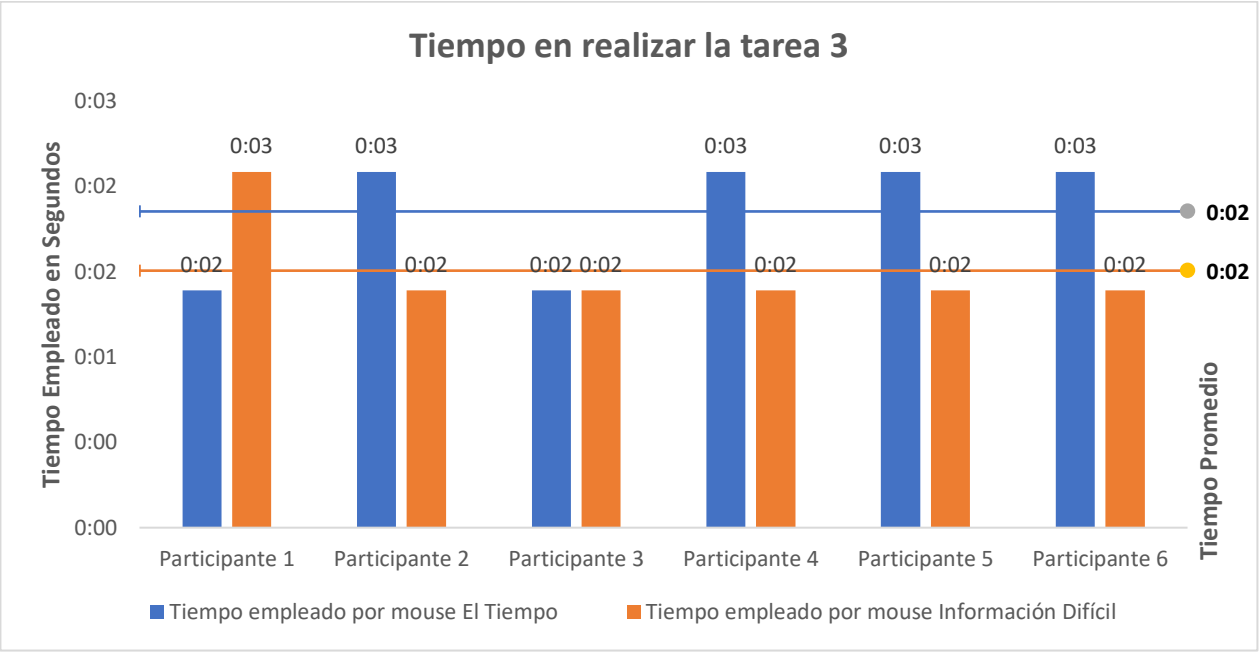


Figura 50. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 2 segundos para los dos simuladores laborales (Figura 50).

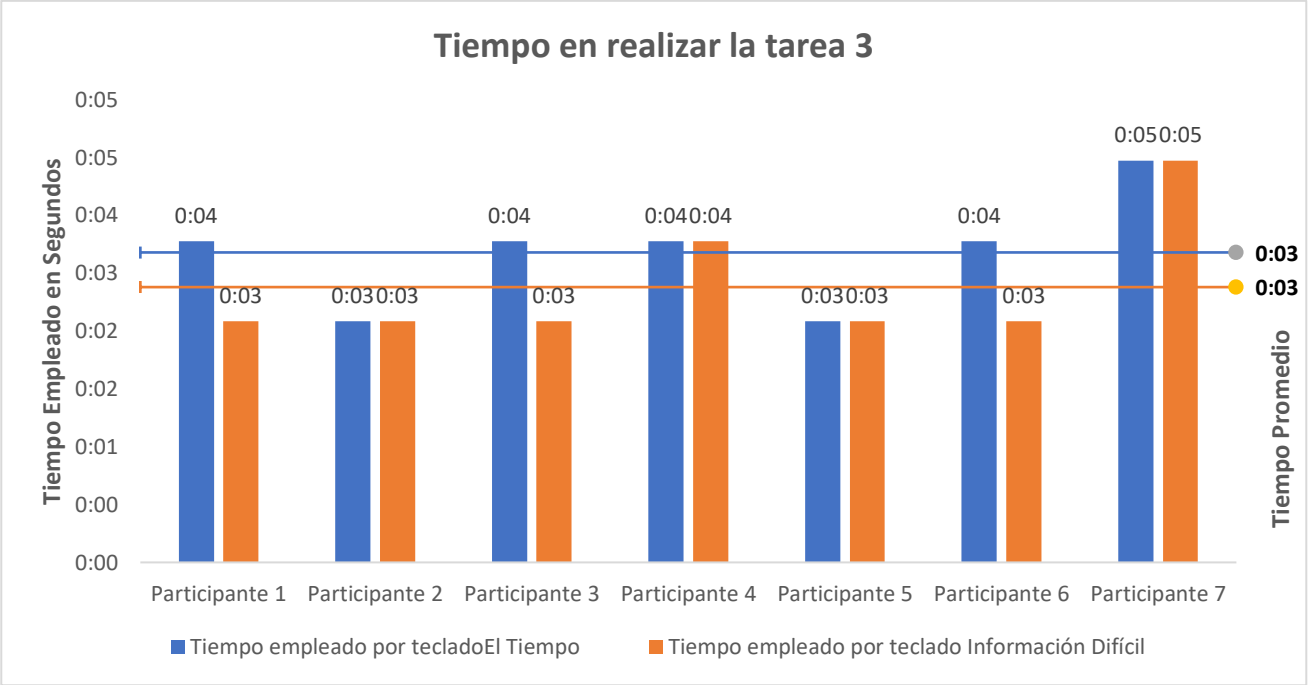


Figura 51. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 3 segundos para los dos simuladores laborales (Figura 51).

4. **Cuarta pregunta: Prueba las diferentes opciones de preferencias (cambiar tipo de fuentes, contrastes, activar o desactivar audio, aumentar o disminuir tamaño de letra).**

Los participantes interactuaron con las distintas opciones de accesibilidad que muestra el simulador laboral a través del teclado o el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Difícil**
Participante 1	0:30	0:31	0:13	0:14
Participante 2	0:29	0:31	0:14	0:14
Participante 3	0:28	0:28	0:13	0:12
Participante 4	0:31	0:30	0:14	0:13
Participante 5	0:28	0:27	0:15	0:14
Participante 6	0:32	0:30	0:13	0:11

Tabla 12. Tiempo empleado en realizar la tarea 4.

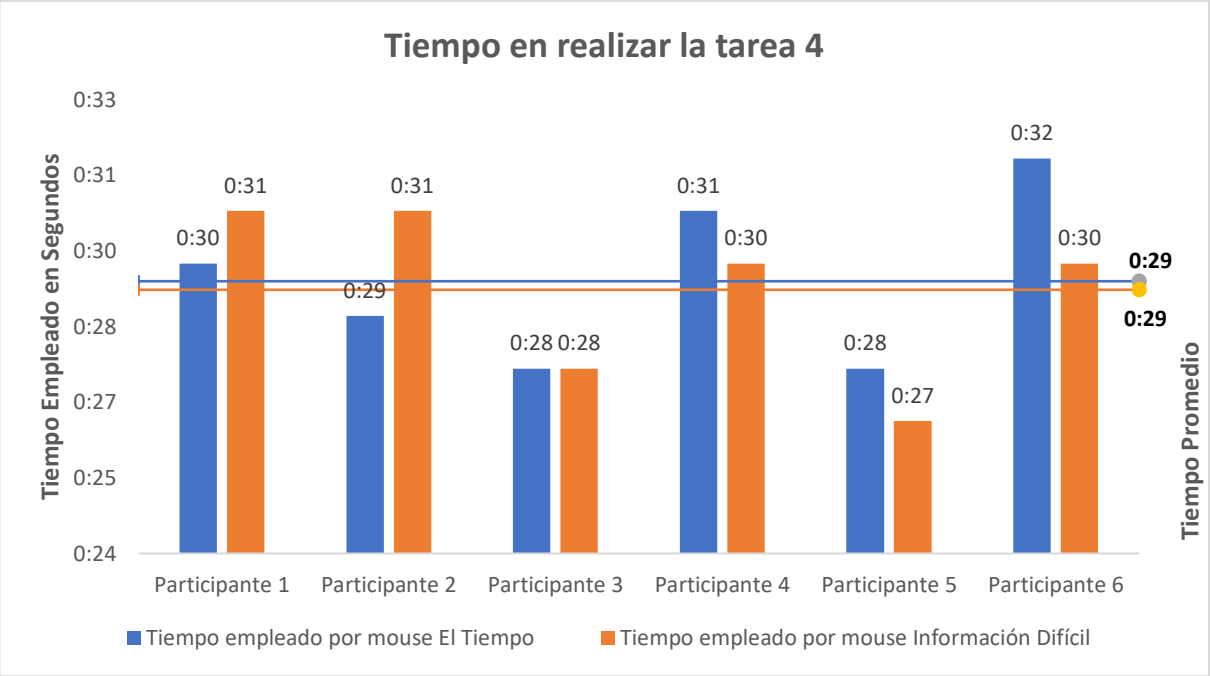


Figura 52. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 29 segundos para los dos simuladores laborales (Figura 52).

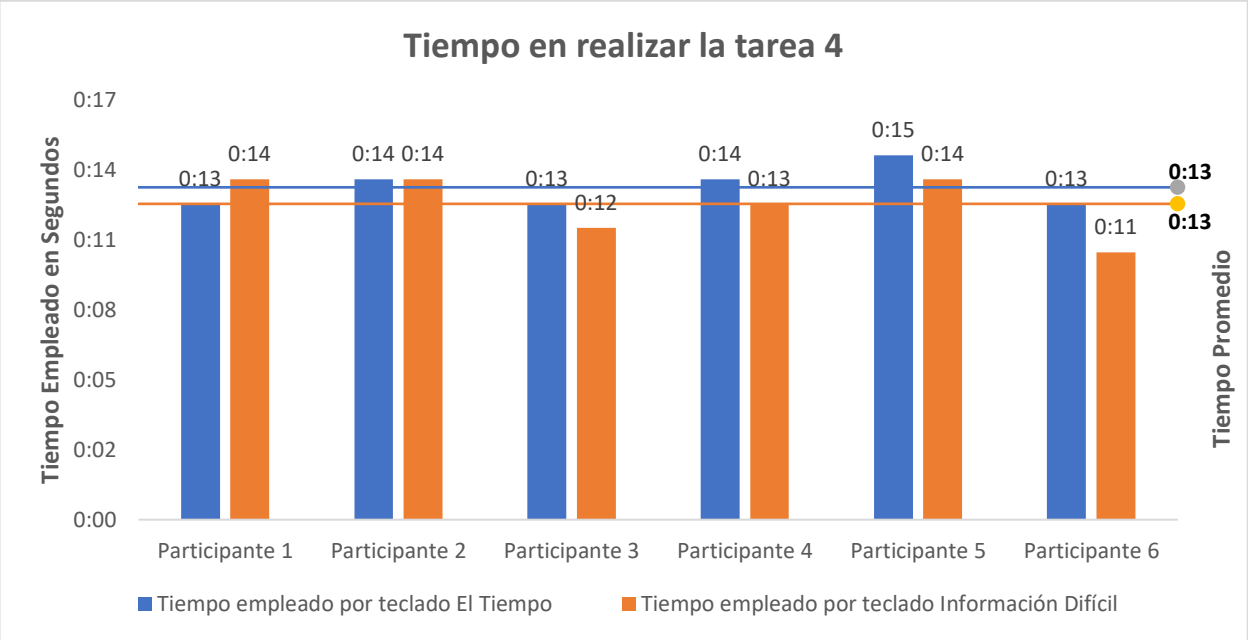


Figura 53. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 13 segundos para los dos simuladores laborales (Figura 53).

5. **Quinta pregunta: Manejo de interfaz únicamente con el teclado.**

Los participantes navegaron e interactuaron con el simulador laboral por medio del teclado.

Participante	**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**
Participante 1	2:40	2:00
Participante 2	3:00	2:30
Participante 3	2:50	2:00
Participante 4	2:45	2:05
Participante 5	2:55	2:10
Participante 6	3:00	2:20
Participante 7	4:00	3:00

Tabla 13. Tiempo empleado en realizar la tarea 5.

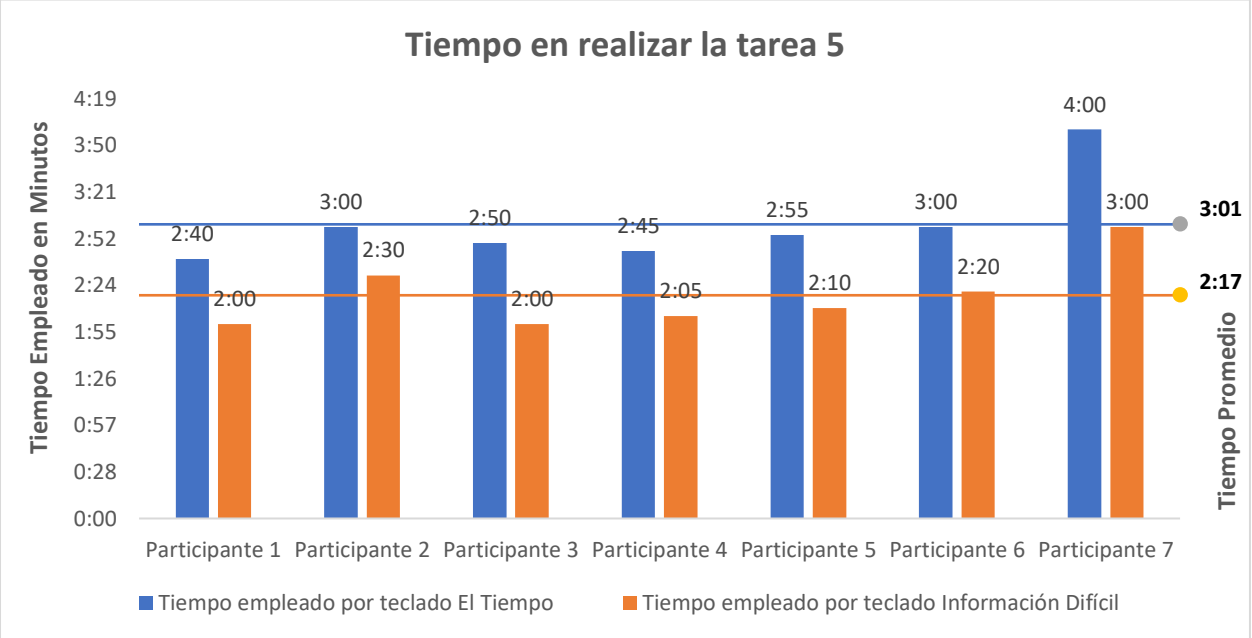


Figura 54. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 3 minutos y 1 segundo. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 2 minutos y 17 segundos (Figura 54).

6. Sexta Pregunta: Manejo de interfaz únicamente con el ratón.

Los participantes navegaron e interactuaron con el simulador laboral por medio del mouse.

Participante	**Tiempo empleado por mouse**	
	El Tiempo	**Información Difícil**
Participante 1	2:00	1:40
Participante 2	2:20	2:00
Participante 3	2:10	1:50
Participante 4	2:00	1:40
Participante 5	2:20	2:10
Participante 6	2:30	2:00
Participante 7	No aplica	No aplica

Tabla 14. Tiempo empleado en realizar la tarea 6.

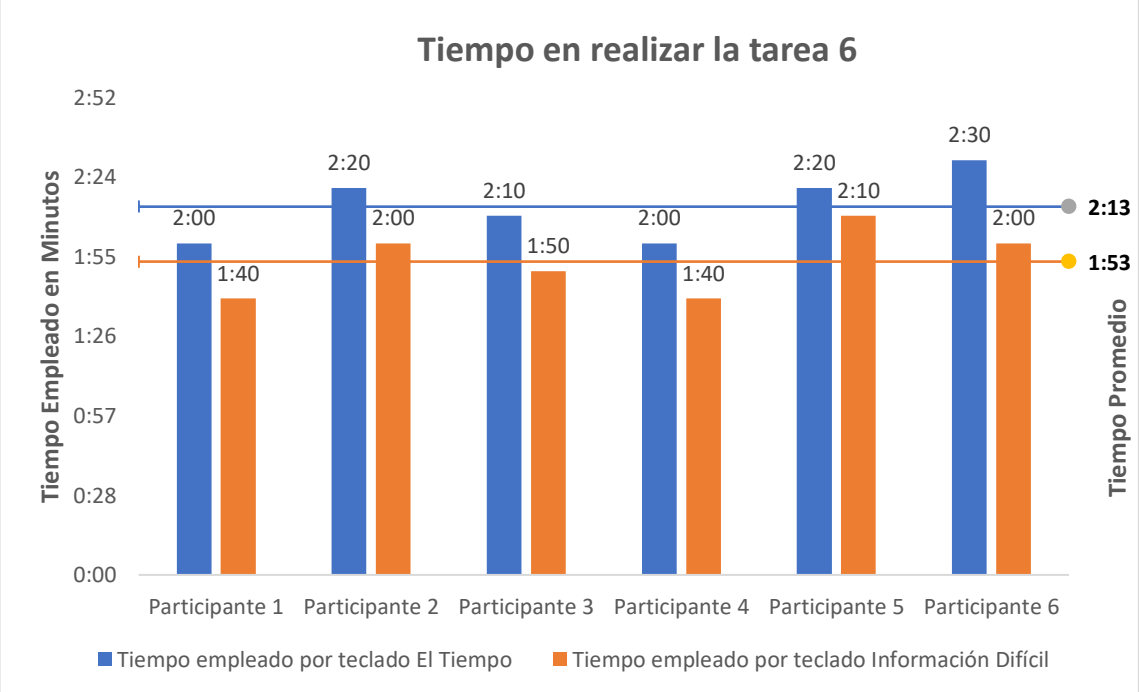


Figura 55. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 2 minutos y 13 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 1 minuto y 53 segundos (Figura 55).

7. **Séptima pregunta: Retroceder a través del botón "atrás".**

Los participantes retrocedieron entre pantallas al presionar el botón atrás, ya sea por teclado o por el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Difícil**
Participante 1	0:01	0:01	0:10	0:10
Participante 2	0:02	0:02	0:12	0:12
Participante 3	0:01	0:01	0:13	0:12
Participante 4	0:01	0:01	0:10	0:13
Participante 5	0:02	0:02	0:10	0:11
Participante 6	0:01	0:01	0:10	0:11
Participante 7	No aplica	No aplica	0:17	0:17

Tabla 15. Tiempo empleado en realizar la tarea 7.

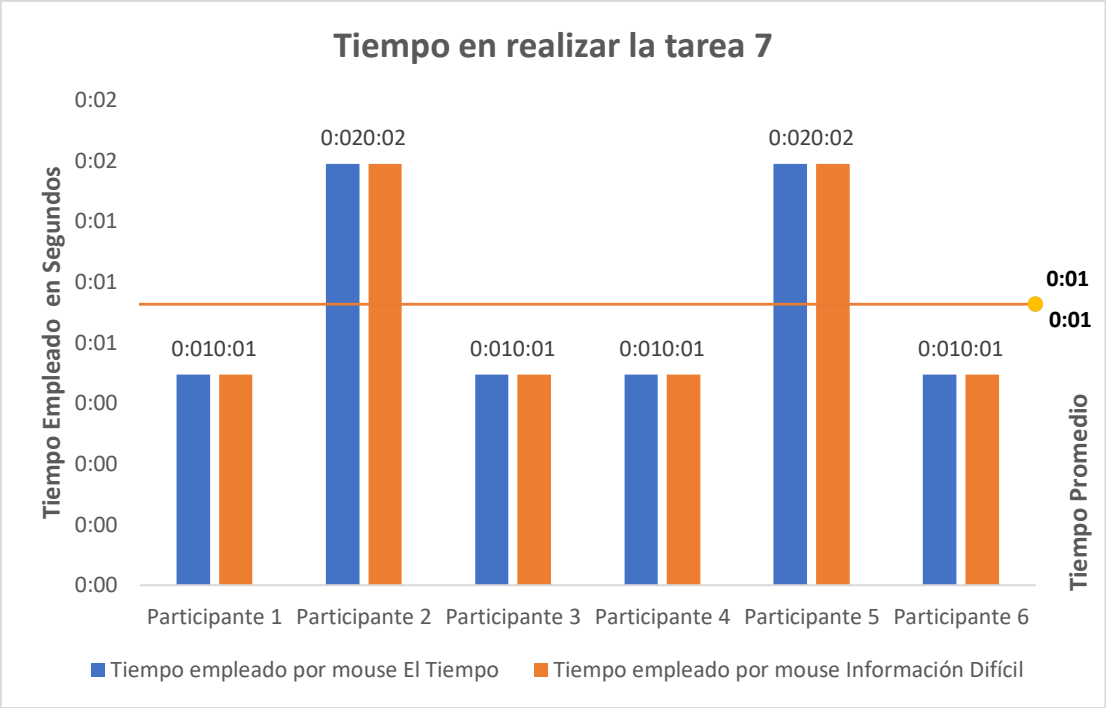


Figura 56. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 1 segundo para los dos simuladores laborales (Figura 56).

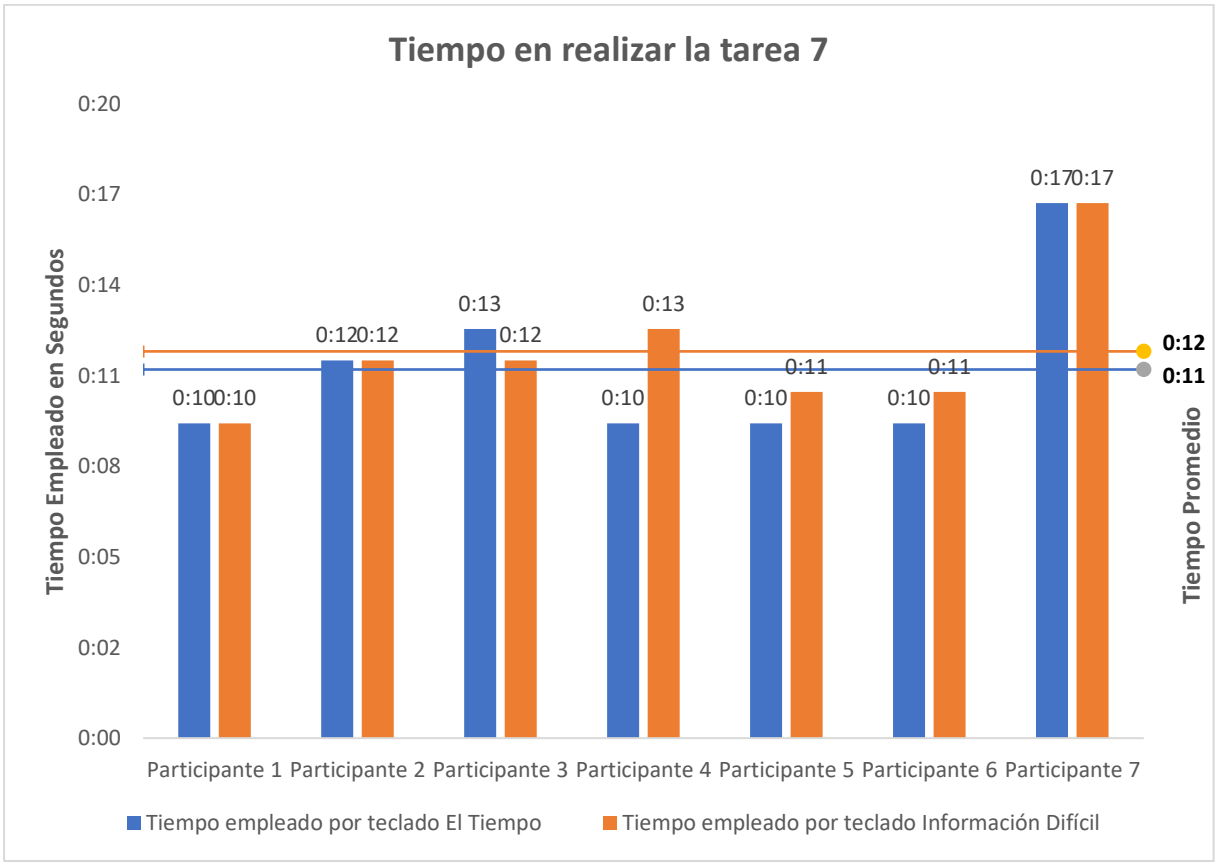


Figura 57. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En este reto planteado del Simulador Laboral El Tiempo se obtiene un tiempo promedio de 11 segundos y en el caso del Simulador Laboral Información Difícil se obtiene un tiempo promedio de 12 segundos (Figura 57).

8. Octava pregunta: Avanzar a través del botón "siguiente".

Los participantes avanzaron entre pantallas al presionar el botón siguiente a través del teclado o el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Difícil**
Participante 1	0:01	0:01	0:11	0:10
Participante 2	0:02	0:02	0:12	0:12
Participante 3	0:01	0:01	0:13	0:12
Participante 4	0:01	0:01	0:14	0:13
Participante 5	0:02	0:02	0:12	0:11

Participante 6	0:01	0:01	0:12	0:11
Participante 7	No aplica	No aplica	0:18	0:17

Tabla 16. Tiempo empleado en realizar la tarea 8.

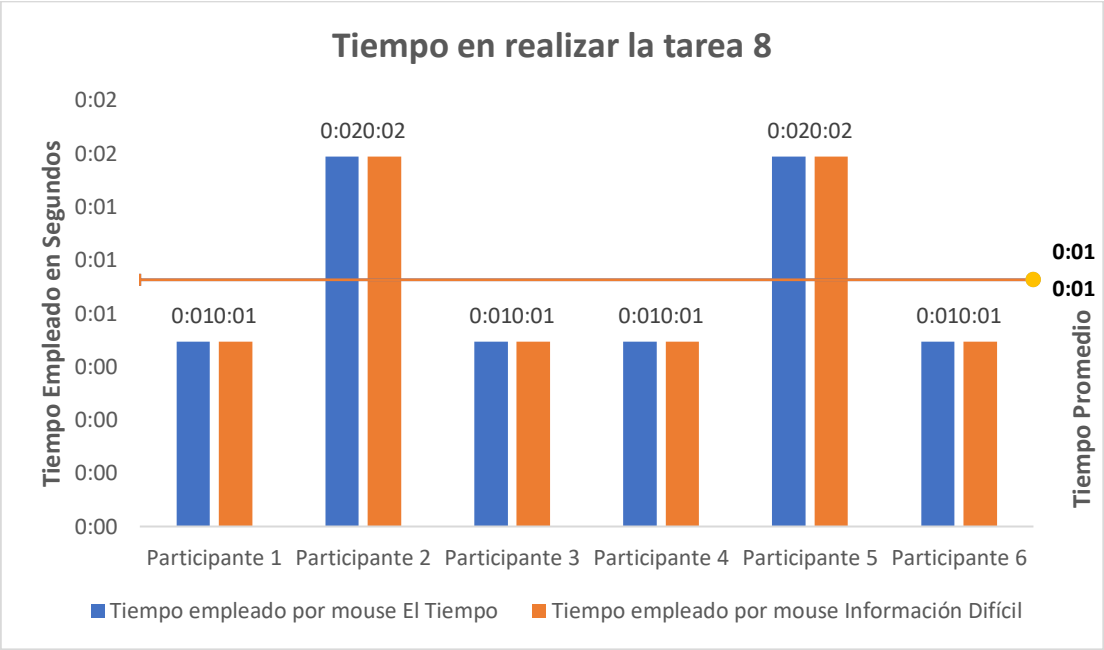


Figura 58. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado, los participantes emplearon un tiempo promedio de 1 segundo para los dos simuladores laborales (Figura 58).

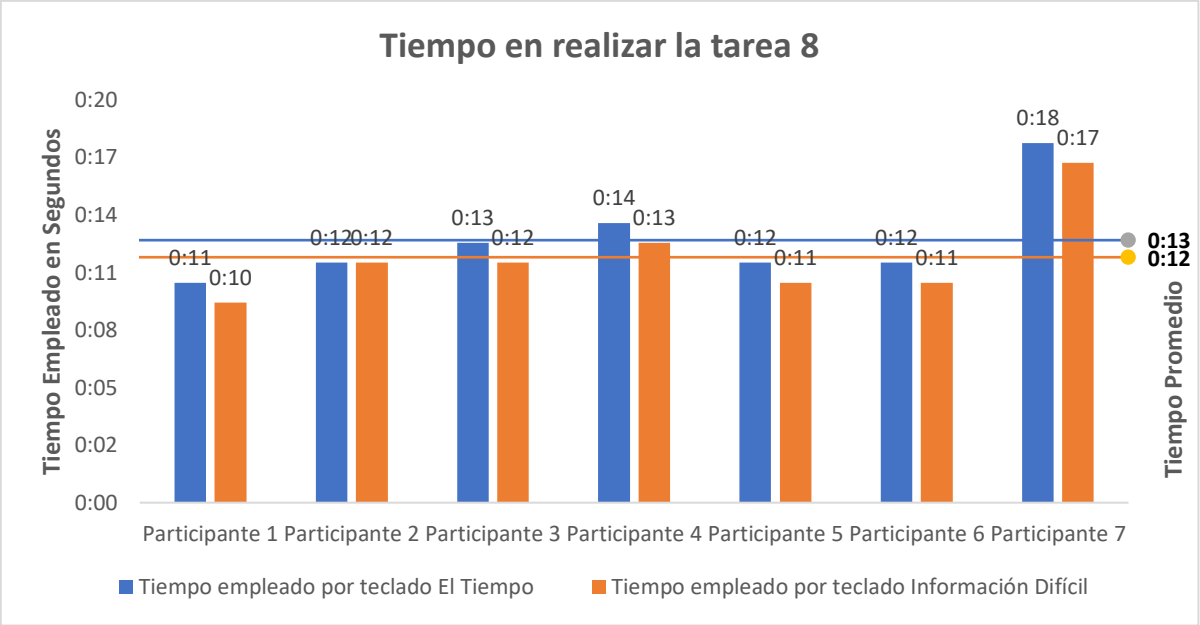


Figura 59. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 13 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 12 segundos (Figura 59).

9. Novena pregunta: Revisión de mensaje de calificación obtenida.

Los participantes accedieron a la pantalla de calificación luego de finalizar las actividades propuestas, ya sea por teclado o el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Difícil**
Participante 1	0:25	0:12	0:25	0:12
Participante 2	0:24	0:10	0:24	0:10
Participante 3	0:27	0:13	0:27	0:13
Participante 4	0:25	0:12	0:25	0:12
Participante 5	0:24	0:13	0:24	0:13
Participante 6	0:25	0:14	0:25	0:14
Participante 7	No aplica	No aplica	0:35	0:28

Tabla 17. Tiempo empleado en realizar la tarea 9.

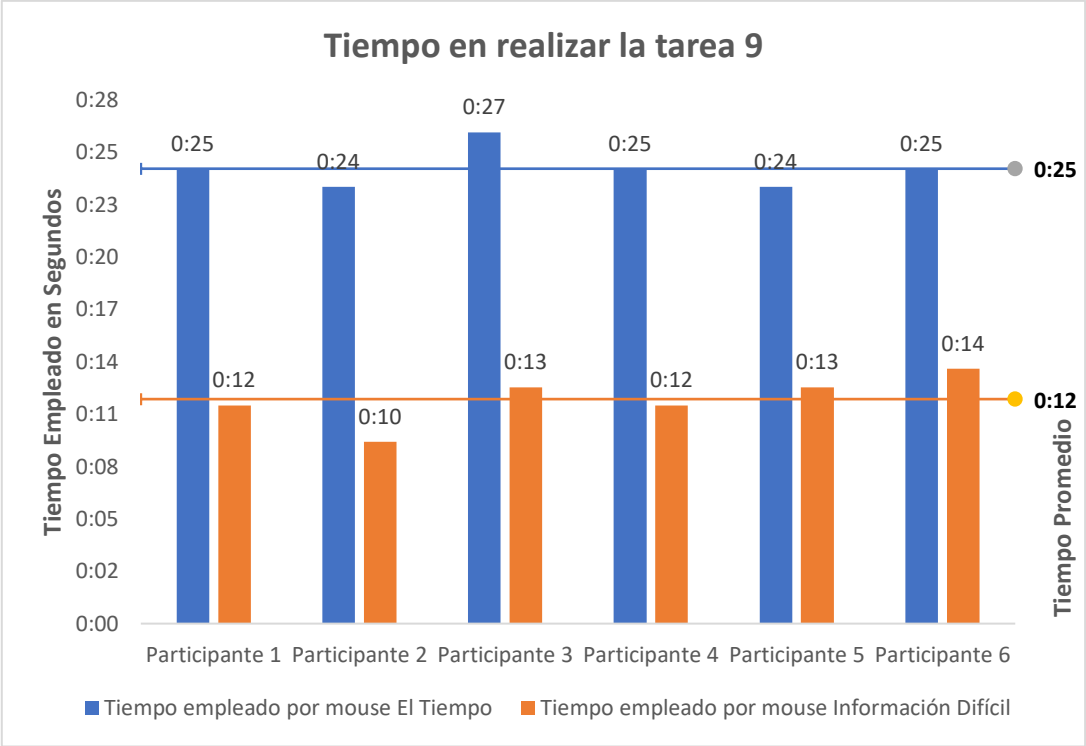


Figura 60. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 25 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 12 segundos (Figura 59).

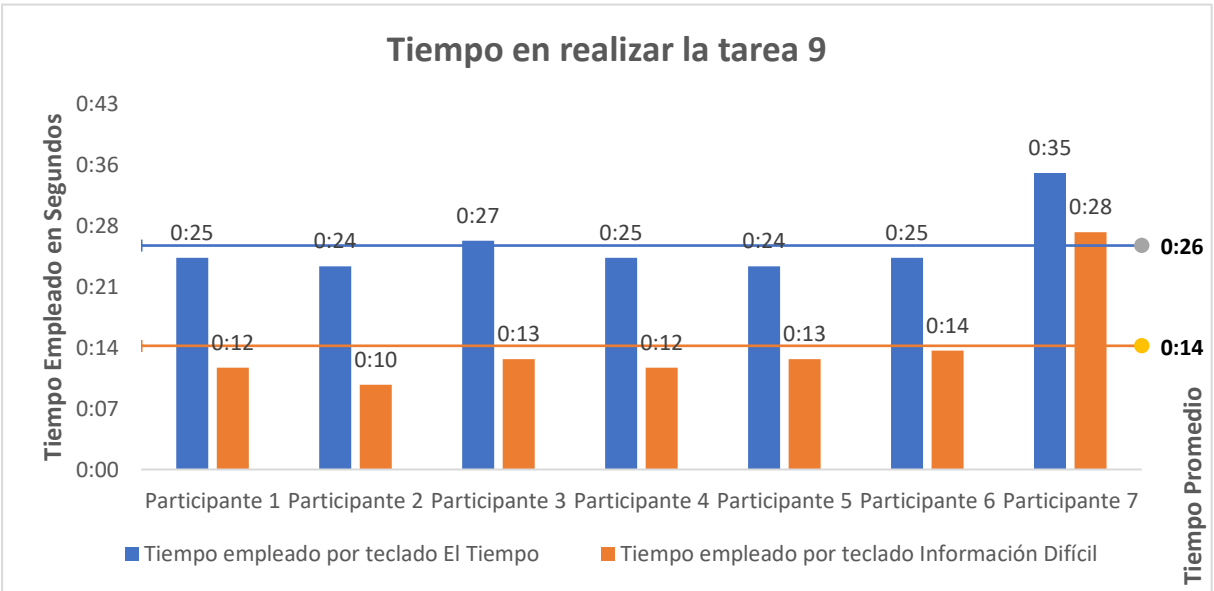


Figura 61. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 26 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 14 segundos (Figura 61).

10. Décima pregunta: Revisión de mensaje de retroalimentación.

Los participantes recibieron la respectiva retroalimentación luego de revisar la pantalla de calificación obtenida a través del teclado o el mouse.

Participante	**Tiempo empleado por mouse**		**Tiempo empleado por teclado**	
	El Tiempo	**Información Difícil**	**El Tiempo**	**Información Difícil**
Participante 1	0:33	1:01	0:35	1:04
Participante 2	0:34	1:03	0:36	1:05
Participante 3	0:32	1:04	0:34	1:06
Participante 4	0:34	1:05	0:36	1:07
Participante 5	0:30	1:01	0:32	1:04
Participante 6	0:35	1:03	0:38	1:06
Participante 7	No aplica	No aplica	0:44	1:20

Tabla 18. Tiempo empleado en realizar la tarea 10.

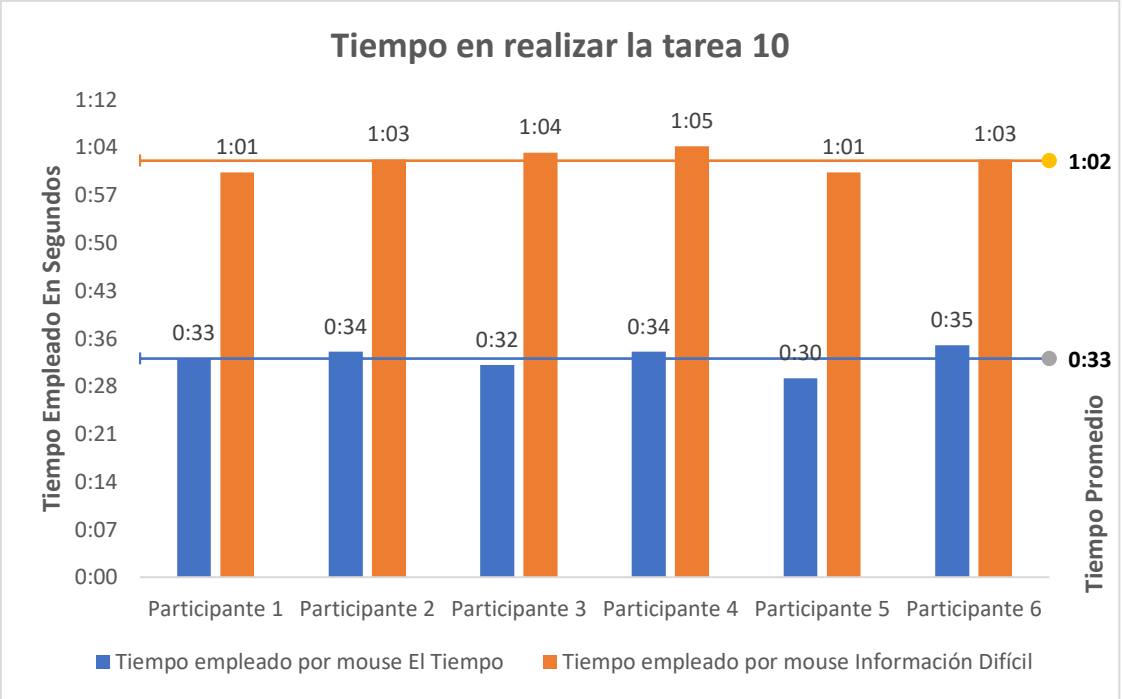


Figura 62. Tiempo empleado por cada participante en realizar la tarea por mouse.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 33 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 1 minuto y 2 segundos (Figura 62).

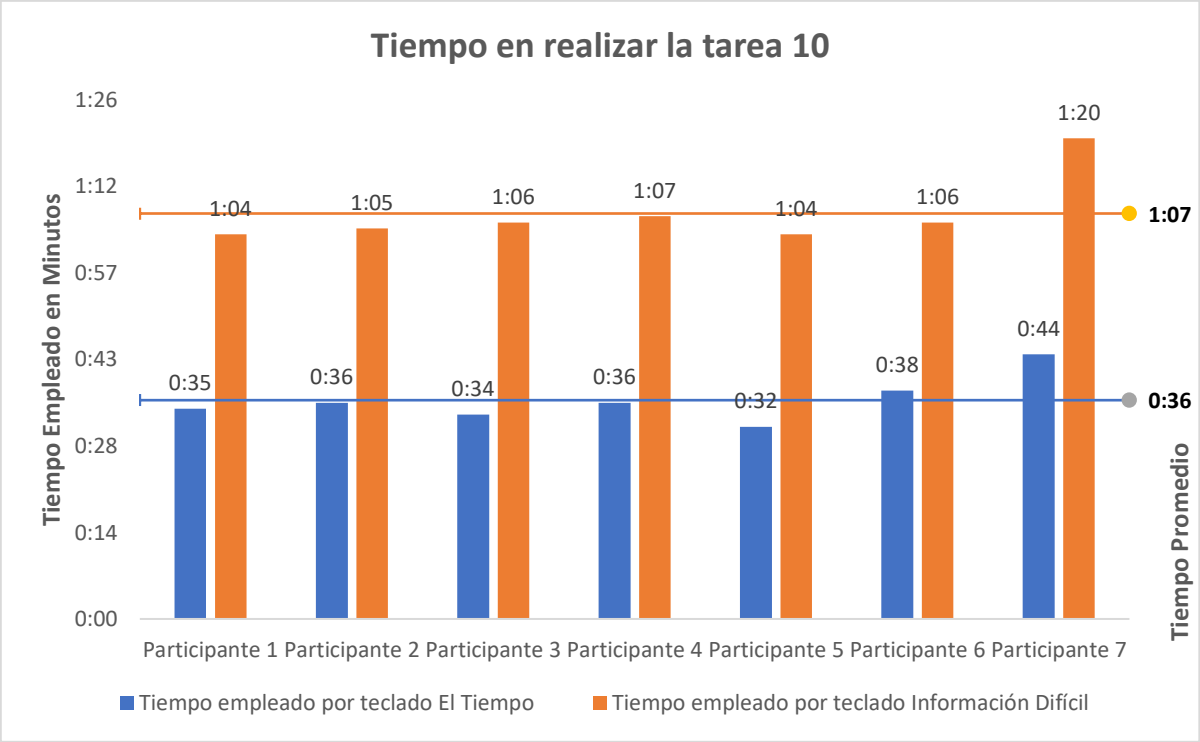


Figura 63. Tiempo empleado por cada participante en realizar la tarea por teclado.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 36 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 1 minuto y 7 segundos (Figura 63).

Finalmente, a cada participante que interactuó con los simuladores se le realizó una serie de preguntas con el fin de saber el nivel de dificultad que tuvo en cada pregunta. Se realizaron preguntas puntuales referentes a cómo le fue en cada tarea o reto planteado. A continuación se detallan los resultados obtenidos.

8.1.1. Resultados encuestas Simulador Laboral Información Difícil

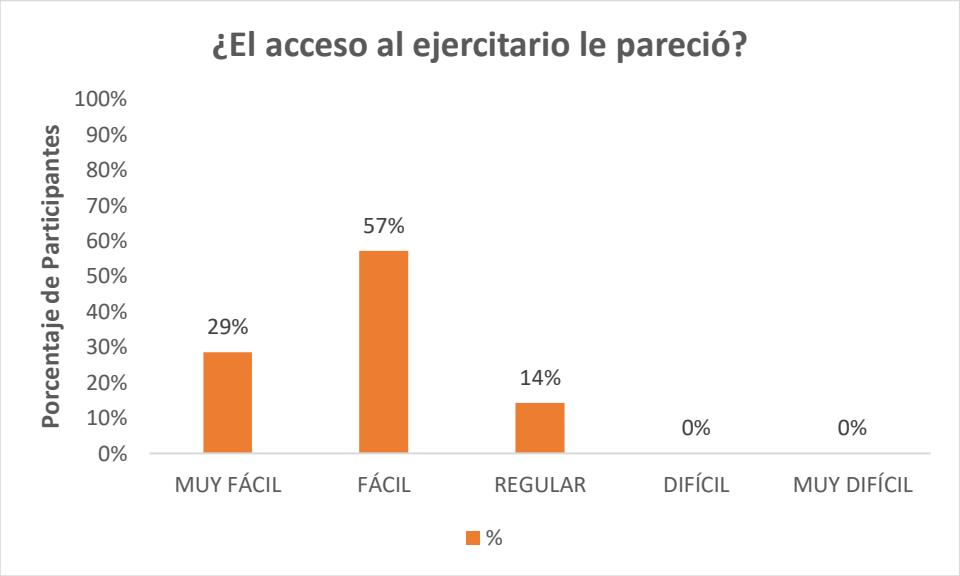


Figura 64. Pregunta 1 sobre la dificultad de acceso al ejercitario (ejercicio) del Simulador Laboral.

Interpretación.

El 29% de los participantes encuestados indican que el acceso al ejercitario(ejercicio) les pareció muy fácil, el 57% indicaron que el acceso fue de fácil acceso y el 14% indicó que el acceso fue regular (Figura 64).

Figura 65. Pregunta 2 sobre la coherencia de los iconos.

Interpretación.

El 100% de los participantes encuestados señalan que los íconos utilizados indican con claridad la acción que realiza cada objeto de la interfaz de usuario (Figura 65).

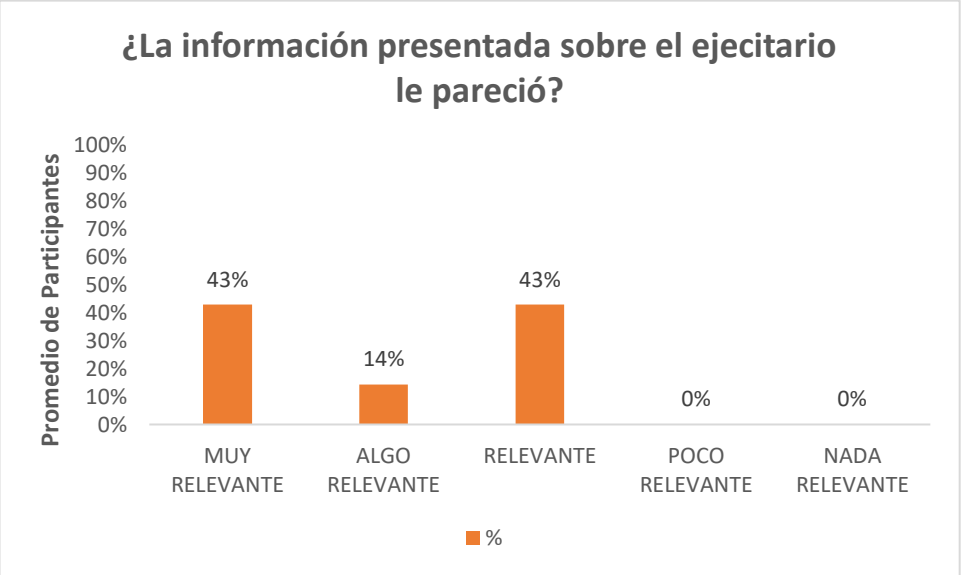


Figura 66. Pregunta 3 sobre la relevancia de la información presentada en el simulador laboral.

Interpretación.

El 43% de los participantes encuestados indican que la información presentada al usuario sobre el ejercitario es muy relevante, el 14% indica que la información es algo relevante y el 43% indica que la información presentada es relevante (Figura 66).

Figura 67. Pregunta 4 si la ambientación de escenario presentada es la apropiada.

Interpretación.

El 67% de los participantes encuestados indican que el escenario del ejercitario simulado les pareció muy adecuado y el 33% indica que el escenario del ejercitario simulado les pareció adecuado (Figura 67).

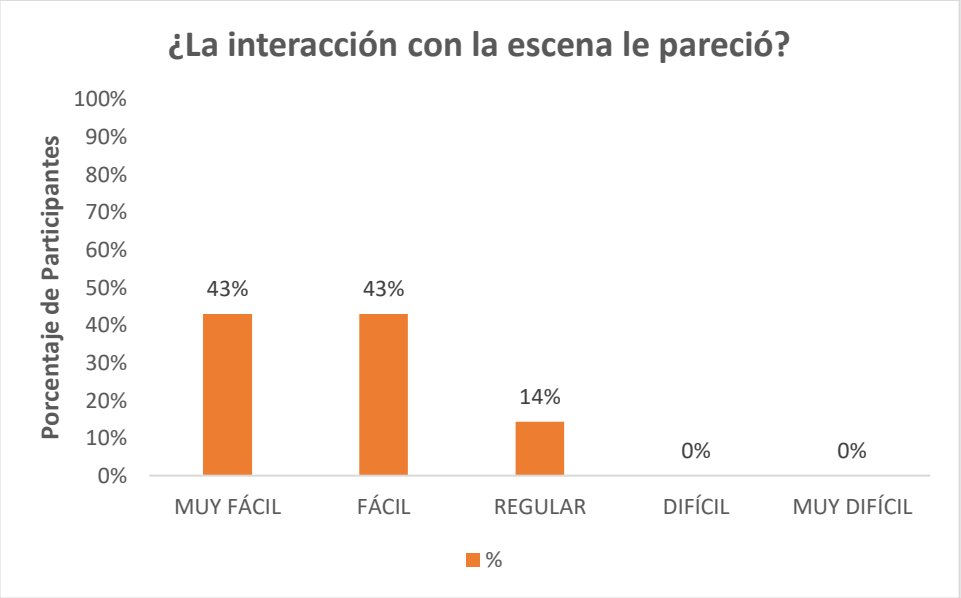


Figura 68. Pregunta 5 sobre la interacción del participante con la escena.

Interpretación.

El 43% de los participantes encuestados indican que fue muy fácil interactuar con la escena, el 43% indica que la interacción con la escena fue fácil y el 14% indica que la interacción fue regular (Figura 68).

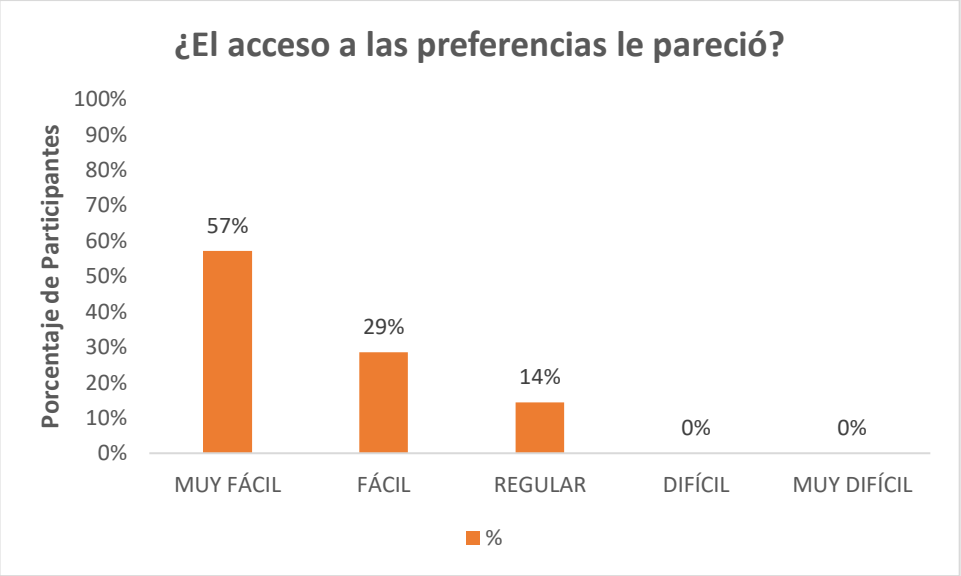


Figura 69. Pregunta 6 sobre el acceso a la barra de preferencias.

Interpretación.

El 57% de los participantes encuestados indican que fue muy fácil acceder a la barra de preferencias, el 29% indica que fue fácil y el 14% indica que el acceso a la barra de preferencias fue regular (Figura 69).

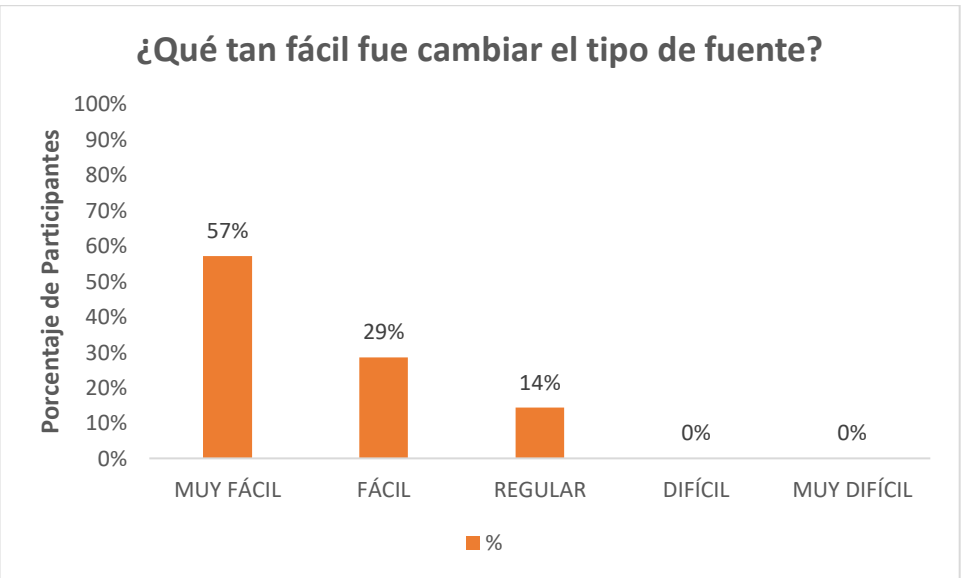


Figura 70. Pregunta 7 sobre la facilidad de cambio de fuentes de texto.

Interpretación.

El 57% de los participantes encuestados indican que fue muy fácil cambiar de fuente de texto, el 29% indica que fue fácil y el 14% indica que fue regular (Figura 70).

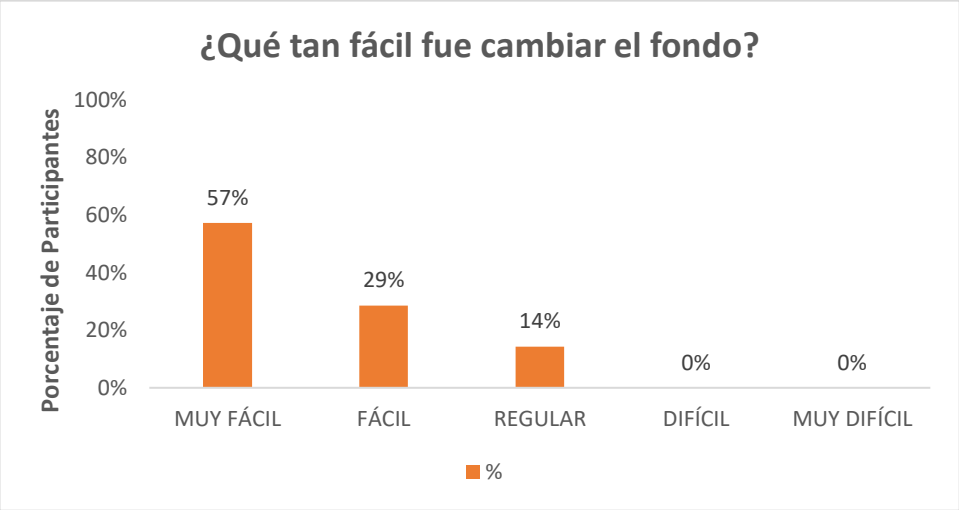


Figura 71. Pregunta 8 sobre la facilidad al cambiar el contraste de la interfaz.

Interpretación.

El 57% de los participantes encuestados indican que fue muy fácil cambiar el fondo de la interfaz de usuario, el 29% indica que fue fácil y el 14% indica que fue regular (Figura 71).

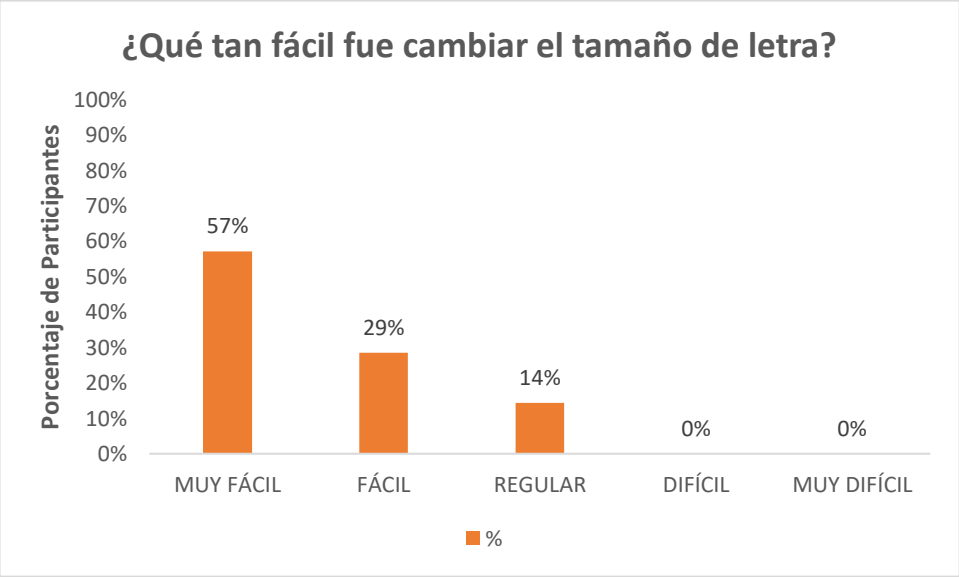


Figura 72. Pregunta 9 sobre la facilidad de cambio de tamaño de texto.

Interpretación.

El 57% de los participantes encuestados indican que fue muy fácil cambiar el tamaño del texto, el 29% indica que fue fácil y el 14% indica que fue regular (Figura 72).

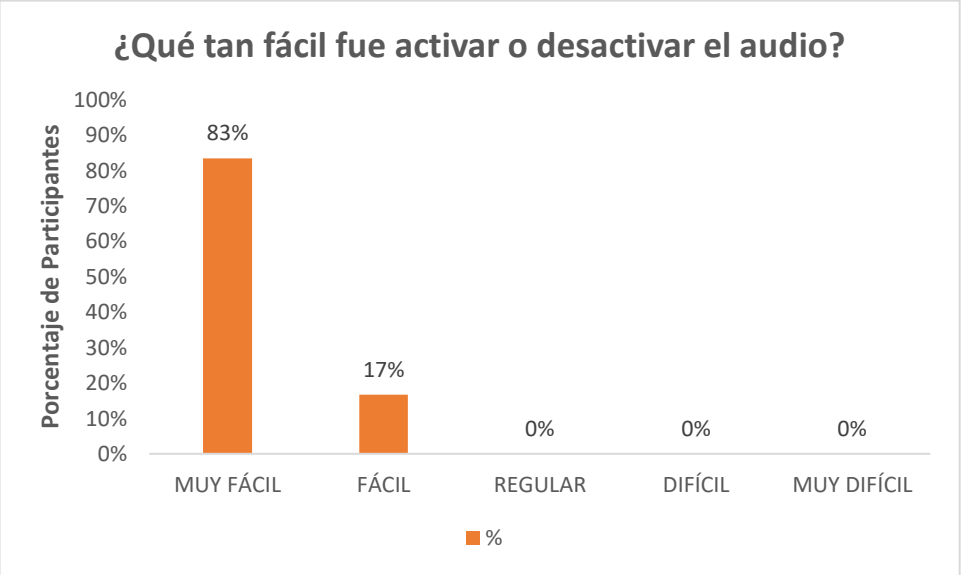


Figura 73. Pregunta 10 sobre la facilidad de activar o desactivar la descripción de audio del contenido del simulador laboral.

Interpretación.

El 83% de los participantes encuestados indican que fue muy fácil activar o desactivar el audio de la escena y el 17% indica que fue fácil (Figura 73).

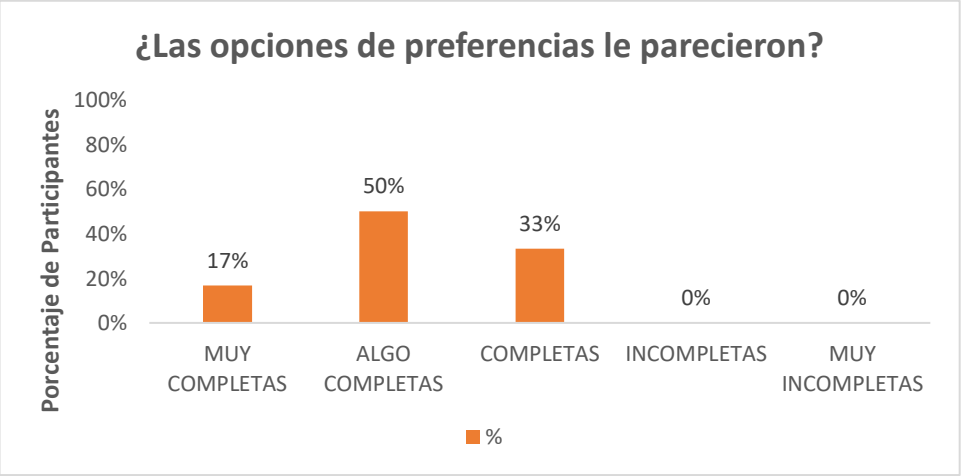


Figura 74. Pregunta 11 sobre la aceptación de las opciones de preferencias mostradas.

Interpretación.

El 17% de los participantes encuestados indican que las opciones de preferencias implementadas para el simulador laboral son muy completas, el 50% indica que son algo completas y el 33% indica que son completas (Figura 74).

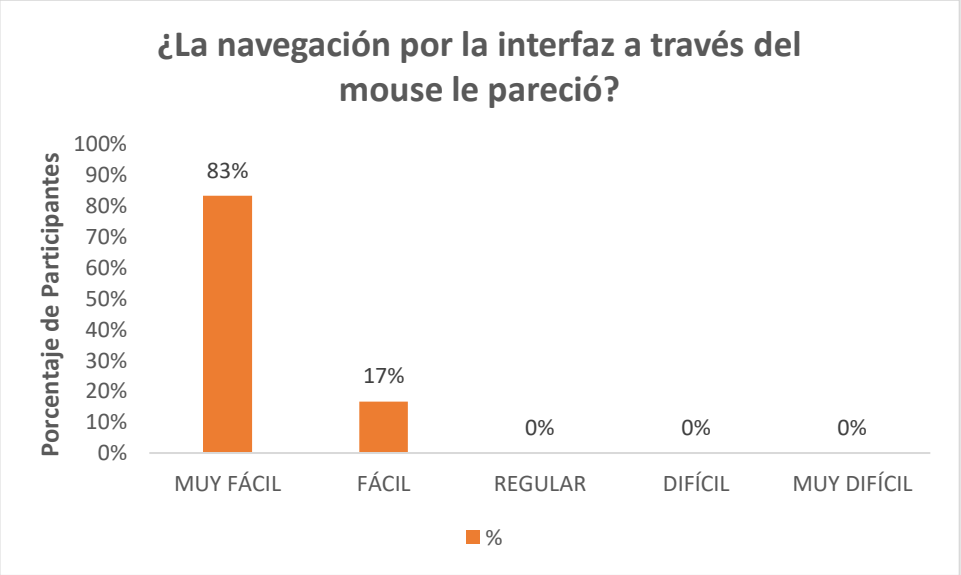


Figura 75. Pregunta 12 sobre la facilidad de navegación mediante el mouse.

Interpretación.

El 83% de los participantes encuestados indican que navegar por la interfaz a través del mouse les resultó muy fácil y el 17% indica que fue fácil (Figura 75).

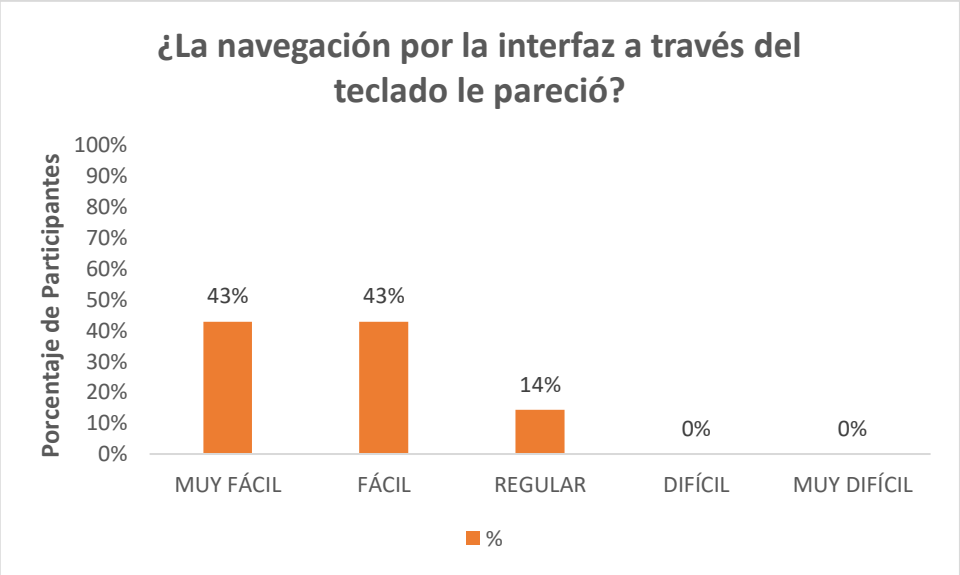


Figura 76. Pregunta 13 sobre la facilidad de navegación mediante el teclado.

Interpretación.

El 43% de los participantes encuestados indican que navegar por la interfaz a través del teclado les resultó muy fácil, el 43% indica que les resultó fácil y el 14% indica que fue regular (Figura 75).

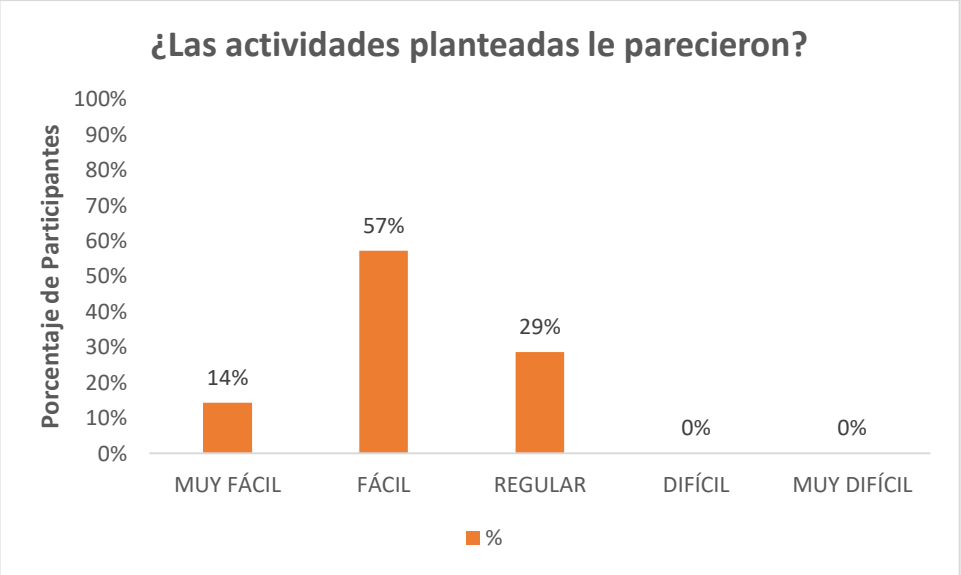


Figura 77. Pregunta 14 sobre la facilidad de interacción con actividades planteadas.

Interpretación.

El 14% de los participantes encuestados indican que las actividades planteadas fueron muy fáciles de realizar, el 57% indica que fue fácil y el 29% indica que fue regular (Figura 77).

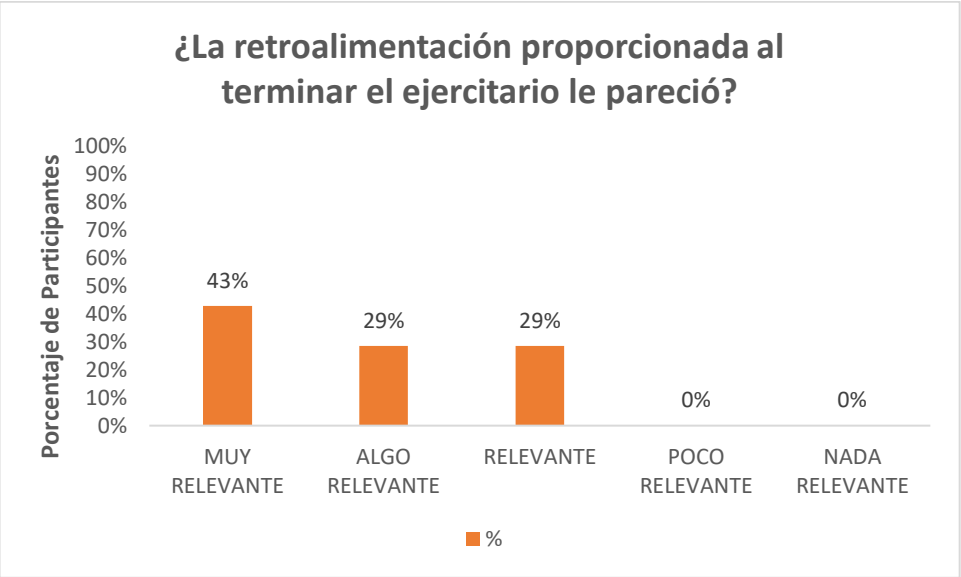


Figura 78. Pregunta 15 sobre la relevancia de retroalimentación presentada.

Interpretación.

El 43% de los participantes encuestados indican que la retroalimentación presentada al terminar la interacción con el ejercitario es muy relevante, el 29% indica que es algo relevante y el 29% indica que es relevante (Figura 78).

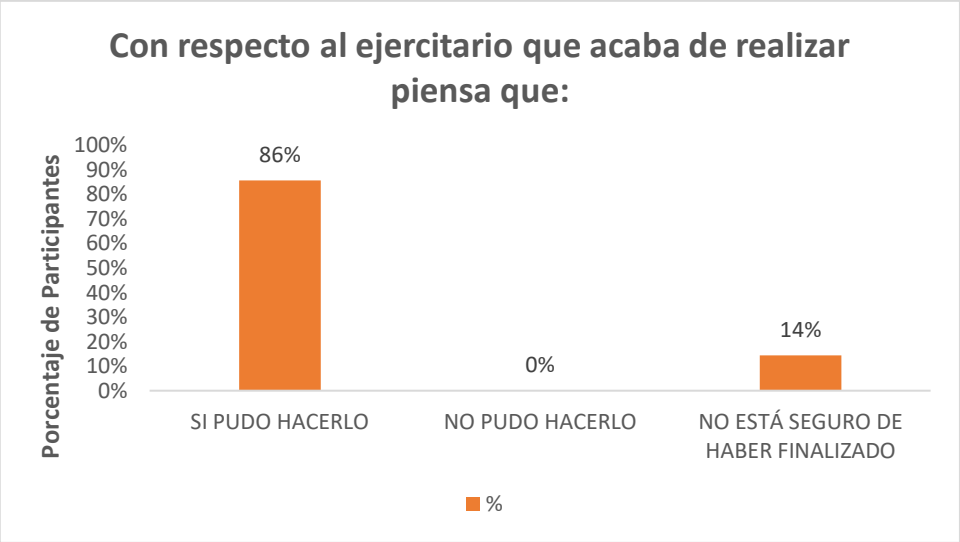


Figura 79. Pregunta 16 al participante si consiguió o no finalizar la interacción de los retos propuestos.

Interpretación.

El 86% de los participantes encuestados indican que sí pudieron realizar los retos planteados y el 14% indica que no está seguro de haber finalizado los retos planteados (Figura 79).

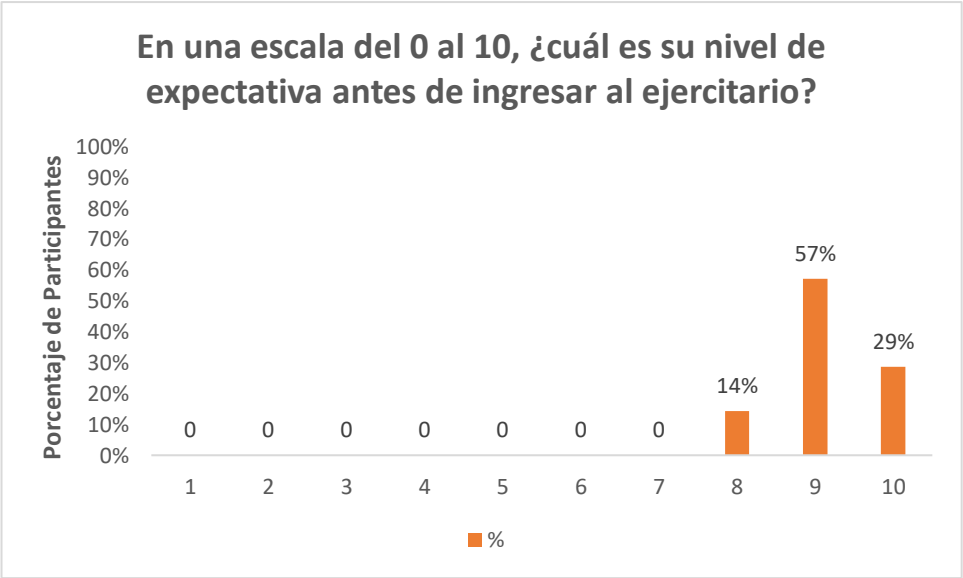


Figura 80. Pregunta 17 sobre el nivel de expectativa reflejada por el participante.

Interpretación.

El 14% de los participantes encuestados indican que en el nivel de expectativa antes de ingresar al ejercitario fue de 8 considerando un rango de 1 a 10, el 57% indica que fue de 9 y el 29% indica que fue una expectativa de 10 (Figura 80).

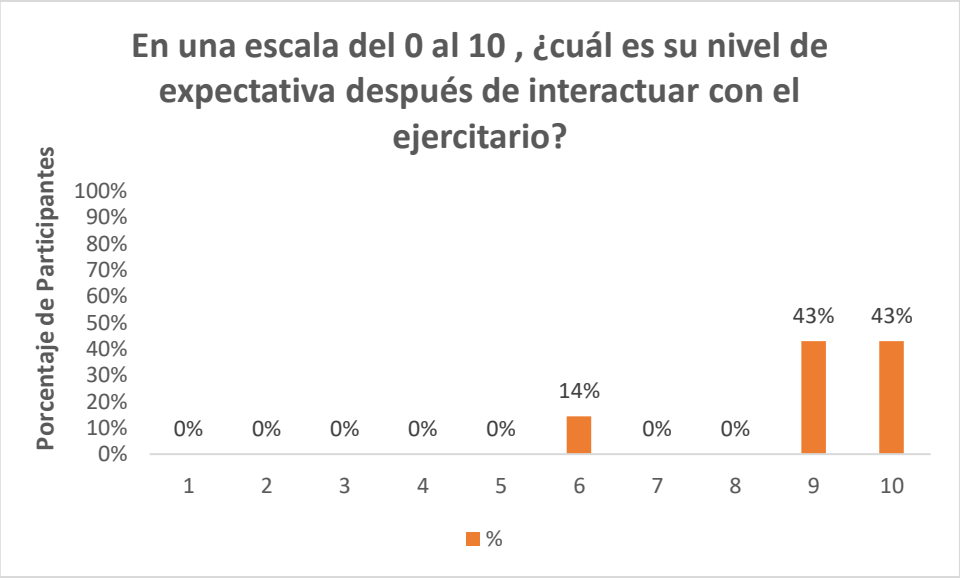


Figura 81. Pregunta 18 sobre el nivel de expectativa reflejada por el participante.

Interpretación.

El 14% de los participantes encuestados indican que en el nivel de expectativa después de interactuar con el ejercitario fue de 6 considerando un rango de 1 a 10, el 43% indica que fue de 9 y el 43% indica que fue una expectativa de 10 (Figura 81).

Argumenta las respuestas 17 y 18 sobre el simulador laboral:

La mayoría de los participantes indicaron que la introducción presentada antes de empezar a interactuar con los simuladores laborales fue relevante, ya que con ello se tuvo una idea del contexto del simulador laboral, lo cual les pareció motivante y también sobre las mecánicas del mismo. Una vez finalizada la interacción con los simuladores, los participantes indicaron que los simuladores pueden ayudar mucho al usuario sobre todo al usuario con discapacidad, ya que están simulados en base a situaciones de la vida real. Por lo que, estos usuarios podrían experimentar con estas herramientas las veces que les sea necesarias sin temor a equivocarse, permitiéndoles formarse para el campo laboral. Con la debida importancia hacia

el participante con discapacidad, se define realizar los cambios sugeridos con respecto a la navegación.

Tomando en cuenta el tiempo empleado por cada participante en realizar cada uno de los retos propuestos, en la tabla 19 se detalla el tiempo total.

Participante	**Profesión/Carrera**	**Discapacidad**	**Hora Inicio**	**Hora Fin**	**Tiempo total**
Participante 1	Estudiante de Computación	No	1:15 pm	1:30 pm	15 minutos
Participante 2	Estudiante de Ing. Sistemas	No	2:08 pm	2: 25 pm	17 minutos
Participante 3	Estudiante de Ing. Sistemas	No	3:20 pm	3:40 pm	20 minutos
Participante 4	Estudiante de Ing. Sistemas	No	11:15 am	11:32 am	17 minutos
Participante 5	Estudiante de Ing. Sistemas	No	12:00 pm	12:19 pm	19 minutos
Participante 6	Estudiante de Ing. Sistemas	No	1:10 pm	1:28 pm	18 minutos
Participante 7	Auxiliar Biblioteca UPS	Visual	11:00 am	11:25 pm	25 minutos

Tabla 19. Tiempo empleado por participantes en resolver las tareas asignadas.

8.1.2. Resultados encuestas Simulador Laboral El Tiempo

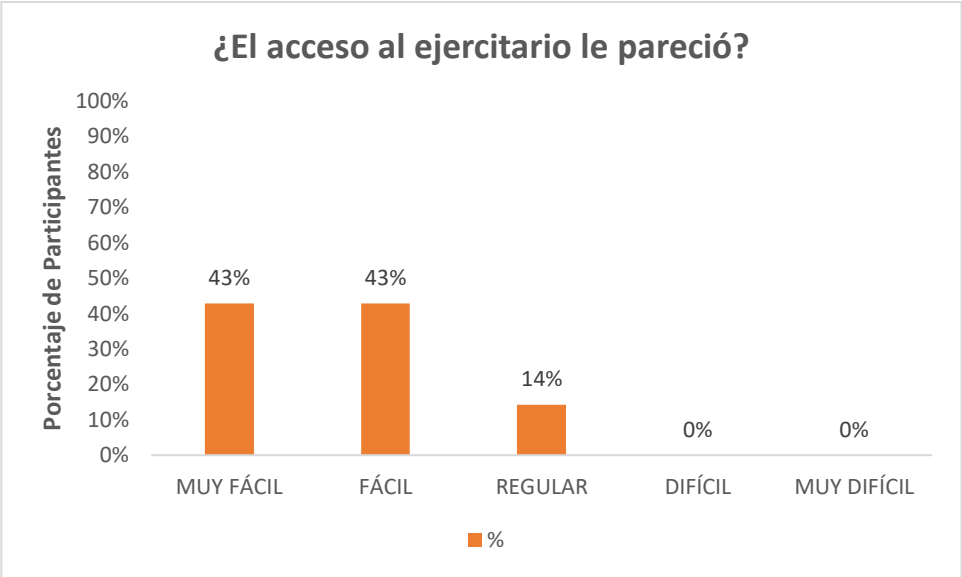


Figura 82. Pregunta 1 sobre la dificultad de acceso al ejercitario (ejercicio) del Simulador Laboral.

Interpretación.

El 29% de los participantes encuestados indican que el acceso al ejercitario(ejercicio) les pareció muy fácil, el 57% indicaron que el acceso fue de fácil acceso y el 14% indicó que el acceso fue regular (Figura 82).

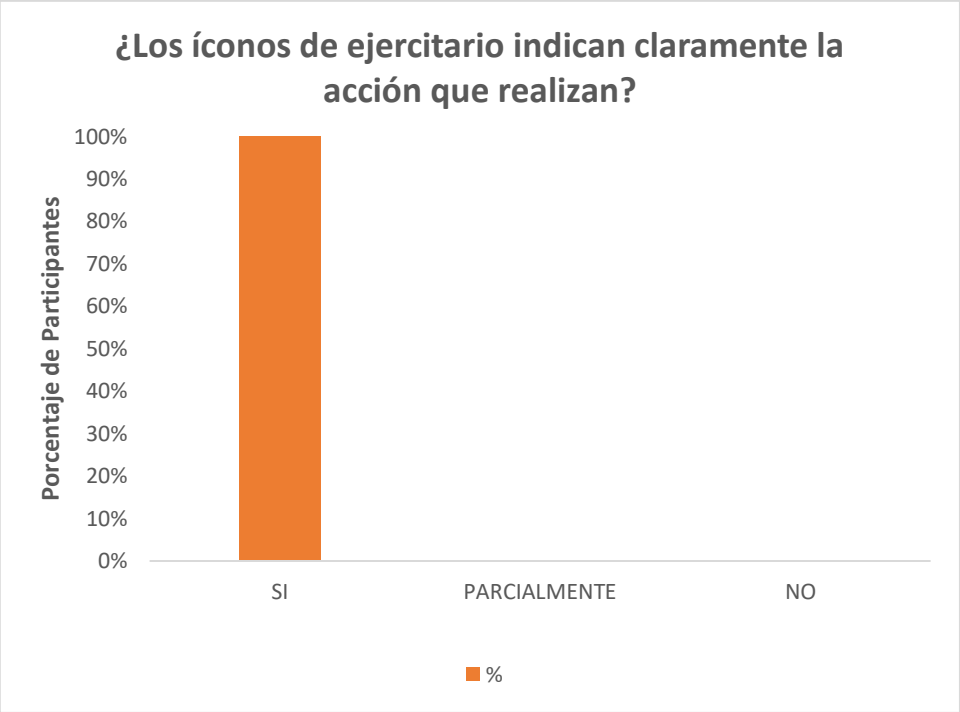


Figura 83. Pregunta 2 sobre la coherencia de los íconos.

Interpretación.

El 100% de los participantes encuestados señalan que los íconos utilizados indican con claridad la acción que realiza cada objeto de la interfaz de usuario (Figura 83).

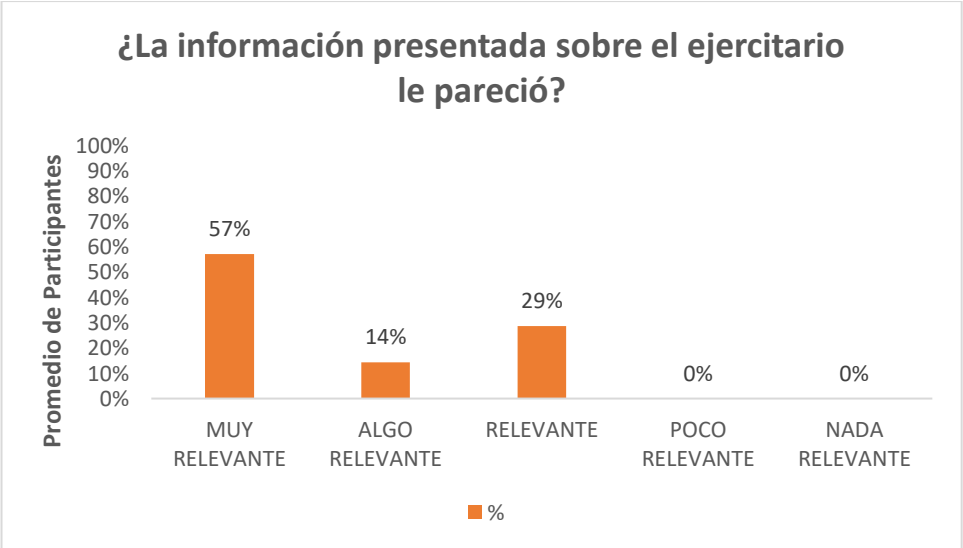


Figura 84. Pregunta 3 sobre la relevancia de la información presentada en el simulador laboral.

Interpretación.

El 57% de los participantes encuestados indican que la información presentada al usuario sobre el ejercitario es muy relevante, el 14% indica que la información es algo relevante y el 29% indica que la información presentada es relevante (Figura 84).

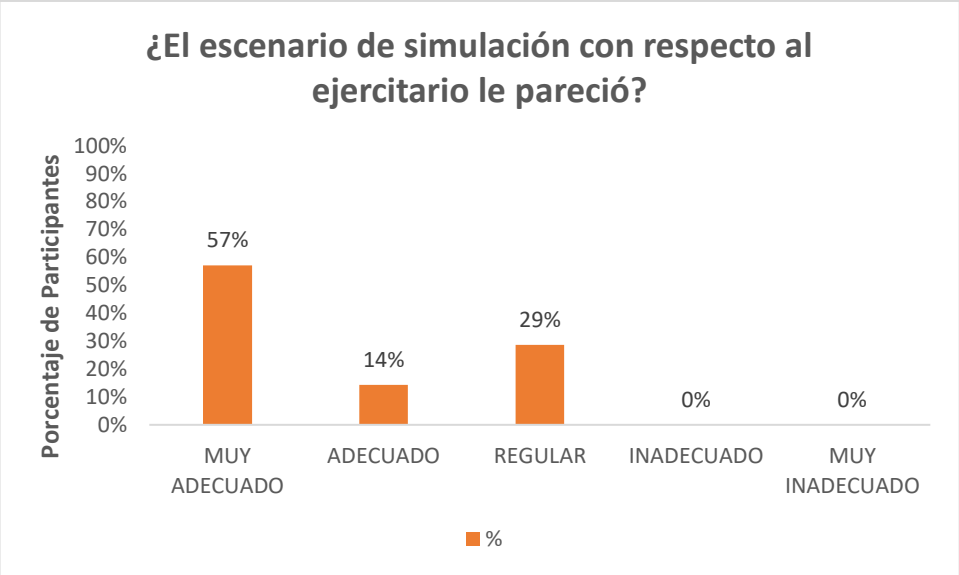


Figura 85. Pregunta 4 si la ambientación de escenario presentada es la apropiada.

Interpretación.

El 57% de los participantes encuestados indican que el escenario del ejercitario simulado les pareció muy adecuado, el 14% indica que el escenario del ejercitario simulado les pareció adecuado y el 29% indica que fue regular (Figura 85).

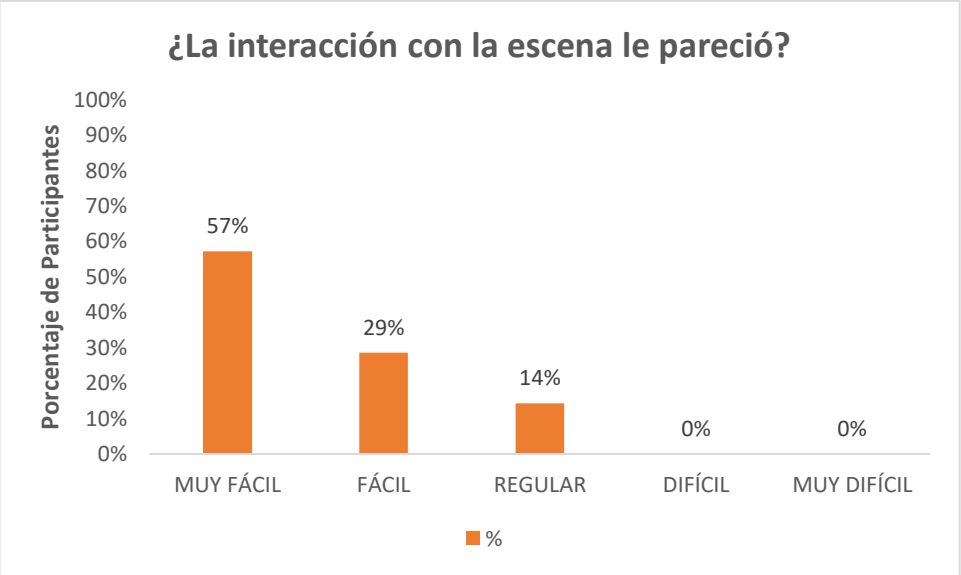


Figura 86. Pregunta 5 sobre la facilidad de interacción con la escena.

Interpretación.

El 57% de los participantes encuestados indican que fue muy fácil interactuar con la escena, el 29% indica que la interacción con la escena fue fácil y el 14% indica que la interacción fue regular (Figura 86).

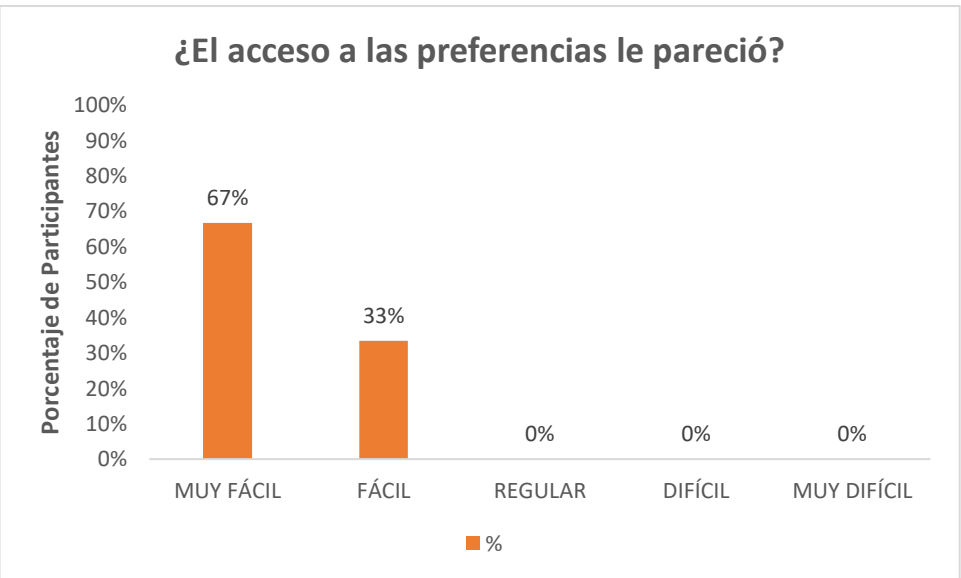


Figura 87. Pregunta 6 sobre la facilidad de acceso a la barra de preferencias del simulador laboral.

Interpretación.

El 67% de los participantes encuestados indican que fue muy fácil acceder a la barra de preferencias y el 33% indica que fue de fácil acceso (Figura 87).

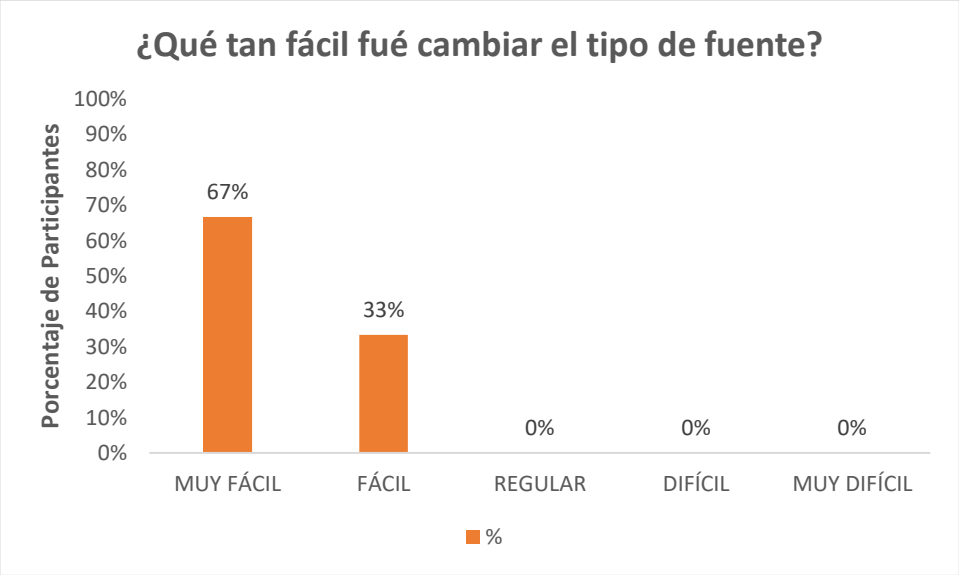


Figura 88. Pregunta 7 sobre la facilidad al cambiar de tipo de fuente.

Interpretación.

El 67% de los participantes encuestados indican que fue muy fácil cambiar de fuente de texto y el 33% indica que fue fácil (Figura 88).

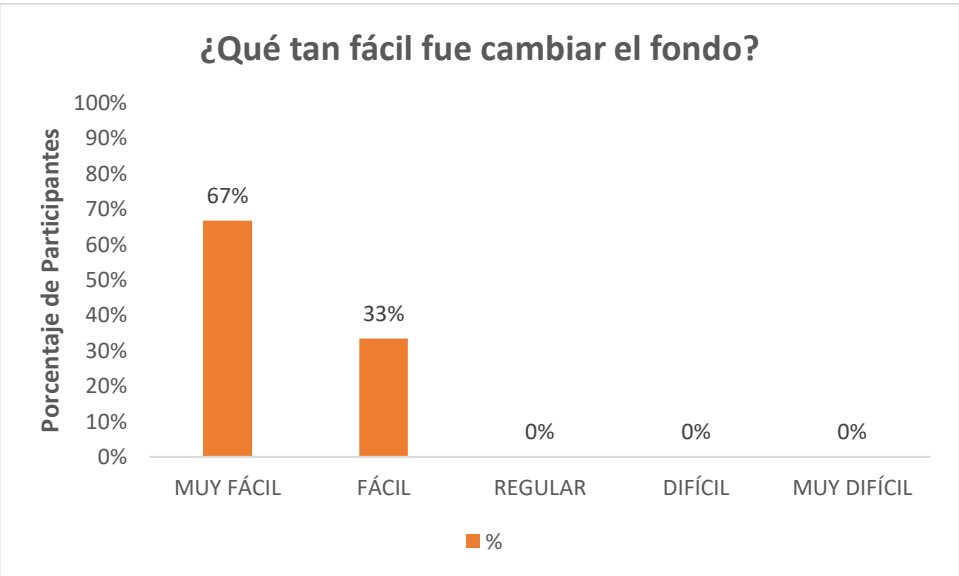


Figura 89. Pregunta 8 sobre la facilidad al cambiar el contraste de la interfaz.

Interpretación.

El 67% de los participantes encuestados indican que fue muy fácil cambiar el fondo de la interfaz de usuario y el 33% indica que fue fácil (Figura 89).

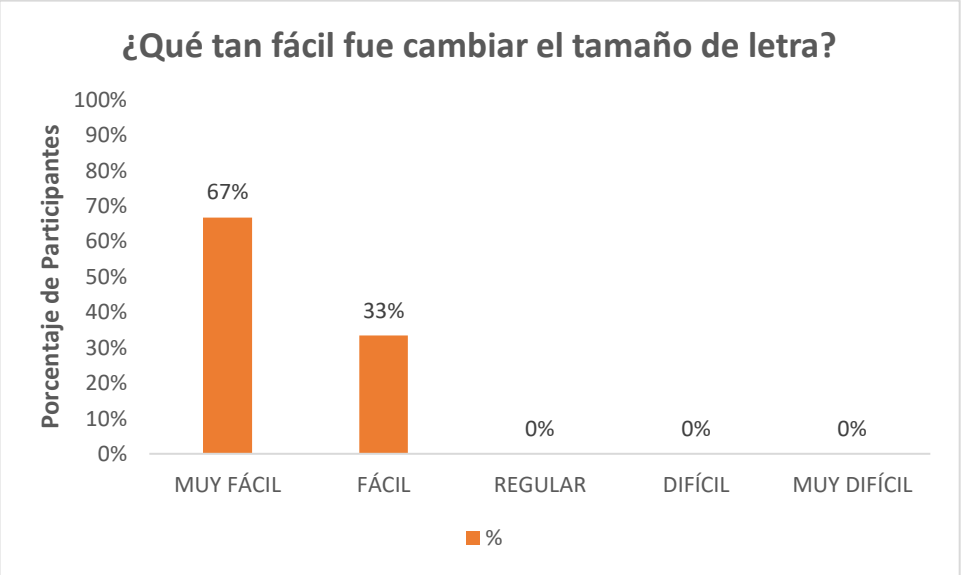


Figura 90. Pregunta 9 sobre la facilidad al cambiar el tamaño del texto.

Interpretación.

El 67% de los participantes encuestados indican que fue muy fácil cambiar el tamaño del texto y el 33% indica que fue fácil (Figura 90).

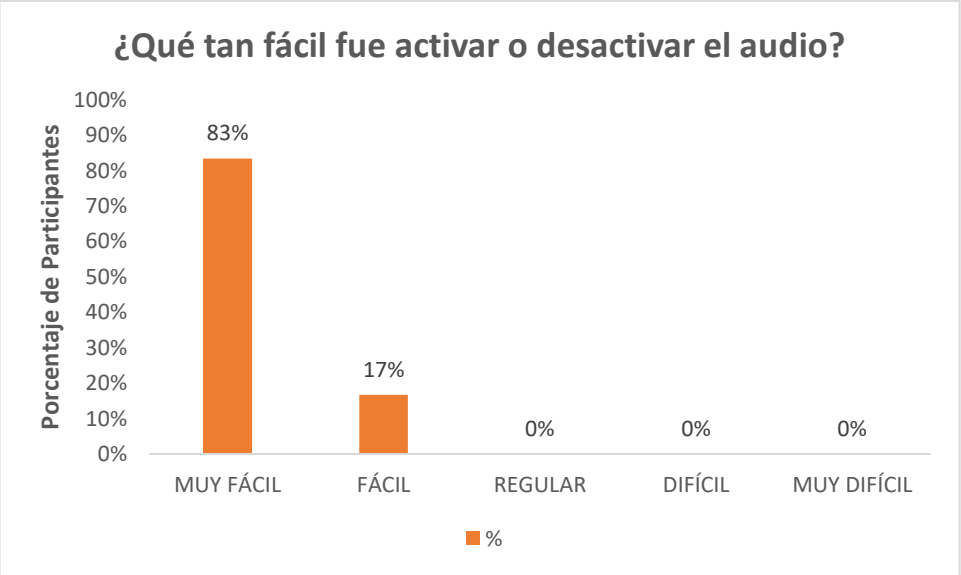


Figura 91. Pregunta 10 sobre la facilidad de activar o desactivar la descripción de audio del contenido del simulador laboral.

Interpretación.

El 83% de los participantes encuestados indican que fue muy fácil activar o desactivar el audio de la escena y el 17% indica que fue fácil (Figura 91).

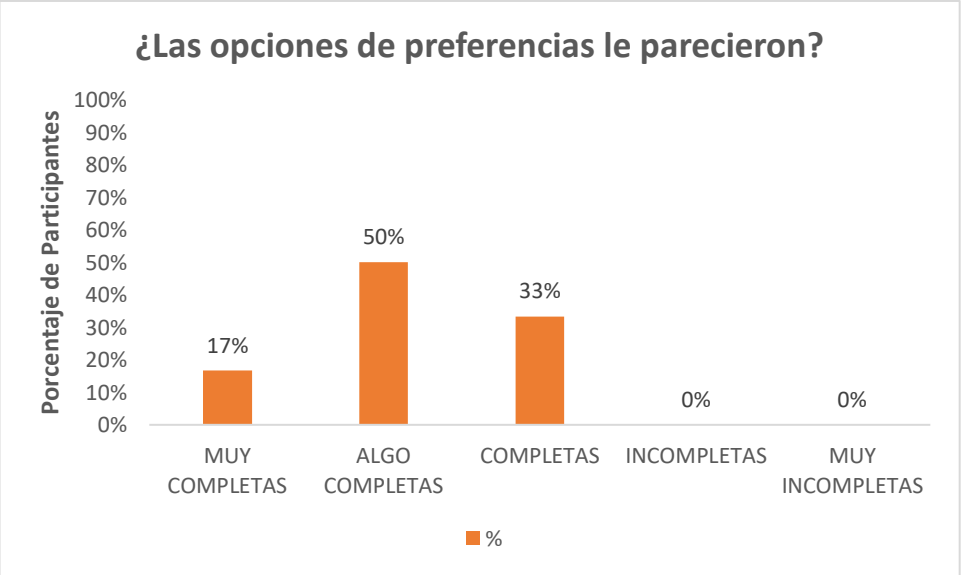


Figura 92. Pregunta 11 sobre la aceptación de las opciones de preferencias mostradas.

Interpretación.

El 17% de los participantes encuestados indican que las opciones de preferencias implementadas para el simulador laboral son muy completas, el 50% indica que son algo completas y el 33% indica que son completas (Figura 92).

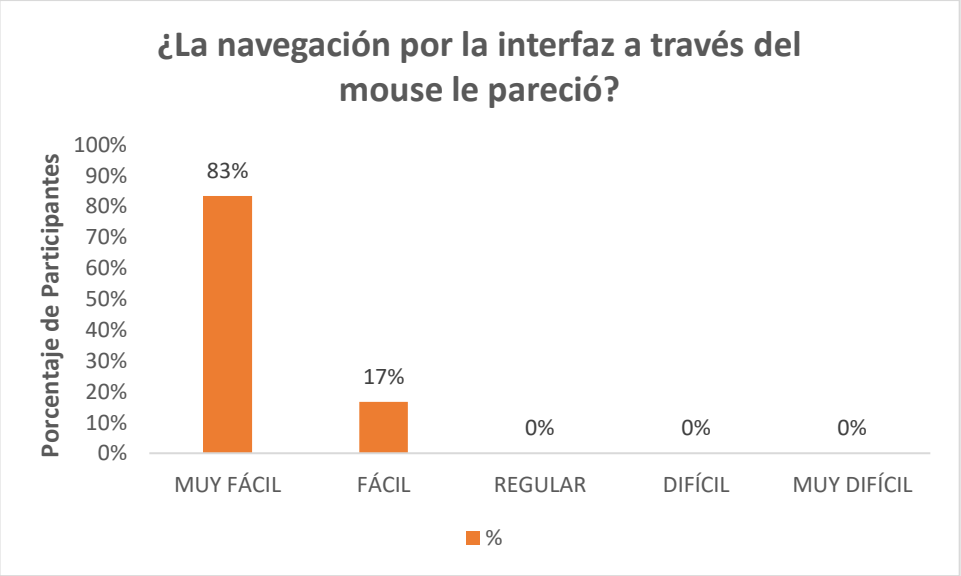


Figura 93. Pregunta 12 sobre la facilidad de navegación mediante el mouse.

Interpretación.

El 83% de los participantes encuestados indican que navegar por la interfaz a través del mouse les resultó muy fácil y el 17% indica que fue fácil (Figura 93).

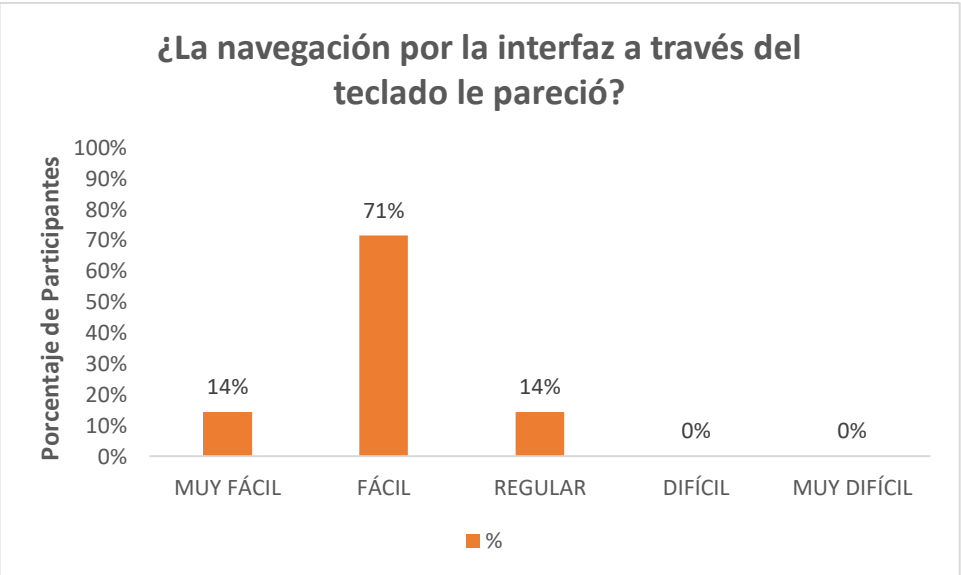


Figura 94. Pregunta 13 sobre la facilidad de navegación mediante el teclado.

Interpretación.

El 14% de los participantes encuestados indican que navegar por la interfaz a través del teclado les resultó muy fácil, el 71% indica que les resultó fácil y el 14% indica que fue regular (Figura 94).

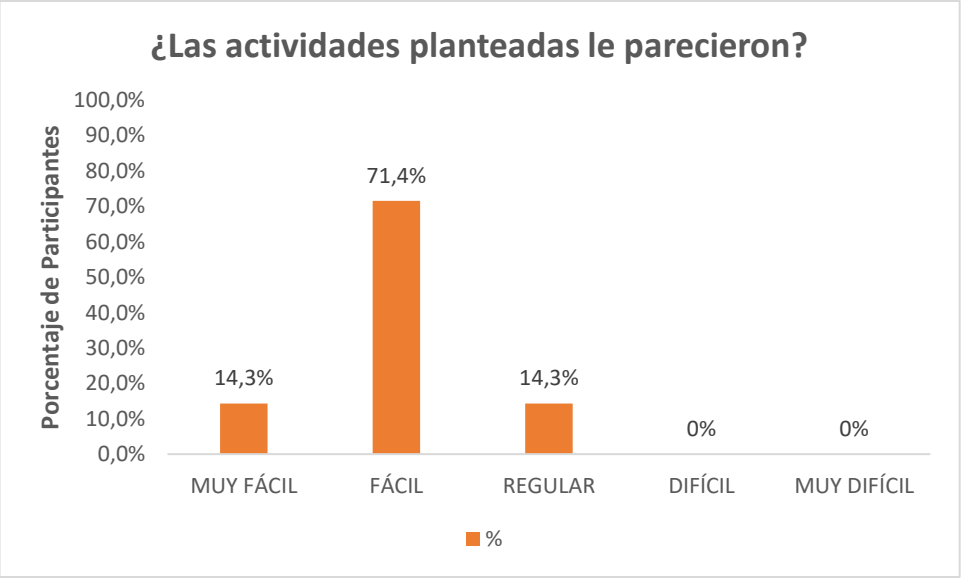


Figura 95. Pregunta 14 sobre la facilidad de interacción con actividades planteadas.

Interpretación.

El 14% de los participantes encuestados indican que las actividades planteadas fueron muy fáciles de realizar, el 71% indica que fue fácil y el 14% indica que fue regular (Figura 95).

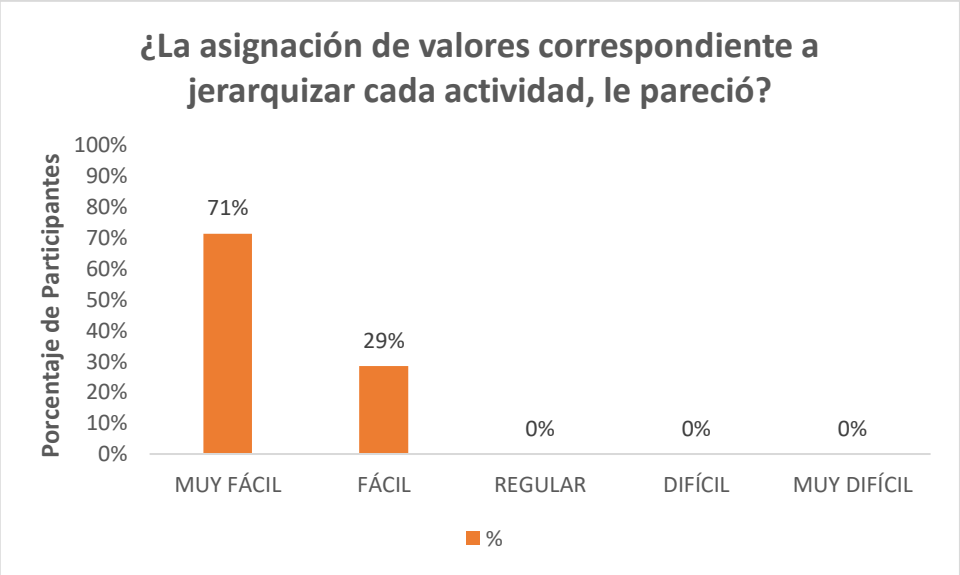


Figura 96. Pregunta 15 sobre la facilidad de ingreso de valores correspondientes a la jerarquización de cada actividad.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 1 minuto y 2 segundos. En cambio, en el Simulador Laboral Información Difícil, los participantes emplearon un tiempo promedio de 33 segundos (Figura 62).

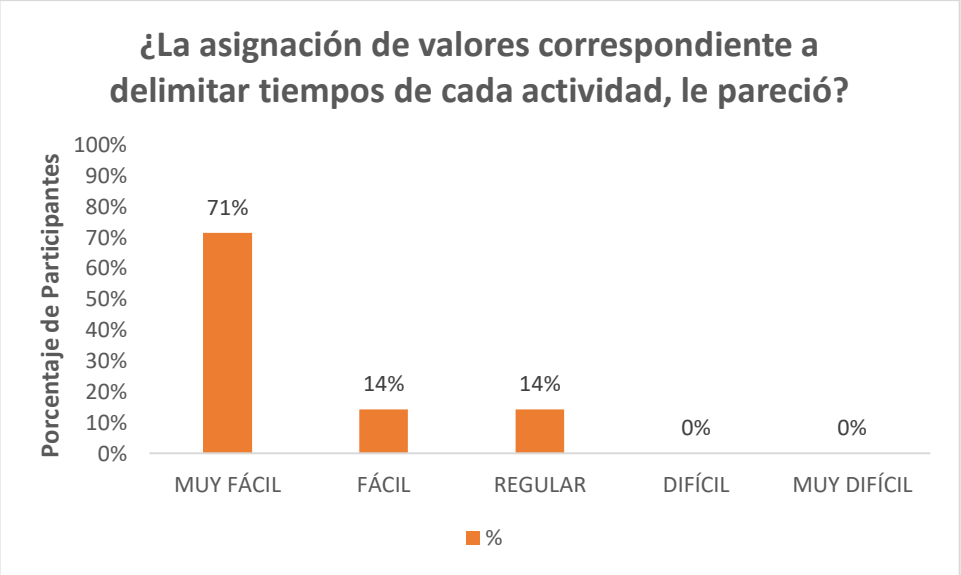


Figura 97. Pregunta 16 sobre la facilidad de ingreso de valores correspondientes a delimitar tiempos de cada actividad.

Interpretación.

En el reto planteado del Simulador Laboral El Tiempo, los participantes emplearon un tiempo promedio de 1 minuto y 2 segundos. En cambio, en el Simulador Laboral Información Difícil los participantes emplearon un tiempo promedio de 33 segundos (Figura 62).

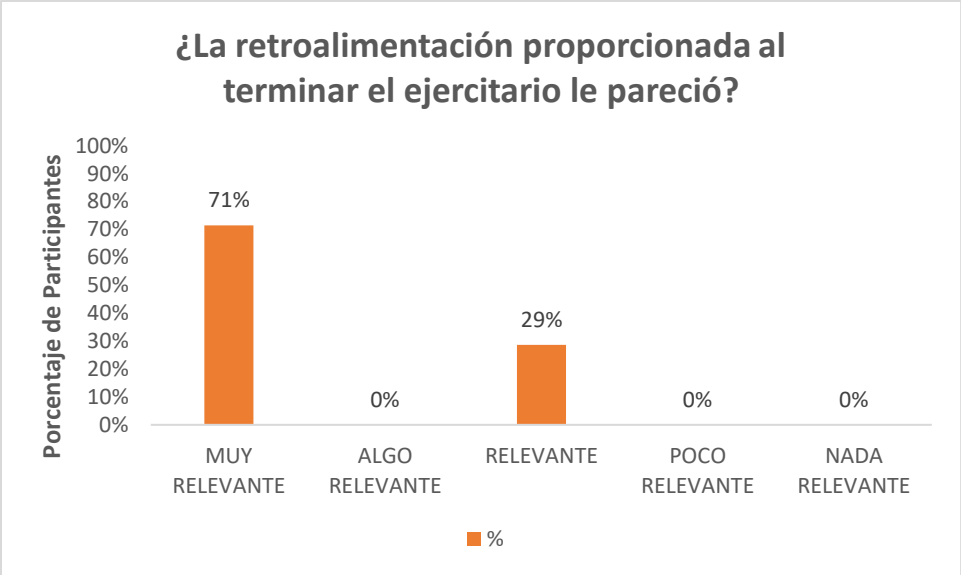


Figura 98. Pregunta 17 sobre la relevancia de retroalimentación presentada.

Interpretación.

El 71% de los participantes encuestados indican que la retroalimentación presentada al terminar la interacción con el ejercitario es muy relevante y el 29% indica que es relevante (Figura 98).

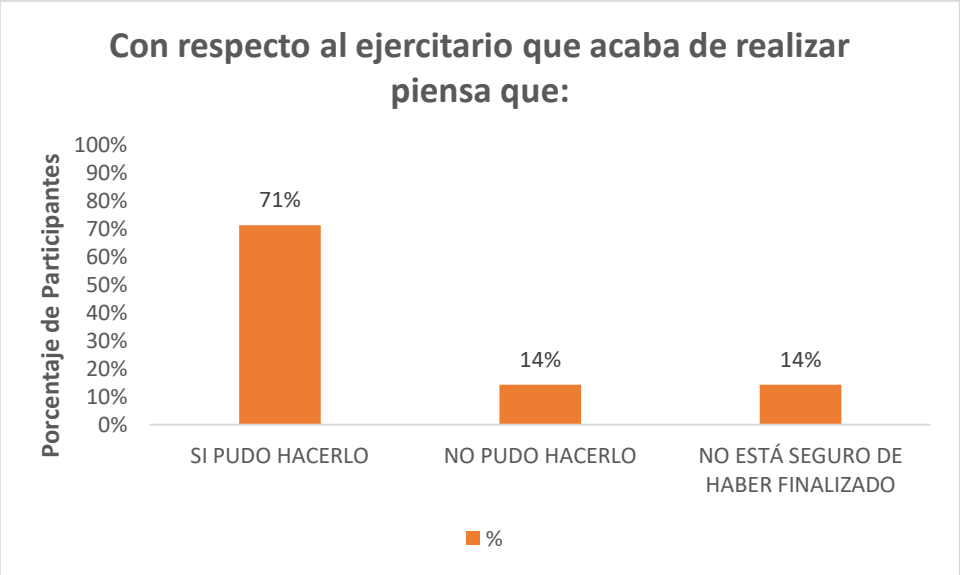


Figura 99. Pregunta 18 al participante si consiguió o no finalizar la interacción de los retos propuestos.

Interpretación.

El 71% de los participantes encuestados indican que sí pudieron realizar los retos planteados, el 14% indica que no pudo hacerlo y el 14% indica que no está seguro de haber finalizado los retos planteados (Figura 99).

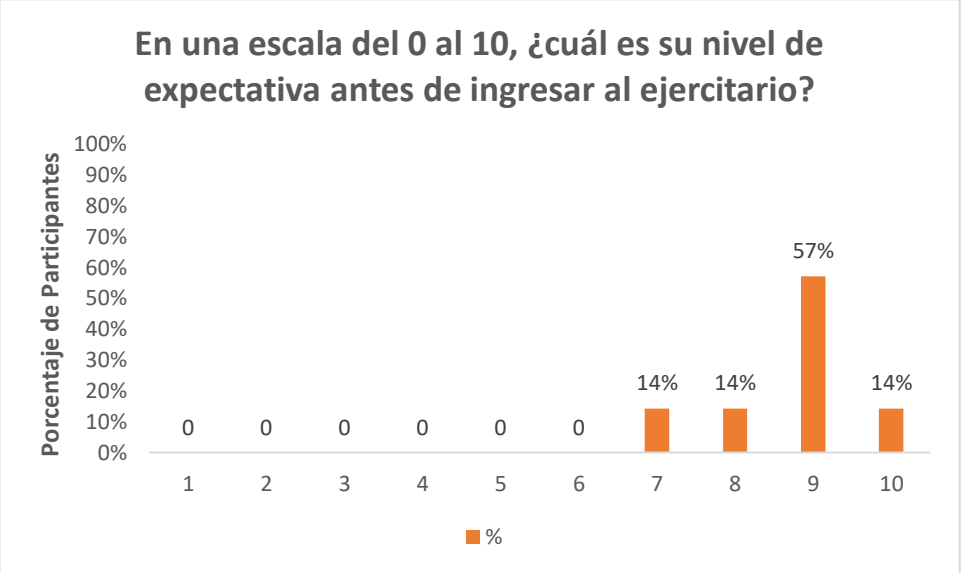


Figura 100. Pregunta 19 sobre el nivel de expectativa reflejada por el participante.

Interpretación.

El 14% de los participantes encuestados indican que en el nivel de expectativa antes de ingresar al ejercitario fue de 7 considerando un rango de 1 a 10, el 14% indica que fue de 8, el 57% indica que fue de 9 y el 14% indica que fue una expectativa de 10 (Figura 100).

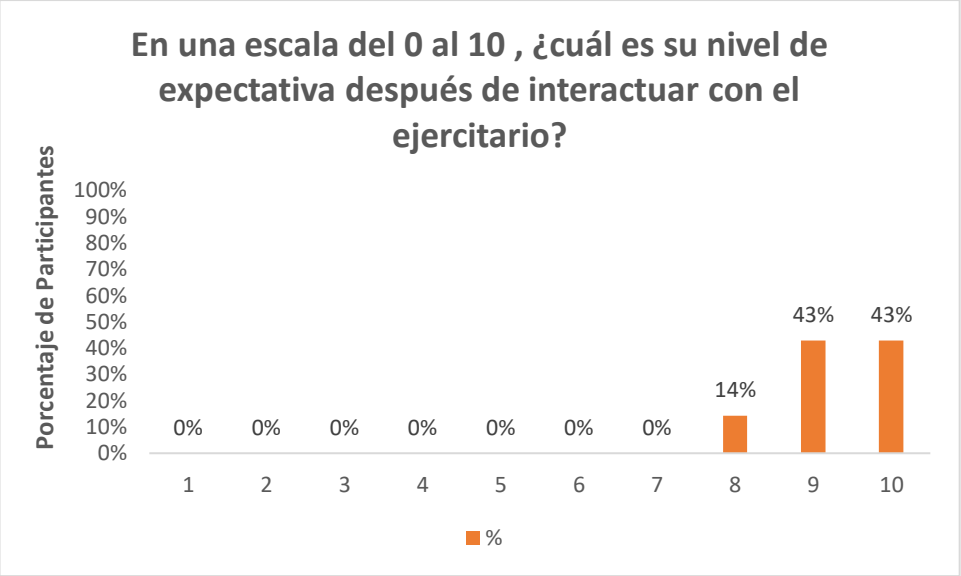


Figura 101. Pregunta 20 sobre el nivel de expectativa reflejada por el participante.

Interpretación.

El 14% de los participantes encuestados indican que en el nivel de expectativa después de interactuar con el ejercitario fue de 8 considerando un rango de 1 a 10, el 43% indica que fue de 9 y el 43% indica que fue una expectativa de 10 (Figura 101).

Argumenta las respuestas 19 y 20 sobre el simulador laboral:

Como en el caso del simulador laboral Información Difícil, la mayoría de los participantes definieron que la introducción presentada antes de empezar a interactuar con los simuladores laborales fue relevante, ya que con ello se tuvo una idea del contexto del simulador laboral, lo cual les pareció motivante y también sobre las mecánicas del mismo. Una vez finalizada la interacción, los participantes indicaron que los simuladores pueden ayudar mucho al usuario sobre todo al usuario con discapacidad, ya que los simuladores se enfocan en situaciones de la vida real. Por lo que, estos usuarios podrían experimentar con estas herramientas las veces que les sea necesarias sin temor a equivocarse, permitiéndoles formarse para el campo laboral. Con la debida importancia hacia el participante con discapacidad, se define realizar los cambios sugeridos con respecto a la navegación.

Tomando en cuenta el tiempo empleado por cada participante en realizar cada uno de los retos planteados del simulador laboral El Tiempo, en la tabla 20 se detalla el tiempo total utilizado.

Participante	Profesión/Carrera	Discapacidad	Hora Inicio	Hora Fin	Tiempo total
Participante 1	Estudiante de Computación	No	1:33 pm	1:58 pm	25 minutos
Participante 2	Estudiante de Ing. Sistemas	No	2:26 pm	2:51 pm	25 minutos
Participante 3	Estudiante de Ing. Sistemas	No	3:42 pm	4:05 pm	23 minutos
Participante 4	Estudiante de Ing. Sistemas	No	11:35 am	12:00 pm	25 minutos
Participante 5	Estudiante de Ing. Sistemas	No	12:22 pm	12:45 pm	23 minutos
Participante 6	Estudiante de Ing. Sistemas	No	1:30 pm	1:50 pm	20 minutos
Participante 7	Auxiliar Biblioteca UPS	Visual	11:30 am	12:00 pm	30 minutos

Tabla 20. Tiempo empleado por participantes en resolver las tareas asignadas.

8.2. Validación de simuladores laborales

Para la validación de estas herramientas partimos de los resultados de las encuestas realizadas, opiniones de los participantes y sus puntos de vista referente a cada simulador laboral.

A continuación, se detalla los resultados de las preguntas de cierre realizadas a todos los participantes.

a) ¿Considera que los retos planteados fueron factibles de realizar?

Los participantes indicaron que cada uno de los retos planteados fueron factibles, ya que la interfaz de ambos simuladores, mecánicas y contexto implementados eran bastante intuitivos, lo cual les permitió finalizar con éxito la interacción con cada simulador laboral.

b) ¿Considera que el tiempo empleado para interactuar con los ejercitarios (ejercicios) fue el adecuado?

Los participantes indicaron que el tiempo empleado fue el adecuado, ya que la interfaz de ambos simuladores fue bastante intuitiva. Por lo cual, no fue necesario emplear mucho tiempo y la información presentada fue relevante según la situación real presentada.

c) ¿Cuál fue la parte que le gustó de cada simulador laboral?

Los participantes indicaron que de los dos simuladores laborales lo que les gustó fue la implementación de la barra de preferencias, ya que eso les permitía adecuar la interfaz acorde a las necesidades del usuario, lo que permitió una mejor comprensión e interacción.

d) ¿Cuál fue la parte que le gusto menos de cada simulador laboral?

Los participantes coincidieron que para alcanzar una mejor navegación sería factible implementar más mecánicas (implementar más teclas para la navegación) de las ya implementadas, por lo que se enfatizó en dichas opiniones para una mejora del proyecto.

IX. Cronograma

En la tabla 21 se presenta el cronograma de actividades ejecutado para el desarrollo del proyecto.

Nombre de la Tarea	Fecha Inicio	Fecha Fin	Horas
Proyecto	**25/10/2021**	**30/10/2022**	**850**
OE.1	**25/10/2021**	**12/11/2021**	**30**
Act.1 Estudiar los diferentes conceptos de accesibilidad.	25/10/2021	26/10/2021	5
Act.2 Estudio de las normas de accesibilidad.	27/10/2021	29/10/2021	5
Act.3 Realizar un estado de arte sobre accesibilidad en interfaces en de juegos o simulación.	04/11/2021	09/11/2021	10
Act.4 Determinar conceptos aplicables de accesibilidad en interfaz de usuario de Unity 3D.	10/11/2021	12/11/2021	10
OE.2	**15/11/2021**	**25/04/2022**	**400**
Act.1 Análisis del ejercitario propuesto por psicólogos de la UDA.	15/11/2021	25/11/2021	50
Act.2 Planteamiento de escena 3D e interfaz gráfica.	26/11/2021	15/12/2021	50
Act.3 Desarrollo de escena 3D e interfaz considerando accesibilidad.	16/12/2021	21/03/2022	200
Act.4 Desarrollo de actividades y control de actividades en simulación.	22/03/2022	25/04/2022	100
OE.3	**26/05/2022**	**24/06/2022**	**170**

Act.1 Análisis de arquitectura de proyecto base de simuladores laborares.	26/05/2022	02/06/2022	50
Act.2 Integración y despliegue de modulo.	03/06/2022	15/06/2022	50
Act.3 Pruebas de verificación y funcionamiento.	16/06/2022	19/06/2022	40
Act.4 Desarrollo de los manuales técnico y de usuario.	20/06/2022	24/06/2022	30
OE.4	**25/06/2022**	**30/10/2022**	**250**
Act.1 Diseñar un plan de experimentación que permita determinar el adecuado funcionamiento y utilidad de los simuladores desarrollados hasta el momento.	25/06/2022	28/06/2022	30
Act.2 Ejecutar el plan de experimentación en una población de personas con discapacidad y expertos en discapacidad.	29/06/2022	18/08/2022	110
Act.3 Análisis de resultados obtenidos.	19/08/2022	25/08/2022	40
Act.3 Revisión con el Tutor.	26/08/2022	30/08/2022	20
Modificaciones de informe y Formato	31/08/2022	30/10/2022	50

Tabla 21. Cronograma de actividades del desarrollo del Proyecto.

X. Presupuesto

Denominación	Cantidad	Costo Unitario	Costo Total
	Unidades	Dólares	Dólares
1. Bienes			
Papel Bond A-4	1	4,50	4,50
Impresiones	250	0.05	12.50
Empastados	1	15.00	15.00
2. Tecnológico			
Computador Portátil	1	900.00	900.00
3. Servicios			
Servicio de Internet	4	33,50	134
Servicio de transporte	90	2.00	180
Material Bibliográfico	3	11.00	33.00

Alimentación	120	2.50	300
4. Personal			
Estudiante investigador	1	900.00	900.00
Asesoría	1	800.00	800.00
5. Imprevistos			
Imprevistos	1	100,00	100,00
Total	573	2.663,3	3.279

Tabla 22. Presupuesto invertido en el desarrollo del Proyecto.

XI. Conclusiones

- El objetivo del proyecto fue implementar simuladores laborales 3D accesibles, en el que se tuvo como grupo objetivo a estudiantes universitarios con discapacidad (visual, auditiva, intelectual), los cuales experimentaron con los simuladores. Al finalizar se aplicó una encuesta a cada participante con el fin de verificar el nivel de aceptación de los simuladores, los resultados arrojados de las encuestas ayudaron mucho para alcanzar el objetivo del proyecto.
- De las investigaciones realizadas acerca de los simuladores laborales en el campo educativo y de los beneficios que nos brinda Unity se logra implementar un módulo de accesibilidad orientado a mejorar la experiencia de los usuarios, basándose en las pautas de accesibilidad del contenido web, el módulo podrá integrarse a los futuros simuladores laborales propuestos por el proyecto Edutech.
- Se han desarrollado dos simuladores laborales 3D accesibles, los cuales brindan una forma interactiva y accesible de aprendizaje a estudiantes universitarios con discapacidad.
- Los simuladores laborales desarrollados fueron alojados en el servidor web perteneciente al proyecto Edutech, al cual los colaboradores tendrán acceso.

- Con la intensión de instruir al estudiante con el uso de los simuladores laborales, se ha desarrollado un manual de usuario. Para una mejor compresión del funcionamiento, configuraciones o actualizaciones que se desee hacer, se ha desarrollo un manual técnico, en el cual se detalla la lógica de los simuladores y recursos utilizados.

XII. Recomendaciones

Se recomienda a las personas que desarrollen videojuegos o simuladores 3D según sea el caso, contar con un computador de gama media-alta que cumpla con los requisitos recomendados de Unity para los desarrolladores, ya que al desarrollar un proyecto 3D con gráficos de alta definición, eventos y más consumirá muchos más recursos que un proyecto en 2D.

Finalmente, en razón que los simuladores laborales estuvieron orientados a estudiantes universitarios con discapacidad asociados a universidades socias del proyecto Edutech, se considera que debería haber más apoyo por parte del proyecto Edutech, ya que son herramientas educativas que ayudarán a fortalecer competencias en este grupo de estudiantes.

XIII. Glosario de Términos

ERASMUS: Erasmus+ es el programa de la Unión Europea para apoyar la educación, la formación, la juventud y el deporte en Europa.

EDUTECH: Empresa que provee las mejores herramientas tecnológicas para el enriquecimiento de la enseñanza y aprendizaje.

GI-IATa: Grupo de Investigación en Inteligencia Artificial y Tecnologías de Asistencia.

UNESCO: Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura.

UDA: Universidad del Azuay.

COMPETENCIAS: Son los conocimientos, habilidades y actitudes necesarios para el correcto desempeño de una determinada actividad.

INCLUSIÓN: Hace referencia a que una persona con discapacidad tiene los mismos derechos, el acceso y las oportunidades que cualquier otra persona.

DISCAPACIDAD: Hace referencia a las limitaciones a las que se ven enfrentadas las personas a la hora de realizar cualquier tipo de actividad.

IDE: Un entorno de desarrollo integrado es una aplicación informática que proporciona un entorno de programación completo para los desarrolladores de software.

WCAG 2.0: Es un estándar técnico organizado en cuatro principios básicos para que cualquier persona pueda acceder y utilizar le contenido de la Web.

UNITY: Es una herramienta de desarrollo de videojuegos multiplataforma creada por la empresa Unity Technologies. Unity se utiliza para crear experiencias de Realidad Virtual interactivas.

ASSETS: Hace referencia a todo elemento o recurso que forma parte de un videojuego. Dichos elementos pueden ser desde geometrías hasta sonidos o animaciones.

INSPECTOR: Es una ventana en la que se puede editar todas las propiedades de cualquier elemento en Unity.

GAMEOBJECTS: Hace referencia a cualquier objeto de nuestro juego, los cuales representan a personajes, accesorios y escenarios.

COMPONENTES: Son la parte más importante de todo juego hecho en Unity, son los que nos permiten movernos, mostrar cosas en pantalla, reproducir sonidos, etc. Son las piezas fundamentales de cada GameObject.

SCRIPT: Es un documento que contiene instrucciones de programación en un determinado lenguaje, el cual hace las conexiones con el funcionamiento interno de Unity.

EVENTOS: Un sistema de eventos en Unity es una manera de enviar eventos a objetos en la aplicación basado en input, ya sea el teclado, mouse.

MONOBEHAVIOUR: Es la clase base que todo script de Unity debe heredar. Esta clase brinda un montón de funciones predefinidas las cuales se ejecutan cuando ocurre un evento.

INSTANCIA: Se llama instancia a todo objeto que se deriva de algún otro.

REFERENCIA: Una referencia nos permite manipular cualquier objeto, acceder a sus datos y a sus funciones públicas.

DEBUG: La depuración de programas es el proceso de identificar y corregir errores de programación.

HTTP: El protocolo de transferencia de hipertexto es el protocolo de comunicación que permite las transferencias de información a través de archivos en la World Wide Web.

URI: (Uniform Resource Identifier) cadena de caracteres que se refiere a un recurso. Los más comunes son URLs, que identifican el recurso dando su ubicación en la Web.

API: (Interfaz de Programación de Aplicaciones) Es un conjunto de requerimientos que se puede utilizar para que varias aplicaciones se comuniquen entre ellas de forma rápida y segura.

API REST: Es una interfaz que dos sistemas de computación utilizan para intercambiar información de manera segura a través de internet.

OPENGL: Se considera como una API que se utiliza para producir imágenes generadas por computadora de alta calidad y aplicaciones de gráficos interactivos mediante la representación de objetos geométricos 2D y 3D, mapas de bits e imágenes.

POST: Este método se utiliza para enviar información y datos al servidor.

DCU: (Diseño Centrado en el Usuario)Es una metodología de diseño que tiene como objetivo la creación de productos que satisfagan las necesidades y problemas del usuario.

CANVAS: El GameObject Canvas es un contenedor en donde podemos incluir todos los objetos que componen nuestra interfaz de usuario(botones, imágenes, texto…).

JAVASCRIPT: Es un lenguaje de programación orientado a objetos utilizado para el desarrollo y diseño de sitios web.

SERIALIZACIÓN: Es la manera de guardar los datos de nuestro juego en formatos universales que puedan ser guardados en el disco duro o transmitidos a través de una conexión en red.

Referencias

Abou Zahra, S. (2019, 10 de mayo). *Accessibility Principles*. World Wide Web Consortium. https://www.w3.org/WAI/fundamentals/accessibility-principles/

Adaptado de Creating and Using Scripts. (4 de diciembre de 2018). [Figura]. Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/CreatingAndUsingScripts.html

Aguilar Juárez, I. y Heredia Alonso, J. R. (2013). Simuladores y laboratorios virtuales para Ingeniería en Computación. *Revista Iberoamericana para la Investigación y el Desarrollo Educativo*. ISSN: 2007-2619, (10). https://1-11.ride.org.mx/index.php/RIDESECUNDARIO/article/viewFile/578/566

Aguilar Ortega, C. O., Tovar Luna, B. y Hernández-Cruz, B. A. (2018). Escenarios de aprendizaje basados en simulación: experiencia multidisciplinaria de la Universidad del Valle de México. *FEM: Revista de la Fundación Educación Médica, 21(4)*, 195-200. https://scielo.isciii.es/pdf/fem/v21n4/2014-9832-fem-21-4-195.pdf

Alamilla Hernández, L. M., Pérez Romero, V. A., Sosa González, S. A. y Valentín Rodríguez, J. A. (2021). Arquitectura REST para el desarrollo de aplicaciones web empresariales. *Revista Electrónica sobre Tecnología, Educación y Sociedad*, *8*(15). https://webcache.googleusercontent.com/search?q=cache:ogcW54BX6bQJ:https://www.ctes.org.mx/index.php/ctes/article/download/748/909/3027+&cd=13&hl=es&ct=clnk&gl=ec

Almenara Masbernat, M. (2018) *Modelo teórico-práctico para la implementación del diseño centrado en el usuario en el desarrollo, la validación y la aceptación de los productos de apoyo para personas con enfermedades de origen neurológico*. [Tesis Doctoral, Universitat Autónoma de Barcelona]. Repositorio Institucional UAB. https://ddd.uab.cat/pub/tesis/2018/hdl_10803_664003/mam1de1.pdf

Amaya Franky, G. (2006). Los entornos virtuales de simulación de la realidad, espacios vistos como ejes que permiten situar el aprendizaje dentro de un contexto institucionalizado de educación. *Ediciones Universidad de Salamanca*. *https://gredos.usal.es/bitstream/handle/10366/56519/TE_2006_V7N1_Entornosvirtuales.pdf?sequence=1&isAllowed=y*

Asamblea Nacional. (2019, 20 de febrero). Ley Orgánica de Comunicación. Registro Oficial Suplemento 22 de 25-jun.-2013. https://www.telecomunicaciones.gob.ec/wp-content/uploads/2020/01/Ley-Organica-de-Comunicaci%C3%B3n.pdf

Asistencia tecnológica a la accesibilidad en la Educación Superior Virtual. (2020, 2 de octubre). EduTech. https://psic-organizacional.uazuay.edu.ec/sites/psic-organizacional.uazuay.edu.ec/files/public/2020-12/PROYECTO%20ERAMUS%20EDUTECH%20II.pdf

Attributes. (2019, abril). Unity Documentation. https://docs.unity3d.com/es/current/Manual/Attributes.html

Ayón-Parrales, E. B. & Victores-Pérez, M.C. (2020). La simulación: Estrategia de apoyo en la enseñanza de las Ciencias Naturales en básica y bachillerato, Portoviejo, Ecuador. Dominio de las Ciencias, 6(3), 4-22. https://dialnet.unirioja.es/servlet/articulo?codigo=7467929

Azcuy, R. A. y Llull Céspedes, L. Á. (2021). Módulo de recomendación de patrones de diseño para EGPat. *Revista Cubana de Ciencias Informáticas, 15* (12), 118-137. https://www.redalyc.org/journal/3783/378367420007/html/

Bonacic Vargas, Y. (2016). *SIGEM - simulación de gestión de empresas-, un modelo de juego de negocios para el desarrollo de las competencias genéricas universales (CGU) en la educación superior en Chile*. [Tesis Doctoral, Universidad de Sevilla] Repositorio Institucional idus. https://idus.us.es/handle/11441/36517

Bonilla Trujillo, D., Villamil Reyes, V. V. y Montes Mora, J. F. (2019). Uso de simuladores 3D como estrategia tecnopedagógica para la transferencia de conocimiento en el aprendizaje de la anatomía animal. *Documentos De Trabajo ECAPMA*, (1). https://hemeroteca.unad.edu.co/index.php/workpaper/article/view/3414/3378

Briones Quiroz, A., Cruzado Suarez, O., Huaman Mendo, D., Rabanal Dos Santos, F., Sánchez Carranza, D. y Vera Arce, I. (2016). Lenguaje de programación C#: Fundamentos de Algoritmos. *Universidad Privada del Norte*. https://www.academia.edu/28564115/Lenguaje_de_programaci%C3%B3n_C_

Cabero Almenara, J. y Costas, J. (2016). La utilización de simuladores para la formación de los alumnos. *Prisma social, (17*), 343-372. https://www.redalyc.org/pdf/3537/353749552015.pdf

Caldwell, B., Cooper, M., Guarino Reid, L., y Vanderheiden, G. (2008, 11 de diciembre). *Web Content Accessibility Guidelines (WCAG) 2.0.* World Wide Web Consortium. https://www.w3.org/TR/WCAG20/#perceivable

Campoverde Molina, M., Luján Mora, S. y García, L. V. (2020) Empirical Studies on Web Accessibility of Educational Websites: A Systematic Literature Review. *IEEE Access*, *vol. 8*, pp. 91676-91700. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9092982

Candelaria Martínez, B., Ruiz Rosado, O., Gallardo López, F., Pérez Hernández, P., Martínez Becerra, Á. y Vargas Villamil, L. (2011). Aplicación de modelos de simulación en el estudio y planificación de la agricultura, una revisión. *Tropical and subtropical agroecosystems, 14(3)*, 999-1010. http://www.scielo.org.mx/pdf/tsa/v14n3/v14n3a4.pdf

Casal Angulo, M.C. (2016). *La simulación como metodología para el aprendizaje de habilidades no técnicas en Enfermería*. Tesis Doctoral, Universitat de Valéncia]. Repositorio Institucional UV. https://core.ac.uk/download/pdf/71059825.pdf

Castillero Mimenza, O. (2018, 16 de agosto). *Los 6 tipos de discapacidad y sus características*. Psicología y Mente. https://psicologiaymente.com/salud/tipos-de-discapacidad

Chaile Alfaro, I. F. (2015). *Cómo utilizar simuladores basados en multiagentes para SCADA en aplicaciones del ámbito industrial*. [Tesis Doctoral, Universitat Autònoma de Barcelona]. Repositorio Institucional UAB. https://www.tesisenred.net/bitstream/handle/10803/384007/isfachal1de1.pdf?sequence=6&isAllowed=y

Chisholm, W., Vanderheiden, M.n G. y Jacobs, M. I. (2021, 18 de mayo). *Web Content Accessibility Guidelines 1.0*. World Wide Web Consortium. https://www.w3.org/TR/WAI-WEBCONTENT/

Controlar GameObjects utilizando componentes. (2019, abril). Unity Documentation. https://docs.unity3d.com/es/current/Manual/ControllingGameObjectsComponents.html

Costas Santos, J. (2013). *Análisis, diseño, construcción y evaluación de simuladores para la familia profesional de Informática y Comunicaciones*. [Tesis Doctoral, Universidad de Sevilla] Repositorio Institucional idUS. https://idus.us.es/handle/11441/15014

Creating and Using Scripts. (2018, 19 de marzo). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/CreatingAndUsingScripts.html

Crespo Amigo, J. (2012). *Diseño y desarrollo de un juego en WebGL*. [Proyecto Final de Carrera, Universidad Politécnica de Catalunya]. Repositorio institucional UPC. https://upcommons.upc.edu/bitstream/handle/2099.1/15478/82457.pdf

¿Cuáles son las discapacidades sensoriales? (s.f.) National Rehabilitation Information Center. Consultado el 04 de junio de 2022. https://naric.com/?q=es/content/selecciones-del-bibliotecario-discapacidades-sensoriales

Cumbreño Juan, R. (2017*). Desarrollo de videojuegos: accesible vs no accesible*. [Tesis de Grado, Universidad de Alcalá]. Repositorio Institucional e_Buah. https://ebuah.uah.es/dspace/bitstream/handle/10017/31983/TFG-Cumbre%c3%b1o-Juan-2017.pdf?sequence=1&isAllowed=y

Deshpande, S. kank, t., Armanyous, M., Singh, S. y Kalita, M. (2020): Improvised learning for pre-primary students using augmented reality. *TechRxiv*. Preprint. https://doi.org/10.36227/techrxiv.12056046.v1

Díaz Martínez, M. A., Zárate Cruz, R. y Román Salinas, R. V. (2018). Simulación FlexSim, una nueva alternativa para la ingeniería hacia la toma de decisiones en la operación de un sistema de múltiples estaciones de prueba. *Científica*, *22*(2), 97-104. https://www.redalyc.org/journal/614/61458109002/html/

Discapacidad intelectual. (s.f.) Medline Plus. Consultado el 05 de junio de 2022. https://medlineplus.gov/spanish/ency/article/001523.htm#:~:text=Es%20una%20afecci%C3%B3n%20diagnosticada%20antes,usaba%20para%20describir%20esta%20afecci%C3%B3n.

Discapacidad psíquica. (s.f.) Corresponsables. Consultado el 05 de junio de 2022. https://ecuador.corresponsables.com/content/discapacidad-ps%C3%ADquica

Dodero, J. M. y Ghiglione, E. (2008). ReST-based web access to learning design services. *IEEE Transactions on Learning Technologies*, *1*(3), 190-195. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4721288

El desempleo entre los jóvenes con discapacidad se dispara un 16%. (2021, 3 de agosto). Compromiso RSE. https://normas-apa.org/referencias/citar-pagina-web/

Espin Loachamin, A., Iza Carrera, D. y Paredes Amaguaya, A. (2022). Diseño centrado en el usuario para la creación de un catálogo de productos de consumo masivo. *Polo del Conocimiento, 7*(4), 650-661. https://polodelconocimiento.com/ojs/index.php/es/article/view/3848/8914

Etter, P. C. (2018). *Underwater acoustic modeling and simulation*. CRC press. https://www.taylorfrancis.com/books/mono/10.1201/9781315166346/underwater-acoustic-modeling-simulation-paul-etter

Evvano, P. (26 de octubre de 2021). *Web accessibility: what it is and why it's important for business*. Logic20/20. https://www.logic2020.com/insight/web-accessibility-why-important-for-business#

Event Functions (Funciones de Evento). (2019, abril). Unity Documentation.

Fonseca Romero, J.C. (2015). Diseño de un Ambiente Simulado para Seguridad de la Información. *Revista Ciencia, Innovación y Tecnología (RCIYT), Vol. I*. https://revista.jdc.edu.co/index.php/rciyt/article/view/117/105

Gaba D. M. (2004). The future vision of simulation in health care. Quality & safety in health care, 13 Suppl 1(Suppl 1), i2–i10. https://doi.org/10.1136/qhc.13.suppl_1.i2

Galeano, R. (2008). Diseño centrado en el usuario. *Revista Q Educación Comunicación Tecnología, Vol. 2 No. 4.* https://repository.upb.edu.co/bitstream/handle/20.500.11912/6524/Dise%c3%b1o%20centrado%20en%20el%20usuario.pdf?sequence=1&isAllowed=y

Garzón Romero, J. A. (2011) *Diseño de ambientes de simulación para prácticas en la formación laboral en el CGMLTIC-SENA D.C.* [Tesis de Maestría, Universidad Nacional de Colombia]. Repositorio Institucional UNAL. https://repositorio.unal.edu.co/handle/unal/21280

Gaviño Ortiz, G., Vásquez Godínez, S., Barrios Borjes, E. y Velarde Martínez, J. E. (2021). Procedimiento matemático, orientado a la simulación en FLEXSIM, mediante un sistema de enseñanza de planificación de requerimientos de materiales (MRP). Revista Investigación Operacional, 42 (3), 409-421. https://rev-inv-ope.pantheonsorbonne.fr/sites/default/files/inline-files/42321-12.pdf

Getting started with WebGL development. (202, 14 de junio). Unity Documentation. https://docs.unity3d.com/2018.4/Documentation/Manual/webgl-gettingstarted.html

González, A. H. y Vallejo, A. E. (2021). Desarrollo de escenarios educativos digitales de decisión. [Archivo PDF]. http://sedici.unlp.edu.ar/bitstream/handle/10915/118306/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Guerrero Caballero, V. C. (2020) *Aplicación del simulador de negocios SIMDEF para el fortalecimiento del aprendizaje de contabilidad en los estudiantes universitarios*. [Tesis de Grado, Universidad de San Martín de Porres] Repositorio Institucional USMP. https://repositorio.usmp.edu.pe/handle/20.500.12727/6869

Hacer que los juegos sean accesibles. (2022, 2 de abril). Microsoft. https://docs.microsoft.com/es-es/windows/uwp/gaming/accessibility-for-games

Herrera, O. J. y Becerra, L. A. (2014). Diseño General de las Etapas de Simulación de Procesos con Énfasis en el Análisis de Entrada. In *12th Latin American and Caribbean Conference for Engineering and Technology* (Vol. 10). http://www.laccei.org/LACCEI2014-Guayaquil/RefereedPapers/RP152.pdf

Important Classes - Time and Framerate Management. (2019, abril). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/TimeFrameManagement.html

Instituto Ecuatoriano de Normalización. (2014, 28 de enero) Norma Técnica Ecuatoriana NTE INEN-ISO/IEC 40500. Registro Oficial Nº 171. https://docplayer.es/34772428-Nte-inen-iso-iec-40500.html

Introduction to Web Accessibility. (2020, 14 de abril). Web Accessibility in Mind. https://webaim.org/intro/

Interfaz de usuario de Unity. (2018, 4 de diciembre). App Game Tutorials. https://appgametutoriales.com/interfaz-de-usuario-de-unity/

Introducción a .NET Framework. (2022, 23 de mayo). Microsoft. https://docs.microsoft.com/es-es/dotnet/framework/get-started/

Jofre, C. M. y Fernández Zalazar, D. (2019). Sociedad del Conocimiento y accesibilidad para la educación inclusiva. *Iberoamérica Social: revista-red de estudios sociales XII, 7*, 97 – 117. https://dialnet.unirioja.es/servlet/articulo?codigo=6992803

Keio, B. (2018). *Accessibility*. World Wide Web Consortium. https://www.w3.org/standards/webdesign/accessibility

La discapacidad física: ¿qué es y qué tipos hay? (2022). Observatorio Discapacidad Física. https://www.observatoridiscapacitat.org/es/la-discapacidad-fisica-que-es-y-que-tipos-hay

Le damos la bienvenida al IDE de Visual Studio. (2022, 21 de mayo). Microsoft. https://docs.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2022&viewFallbackFrom=vs-2022%C3%A7

Matosas López, L. y Romero Luis, J. (2019). Correlaciones entre redes sociales y recursos educativos digitales en estudiantes universitarios de Marketing en el EEES. *Contenidos audiovisuales, narrativas y alfabetización mediática; Sierra Sánchez, J., Ed*, 393-402. https://www.researchgate.net/profile/Luis-Matosas-Lopez/publication/333176428_Correlaciones_entre_redes_sociales_y_recursos_educativos_digitales_en_estudiantes_universitarios_de_Marketing_en_el_EEES/links/5fce5f7aa6fdcc697be9df72/Correlaciones-entre-redes-sociales-y-recursos-educativos-digitales-en-estudiantes-universitarios-de-Marketing-en-el-EEES.pdf

Martín Fernández, F. J. y Hassan Montero, Y. (2004). Propuesta de adaptación de la metodología de diseño centrado en el usuario para el desarrollo de sitios web accesibles. *Revista Española De Documentación Científica*, *27*(3), 330–344. http://riberdis.cedid.es/bitstream/handle/11181/3877/propuesta_de_adaptacion_de_la_metodologia.pdf?sequence=1&rd=0031142128525955

Monroy Osorio, J.C. (2012). *Informe de proyecto de grado presentado como requisito final para optar al título de Magíster en Ingeniería con especialidad en Tecnologías de información para Educación*. [Tesis de Maestrìa, Universidad EAFIT]. Repositorio Institucional EAFIT. https://repository.eafit.edu.co/bitstream/handle/10784/1411/MonroyOsorio_JuanCarlos_2012.pdf;jsessionid=0557858E12C4CDB83885194DD0D604FC?sequence=3

Nakov, S. y Kolev, V. (2013). *Fundamentals of Computer Programming with C#: The Bulgarian C# Book*. Faber Publishing. https://introprogramming.info/wp-content/uploads/2013/07/Books/CSharpEn/Fundamentals-of-Computer-Programming-with-CSharp-Nakov-eBook-v2013.pdf

Namespaces. (2019, abril). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/Namespaces.html

Neri Vela, R. (2017). El origen del uso de simuladores en Medicina. *Revista de la Facultad de Medicina UNAM, 60*, 21-27. https://www.medigraphic.com/pdfs/facmed/un-2017/uns171c.pdf

Null Reference Exceptions. (2019, abril). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/NullReferenceException.html

Organización de las Naciones Unidas, Asamblea General. (2007). Convención sobre los derechos de las personas con discapacidad. https://treaties.un.org/doc/Publication/CTC/Ch_IV_15.pdf

Organización Internacional del Trabajo. (2021). *Discapacidad y trabajo*. https://www.ilo.org/skills/about/WCMS_475652/lang--es/index.htm

Order of execution for event Functions. (2019, 18 de marzo). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/ExecutionOrder.html

Organización Mundial de la Salud (OMS) y Discapacidad. (2000). Bienestar y Protección Infantil. https://www.bienestaryproteccioninfantil.es/fuentes1.asp?sec=17&subs=202&cod=1873&page=

Paseo por el lenguaje C#. (2022, 23 de mayo) Microsoft. https://docs.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/

Primeros Pasos con WebGL. (2022). MDN PLUS. https://developer.mozilla.org/es/docs/Web/API/WebGL_API/Tutorial/Getting_started_with_WebGL

Ribelles, J., & López, A. (2019). Informática gráfica. *Segunda Edición*. Publicacions de la Universitat Jaume I. http://repositori.uji.es/xmlui/bitstream/handle/10234/182894/s151.pdf?sequence=1&isAllowed=y

Qué es Discapacidad. (2018, 28 de marzo). Instituto Tlaxcalteca para Personas con Discapacidad. https://www.itpcd.gob.mx/index.php/que-es-discapacidad

¿Qué es REST (Transferencia de Estado REpresentacional)? (2020). Krypton. https://kryptonsolid.com/que-es-rest-transferencia-de-estado-representacional/

Sam Anlas, C. A. y Stable Rodríguez, Y. (2016). Evaluación de la accesibilidad web de los portales del Estado en Perú. Revista Española de Documentación Científica. 39 (1). e120. https://redc.revistas.csic.es/index.php/redc/article/view/923

Sandí Delgado, J. C. y Sanz, C. V. (2020). Juegos serios para potenciar la adquisición de competencias digitales en la formación del profesorado. *Revista Educación, 44(1)*, 471-489. https://revistas.ucr.ac.cr/index.php/educacion/article/view/37228/40941

Toledo Toledo, G., Nieva García, O. y Bezares Molina, G. (2019). Aplicación del diseño centrado en el usuario en curso universitario de interacción humano computadora para estudiantes de

ingeniería en computación. *Virtualidad, Educación Y Ciencia*, *10*(18), 81–99. https://revistas.unc.edu.ar/index.php/vesc/information/readers

Tortarolo, P. (2018*). Incorporación de TICs en la formación de los Oficiales del Instituto Universitario de Seguridad Marítima de la Prefectura Naval Argentina: el caso de la enseñanza mediante Simulador de Navegación*. [Tesis de Maestría, Universidad de San Andrés] Repositorio Institucional UDESA. https://repositorio.udesa.edu.ar/jspui/handle/10908/16684

UnityEvents. (2019, abril). Unity Documentation. https://docs.unity3d.com/2019.4/Documentation/Manual/UnityEvents.html

Vanegas Prieto, M. F. (2020, 10 de noviembre). *Modelos 3D*. Niixer. https://niixer.com/index.php/2020/11/10/modelos-3d/

Vargas Murillo, G. (2017). Recursos educativos didácticos en el proceso enseñanza aprendizaje. *Cuadernos Hospital de Clínicas, 58(1),* 68-74. http://www.scielo.org.bo/pdf/chc/v58n1/v58n1_a11.pdf

Variables and the Inspector. (2019, abril). Unity Documentation. https://docs.unity3d.com/es/current/Manual/VariablesAndTheInspector.html

Vidal Ledo, M. J., Martínez, R. A., Rodríguez Monteagudo, M. A., Menéndez Bravo, J. A. (2019). Simuladores como medios de enseñanza. *Revista Cubana de Educación Médica Superior, 33(4),* 37-49. https://www.medigraphic.com/pdfs/educacion/cem-2019/cem194j.pdf

Web Accessibility. (s.f.). Technology Accessibility. The University of Alabama. https://accessibility.ua.edu/accessibilityresources/web-accessibility/

Web Accessibility Principles. (2020, 21 de septiembre). Web Accessibility in Mind. https://webaim.org/resources/quickref/

Zurita López, S. R. (2015). *Simuladores virtuales como recurso didáctico para fortalecer el interaprendizaje en las prácticas de laboratorio de física del primer año de bachillerato del Colegio Nacional Mariano Benítez*. [Tesis de Maestría, Universidad Católica del Ecuador] Repositorio Institucional PUCESA. https://repositorio.pucesa.edu.ec/handle/123456789/1196

4 Most Common Types of Sensory Disabilities. (s.f.). Enable Me. Consultado el 04 de junio de 2022. https://www.enableme.ke/en/article/4-most-common-types-of-sensory-disabilities-1535#:~:text=Sensory%20disabilities%20are%20the%20disabilities,%2C%20illnesses%2C%20or%20environmental%20factors

ANEXOS

Anexos A

Encuesta Realizada

Investigadores:

Est. Narcisa de Jesús Araujo Pérez

Dra. Monica Rodas

Ing. Paola Ingavélez G.

Tiempo total estimado de todas las actividades:

Nombre del participante:

_____

Edad:

_____

Posee Discapacidad: si/no, Especifique:

_____

Profesión/Carrera:

_____

Fecha:

_____

1.1 Formato de encuestas realizadas a participantes

Simulador Información Difícil

Cuando termine, nuestro compañero entrevistador le preguntará y registrará los siguientes datos:

1. El acceso al ejercitario le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

2. Los íconos del ejercitario indican claramente la acción que realizan:

O Si O Parcialmente O No

3. La información presentada sobre el ejercitario le pareció:

O Muy relevante O Algo relevante O Relevante O Poco relevante O Nada relevante

4. El escenario de la simulación con respecto al ejercitario le pareció:

O Muy adecuado O Adecuado O Regular O inadecuado O Muy inadecuado

5. La interacción con la escena le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

6. El acceso a las preferencias le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

7. ¿Qué tan fácil fue cambiar el tipo de fuente?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

8. ¿Qué tan fácil fue cambiar el fondo?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

9. ¿Qué tan fácil fue cambiar el tamaño de letra?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

10. ¿Qué tan fácil fue activar o desactivar el audio?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

11. Las opciones de preferencias le parecieron:

O Muy completas O Algo completas O completas O incompletas O Muy incompletas

12. La navegación por la interfaz a través de ratón le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

13. La navegación por la interfaz a través de teclado le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

14. Las actividades planteadas le parecieron:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

15. La retroalimentación proporcionada al terminar el ejercitario le pareció:

O Muy relevante O Algo relevante O Relevante O Poco relevante O Nada relevante

16. Con respecto al ejercitario que acaba de realizar piensa que:

O Si pudo realizarlo

O No pudo realizarlo

O No está seguro de haber finalizado

17. ¿En una escala del 0 al 10, cuál es su nivel de expectativa antes de ingresar al ejercitario?

Muy bajo 1 2 3 4 5 6 7 8 9 10 Muy alto

O O O O O O O O O O

18. ¿En una escala del 0 al 10, cuál es su nivel de expectativa después de interactuar con el ejercitario?

Muy bajo 1 2 3 4 5 6 7 8 9 10 Muy alto

O O O O O O O O O O

Argumenta las respuestas 17 y 18 sobre el simulador laboral:

Simulador El Tiempo

Cuando termine, nuestro compañero entrevistador le preguntará y registrará los siguientes datos:

1. El acceso al ejercitario le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

2. Los íconos del ejercitario indican claramente la acción que realizan:

O Si O Parcialmente O No

3. La información presentada sobre el ejercitario le pareció:

O Muy relevante O Algo relevante O Relevante O Poco relevante O Nada relevante

4. El escenario de la simulación con respecto al ejercitario le pareció:

O Muy adecuado O Adecuado O Regular O inadecuado O Muy inadecuado

5. La interacción con la escena le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

6. El acceso a las preferencias le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

7. ¿Qué tan fácil fue cambiar el tipo de fuente?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

8. ¿Qué tan fácil fue cambiar el fondo?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

9. ¿Qué tan fácil fue cambiar el tamaño de letra?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

10. ¿Qué tan fácil fue activar o desactivar el audio?

O Muy fácil O Fácil O Regular O Difícil O Muy difícil

11. Las opciones de preferencias le parecieron:

O Muy completas O Algo completas O completas O incompletas O Muy incompletas

12. La navegación por la interfaz a través de ratón le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

13. La navegación por la interfaz a través de teclado le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

14. Las actividades planteadas le parecieron:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

15. La asignación de valores correspondiente a jerarquizar cada actividad, le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

16. La asignación de valores correspondientes a delimitar tiempos de cada actividad, le pareció:

O Muy fácil O Fácil O Regular O Difícil O Muy Difícil

17. La retroalimentación proporcionada al terminar el ejercitario le pareció:

O Muy relevante O Algo relevante O Relevante O Poco relevante O Nada relevante

18. Con respecto al ejercitario que acaba de realizar piensa que:

O Si pudo realizarlo

O No pudo realizarlo

O No está seguro de haber finalizado

19. ¿En una escala del 0 al 10, ¿cuál fue su nivel de expectativa antes de ingresar al ejercitario?

Muy bajo	1	2	3	4	5	6	7	8	9	10	Muy alto
O	O	O	O	O	O	O	O	O	O		

20. ¿En una escala del 0 al 10, ¿cuál es su nivel de expectativa después de interactuar con el ejercitario?

Muy bajo	1	2	3	4	5	6	7	8	9	10	Muy alto
O	O	O	O	O	O	O	O	O	O		

Argumenta las respuestas 19 y 20 sobre el simulador laboral:

Preguntas de Cierre (10 min)

1. **¿Considera que Los retos planteados fueron factibles de realizar?**

_____
_

2. **¿Considera que el tiempo empleado para interactuar con los ejercitarios fue el adecuado?**

_____
_

3. **¿Cuál fue la parte que le gustó de cada simulador laboral?**

_____

_

4. **¿Cuál fue la parte que le gusto menos de cada simulador laboral?**

_____

_

Tareas de las personas incluidas en el estudio

Est. Narcisa de Jesús Araujo Pérez: Explicación del proyecto Edutech en la Universidad y sus objetivos, presentación del proyecto desarrollado.

Ing. Paola Ingavélez G: Planificación y gestión de espacios y usuarios entrevistados.

Dra. Mónica Rodas: Validación y aprobación del plan de pruebas.

Reporte de Evaluación

Con los datos obtenidos se procederá a realizar las tabulaciones en base a las preguntas planteadas y su grado de cumplimiento. Adicionalmente se considera.

- Bitácoras de tiempos empleados por parte de los entrevistados.

- Errores detectados con descripción de cada uso.

- Comentarios identificados de nuestros usuarios.

- Fotografías

Anexos B

Protocolo de Evaluación

Figura 102. Evaluación de Simuladores Laborales con Roberto Pacho.

Figura 103. Evaluación de Simuladores Laborales con Noe Ayavaca.

Figura 104. Evaluación de Simuladores Laborales con Michelle Peñalosa.

Figura 105. Evaluación de Simuladores Laborales con Fernando Deleg.

Figura 106. Evaluación de Simuladores Laborales con Fabian Armijos.

Figura 107. Evaluación de Simuladores Laborales con Gabriel Chuchuca.

Anexos C

Modelado de Escenario, Personaje e Interfaz de Usuario Accesible

Escenario Información Difícil

Figura 108. Vista entrada oficina rectorado.

Figura 109. Vista parte interna de oficina.

Escenario El Tiempo

Figura 110. Vista escenario de Talento Humano.

Figura 111. Vista escenario de Talento Humano.

Personaje Simuladores Laborales

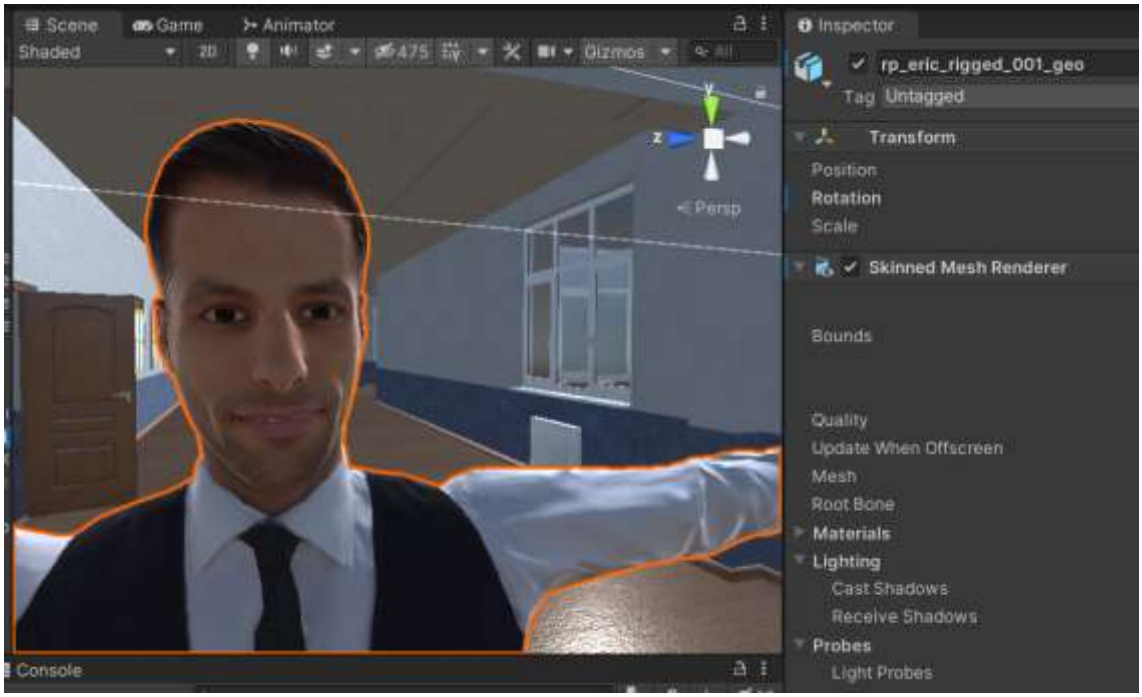


Figura 112. Personaje utilizado para simuladores laborales.

Figura 113. Diseño de interfaz de usuario de simuladores laborales.

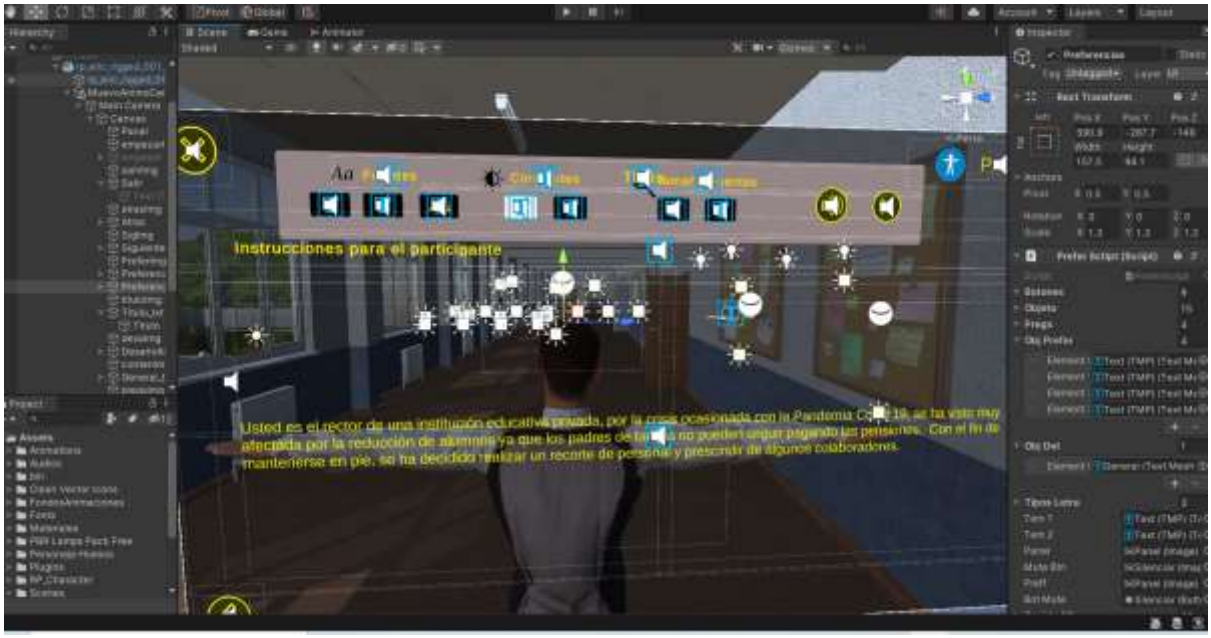


Figura 114. Diseño de barra de instrucciones de simuladores laborales.

Anexos D

Animación de Personaje e interfaz de usuario

Animación Personaje Simuladores Laborales

Animación Personaje Simulador Laboral Información Difícil

Figura 115. Animación del personaje en el escenario del Simulador Laboral Información Difícil.

Figura 116. Animación del rostro del personaje del Simulador Laboral Información Difícil.

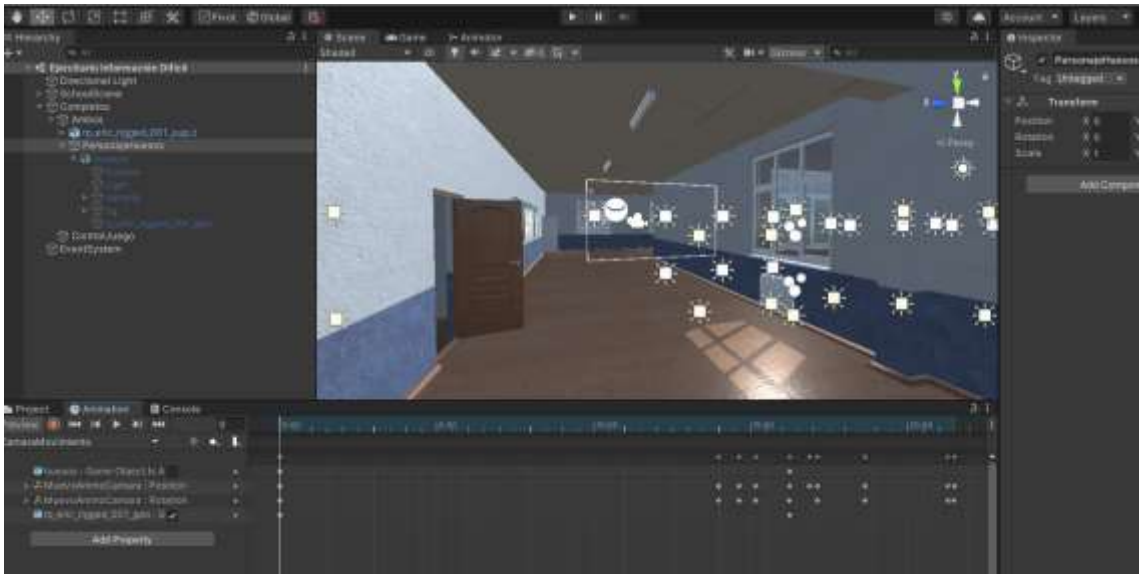


Figura 117. Animación vista de cámara en primera persona correspondiente al Simulador Laboral Información Difícil.

Animación personaje Simulador Laboral El Tiempo

Figura 118. Animación de personaje en el escenario del Simulador Laboral El Tiempo.

Figura 119. Animación de vista de cámara en primera persona correspondiente al personaje del Simulador Laboral El Tiempo.

Animación de Interfaz de usuario

Figura 120. Animación creada dentro del Animator Controler Canvas correspondiente a la interfaz de usuario del Simulador Laboral Información Difícil.

Figura 121. Animación creada dentro del Animator Controler Canvas correspondiente a la interfaz de usuario del Simulador Laboral el Tiempo.

Anexos E

Simuladores Laborales 3D en ejecución

Simulador Información Difícil en Ejecución

Figura 122. Vista inicial de ejecución de simulador laboral Información Difícil.

Figura 123. Vista ingreso de personaje a oficina.

Figura 124. Vista expresión de preocupación del personaje a causa de la difícil decisión que debe tomar.

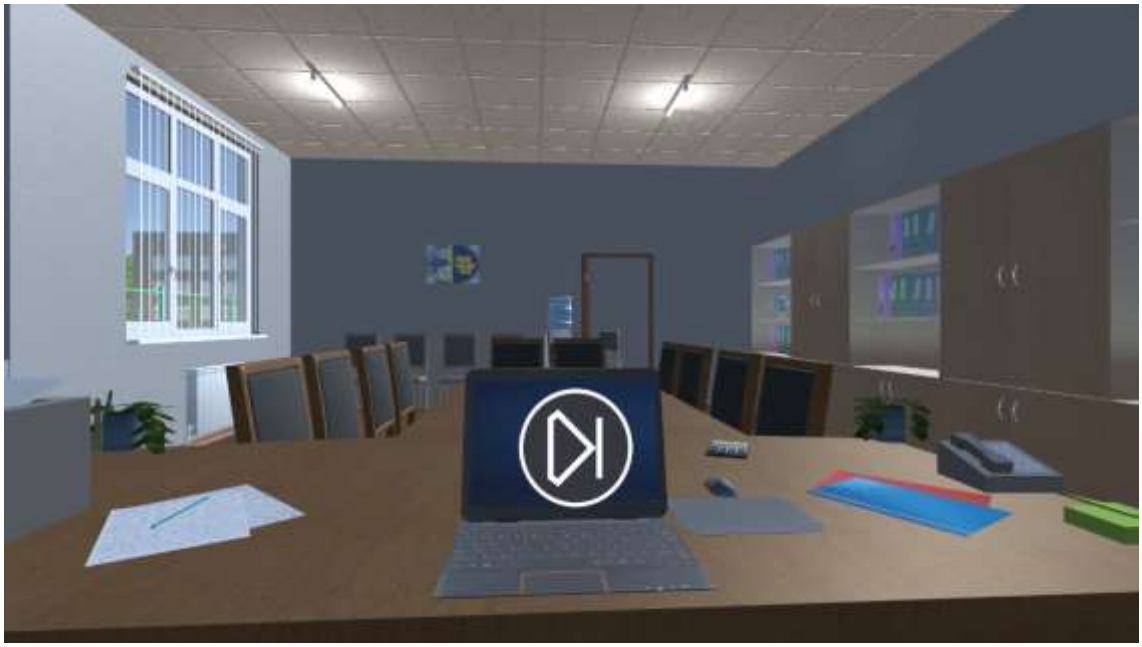


Figura 125. Vista panel en donde se muestra el botón empezar del simulador laboral.

Figura 126. Vista de interacción de personaje con el escenario, en el cual puede mover la vista de la cámara en primera persona e interactuar con objetos del mismo.

Figura 127. Vista en donde el personaje interactúa con el botón empezar.

Figura 128. Vista del primer panel en donde se presentan instrucciones para el participante referente al ejercitario (ejercicio) planteado.

Figura 129. Vista interacción de participante con el botón siguiente mediante el teclado.

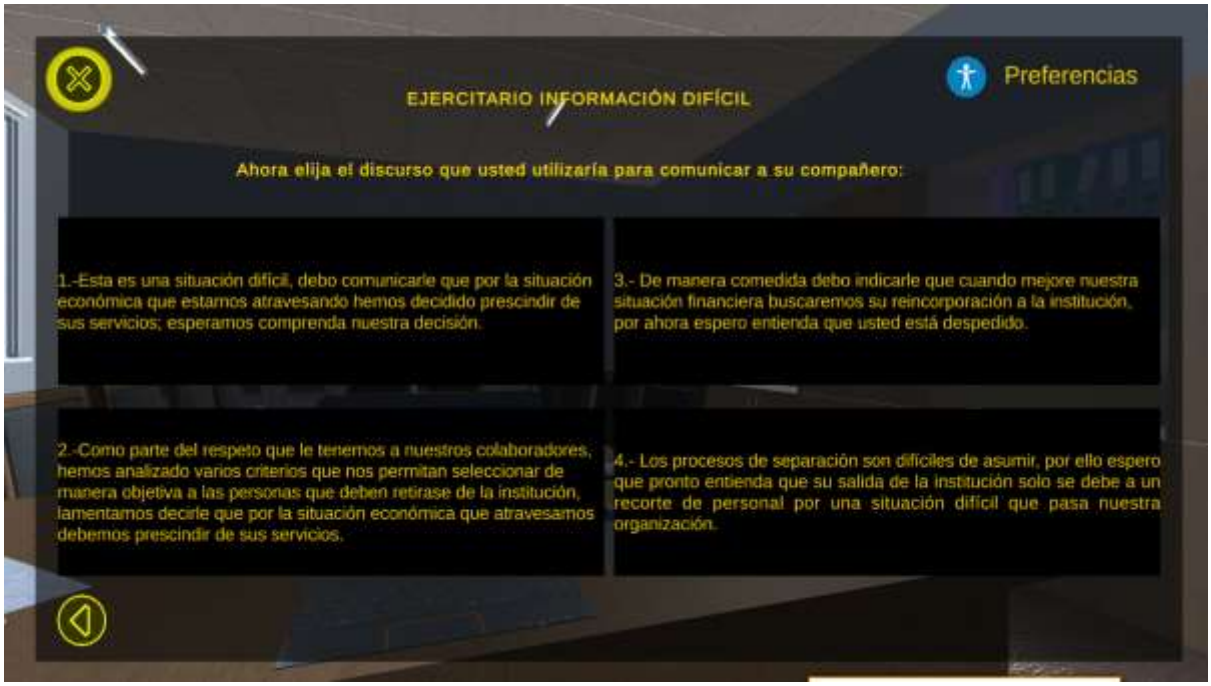


Figura 130. Vista panel de actividades presentadas al participante.

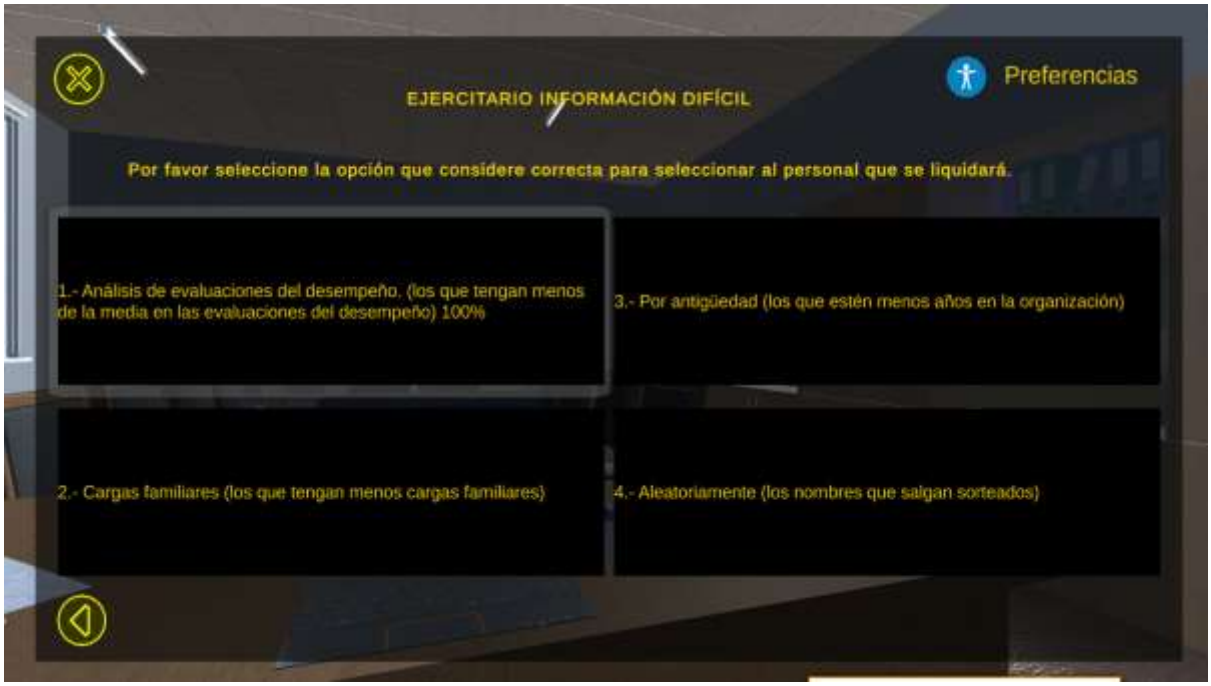


Figura 131. Vista de interacción del participante con las actividades planteadas mediante teclado.

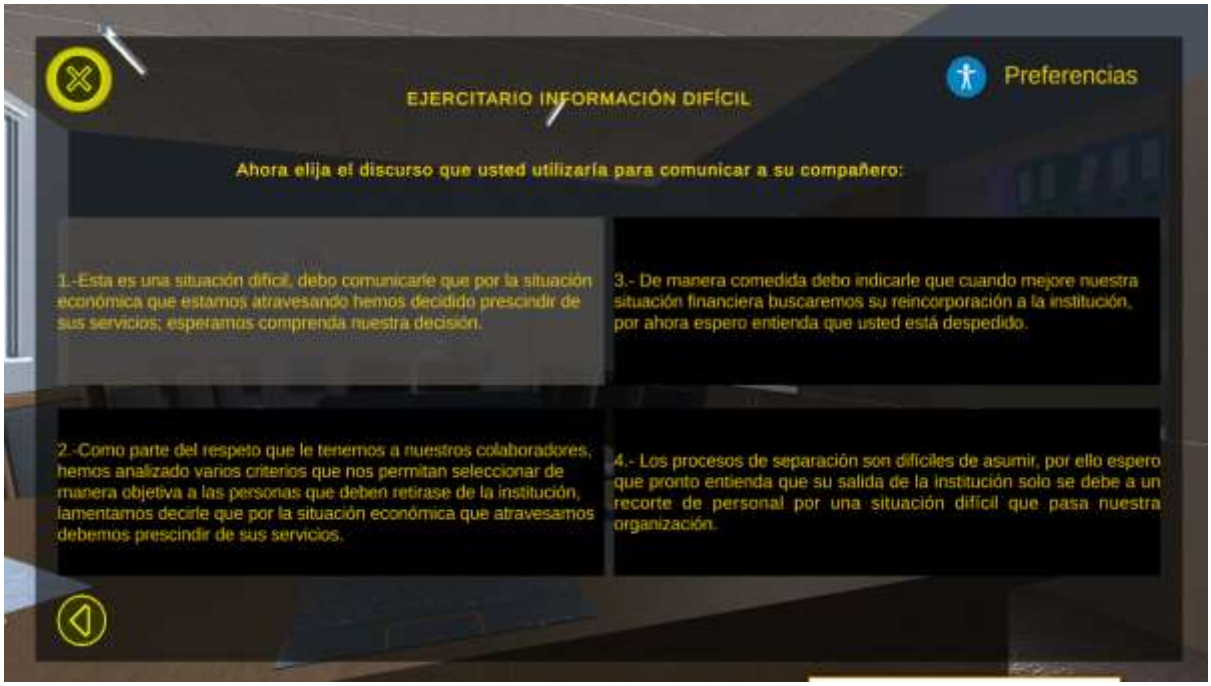


Figura 132. Vista de interacción del participante con las actividades presentadas mediante el mouse.

Figura 133. Vista en donde el participante revisa la calificación obtenida luego de haber finalizado la interacción con las actividades planteadas; cuando el participante llegue a este panel ya no podrá regresar.

Figura 134. Vista final del simulador laboral en donde el participante revisa la retroalimentación referente al ejercitario (ejercicio) planteado.

Figura 135. Vista de interacción de participante con el botón volver a menú principal.

- **Interacción del participante con las distintas opciones de accesibilidad del simulador laboral**

Figura 136. Vista de participante interactuando con la barra de preferencias del simulador laboral mediante el teclado.

Figura 137. Vista en donde el participante selecciona la fuente de texto Arial Bold.

Figura 138. Vista en donde el participante selecciona la fuente de texto Lato.

Figura 139. Vista en donde el participante selecciona la fuente Dyslexic.

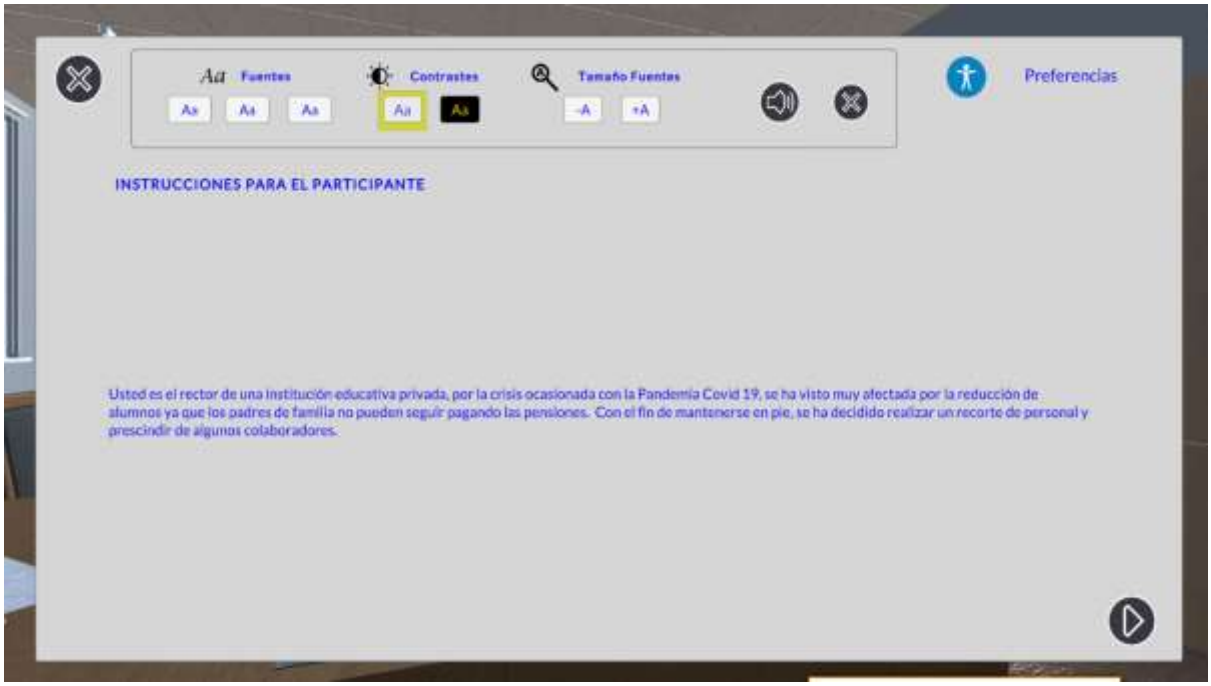


Figura 140. Vista en donde el participante selecciona un contraste claro.

Figura 141. Vista en donde el participante selecciona un contraste oscuro.

Figura 142. Vista en donde participante presiona el botón de reducir el tamaño de texto.

Figura 143. Vista en donde el participante presiona el botón para aumentar el tamaño de texto.

Figura 144. Vista en donde el participante desactiva la descripción de audios.

Figura 145. Vista en donde el participante cierra la barra de preferencias al presionar el botón salir de preferencias.

Simulador Laboral El Tiempo en Ejecución

Figura 146. Vista inicial de ejecución de simulador laboral El Tiempo.

Figura 147. Vista de ingreso de personaje a oficina.

Figura 148. Vista personaje acercándose a su puesto de trabajo.

Figura 149. Vista panel en donde se muestra el botón empezar del simulador laboral.

Figura 150. Vista de interacción de personaje con el escenario, en el cual puede mover la vista de la cámara en primera persona e interactuar con objetos del mismo.

Figura 151. Vista interacción de personaje con botón empezar, en donde puede hacerlo mediante el teclado o el mouse.

Figura 152. Vista del primer panel en donde se presentan instrucciones para el participante referente al ejercitario (ejercicio) planteado.

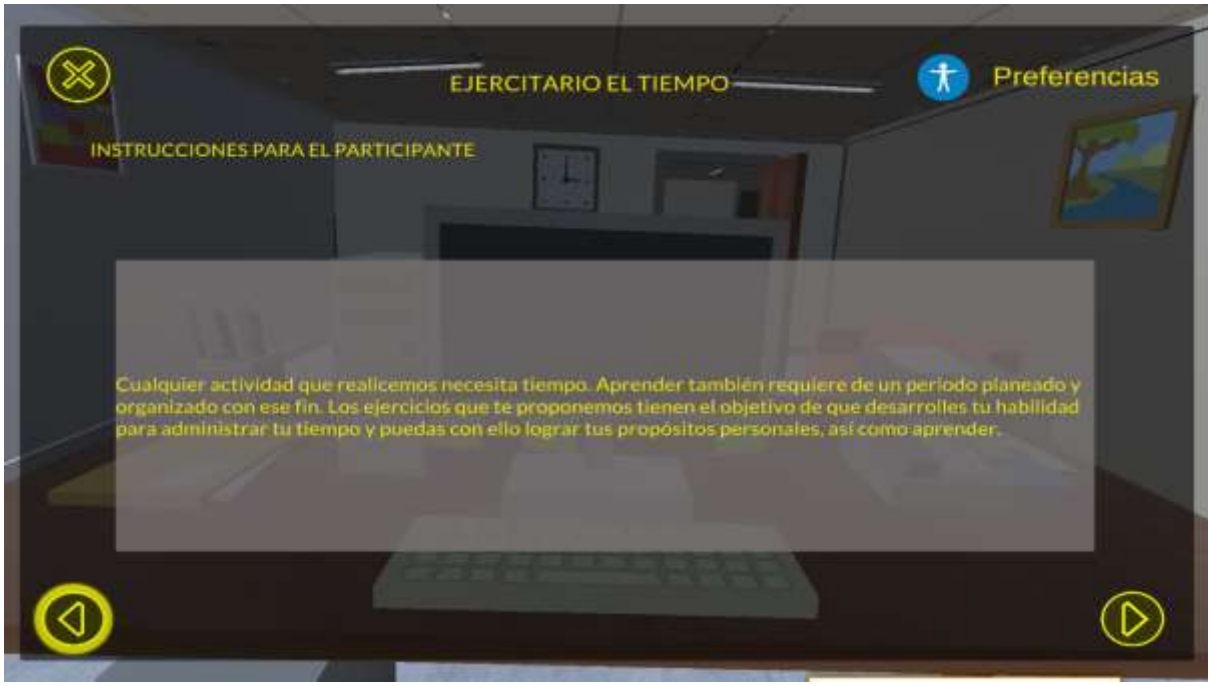


Figura 153. Vista en donde el participante interactúa con el panel de instrucciones e1 mediante el mouse.

Figura 154. Vista de segundo panel de instrucciones, en donde el participante interactúa con el botón siguiente mediante el mouse.

Figura 155. Vista de tercer panel de instrucciones en donde el participante interactúa con el teclado.

Figura 156. Vista de cuarto panel de instrucciones, en donde le participante interactúa con el teclado.

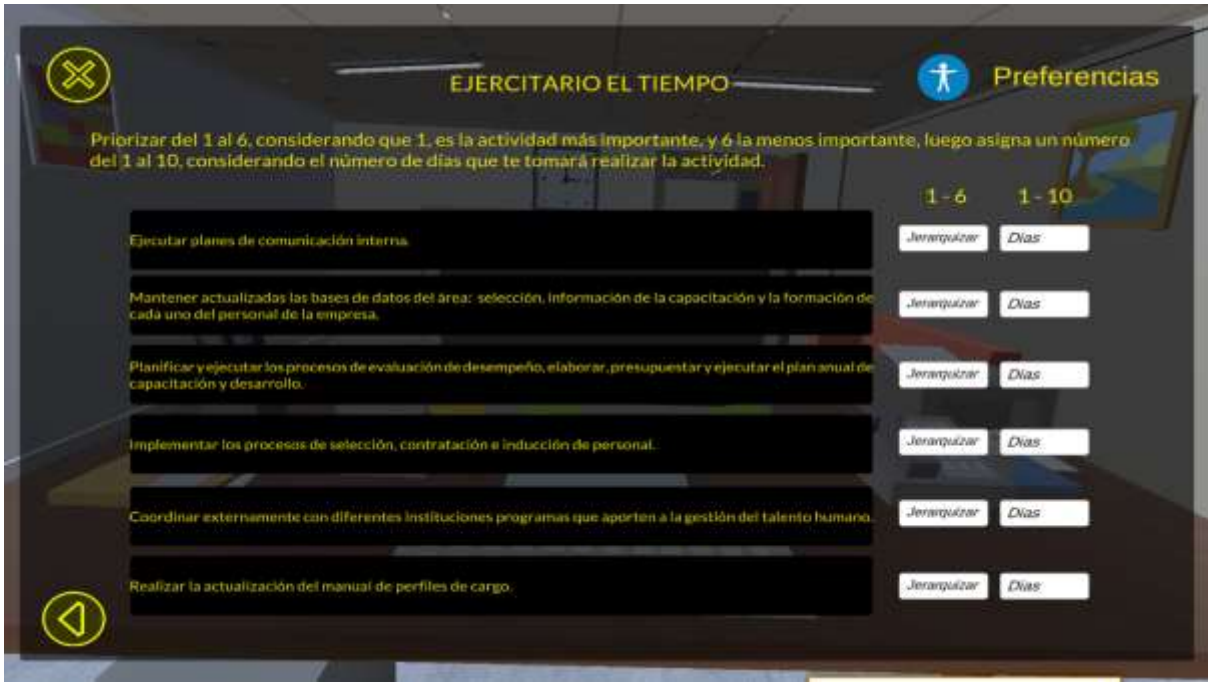


Figura 157. Vista panel en donde se indica las actividades planteadas con las que deberá interactuar el participante.

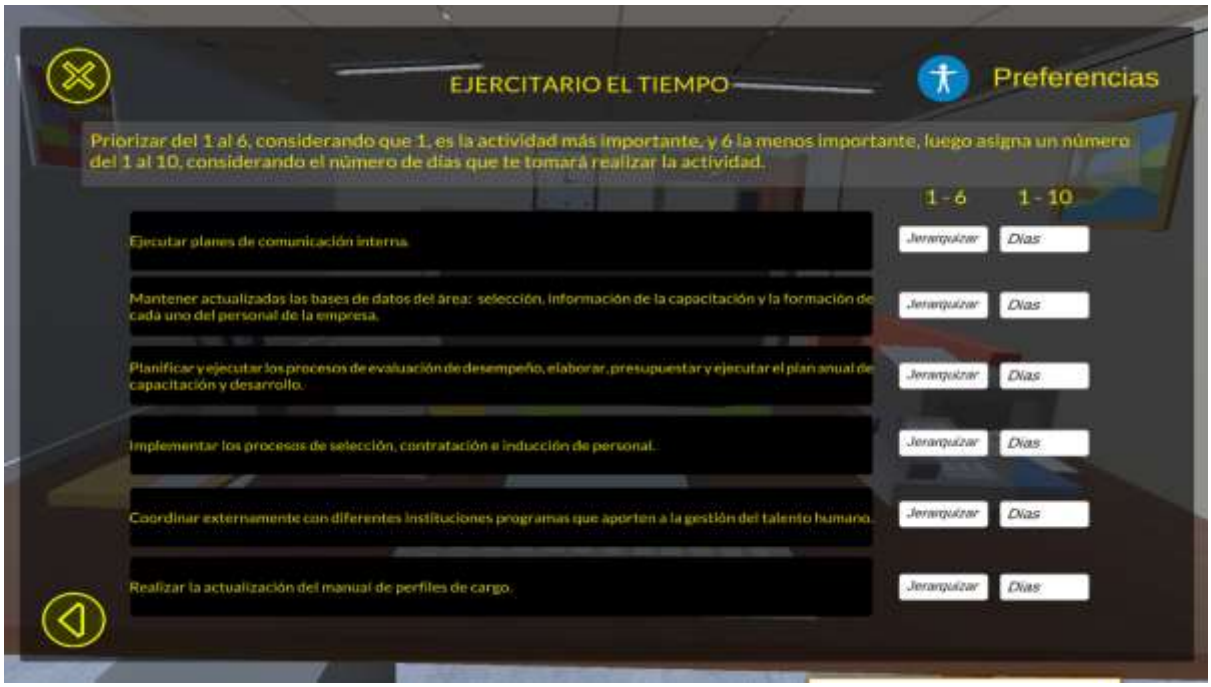


Figura 158. Vista de interacción del participante con panel de actividades mediante el teclado.

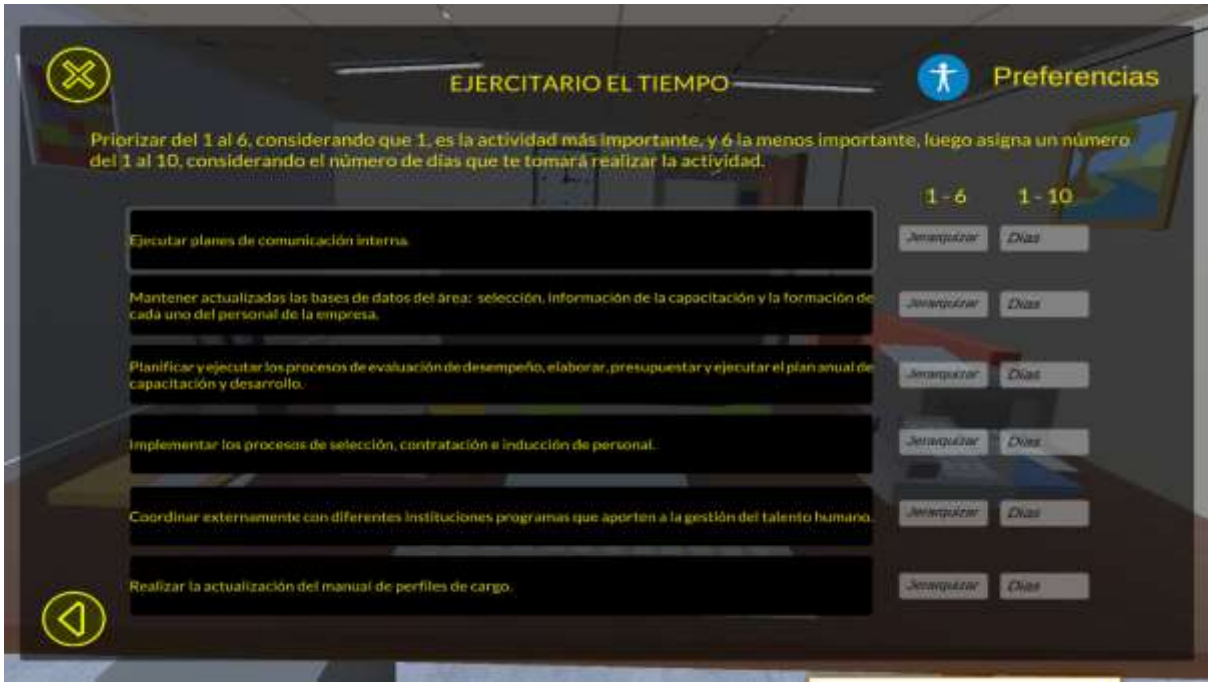


Figura 159. Vista en donde se inactivan los campos de texto de las respuestas al interactuar con el teclado.

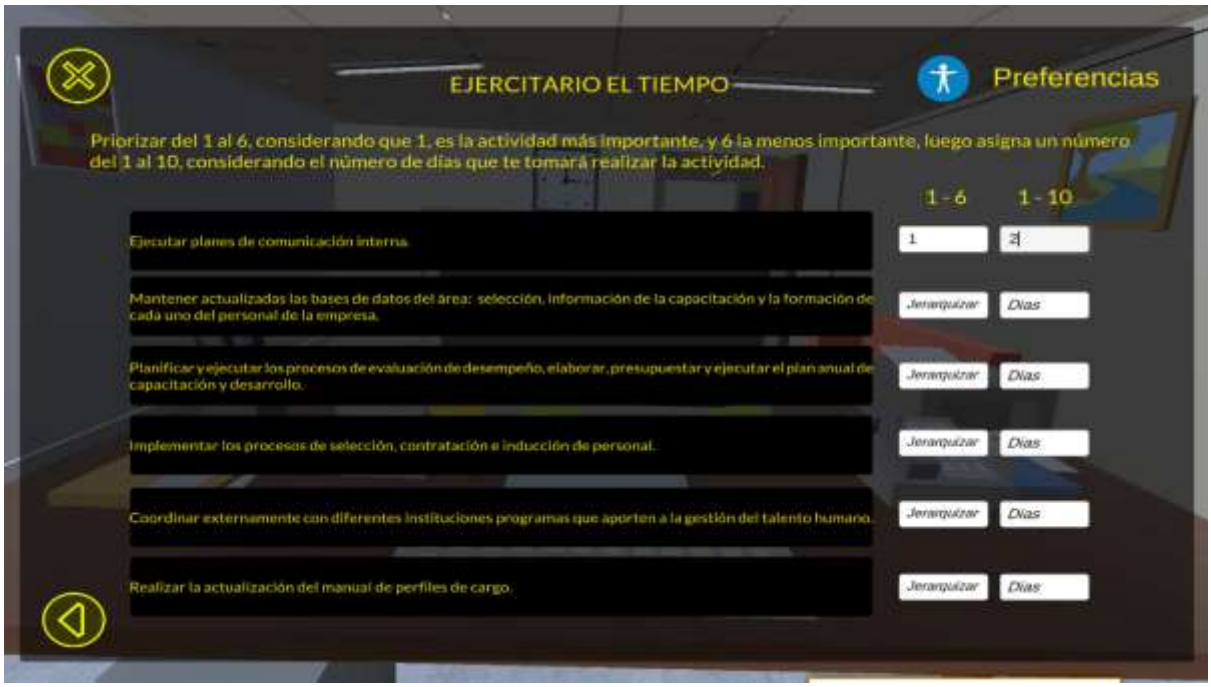


Figura 160. Vista de ingreso de respuestas del participante.

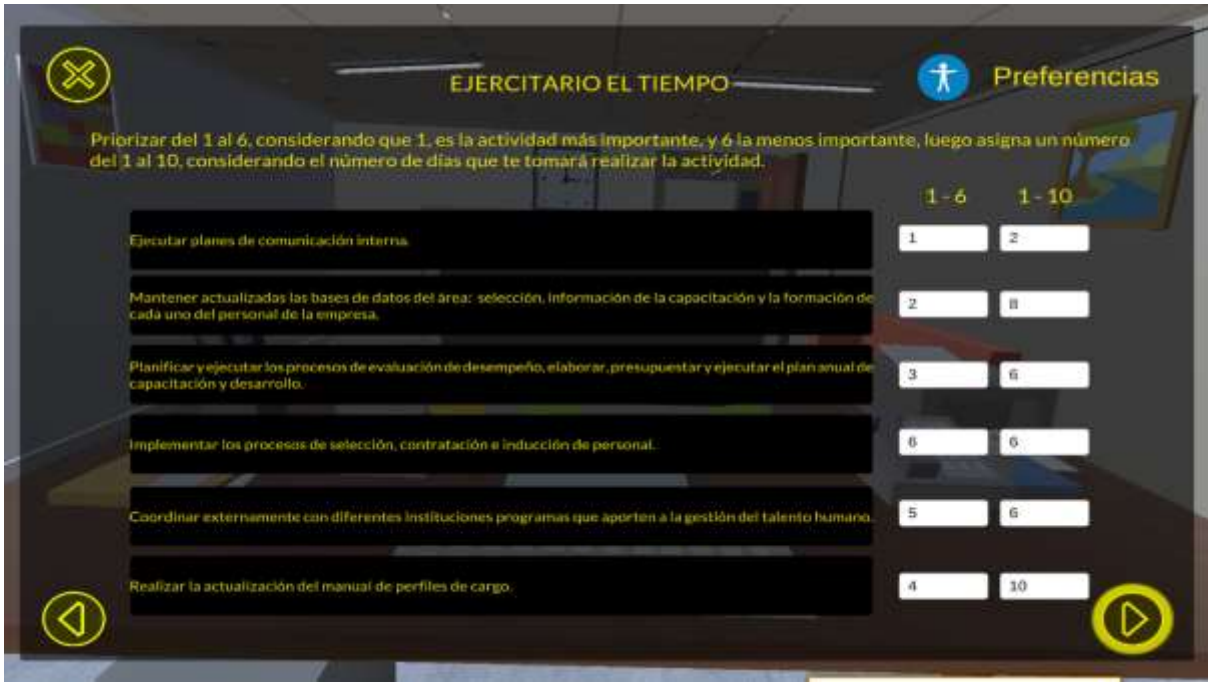


Figura 161. Vista en donde en participante ha completado todas las respuestas y por ello se activa el botón siguiente.

Figura 162. Vista panel de calificación luego de que el participante haya finalizado las actividades planteadas.

Figura 163. Vista panel de retroalimentación.

Figura 164. Vista en donde el participante interactúa con el botón para volver al menú principal.

Interacción del participante con las distintas opciones de accesibilidad del simulador laboral

Figura 165. Vista de participante en la barra de preferencias del simulador laboral mediante el mouse"

Figura 166. Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Arial Bold.

Figura 167. Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Lato.

Figura 168. Vista en donde el participante interactúa con el teclado y selecciona la fuente de texto Dyslexic.

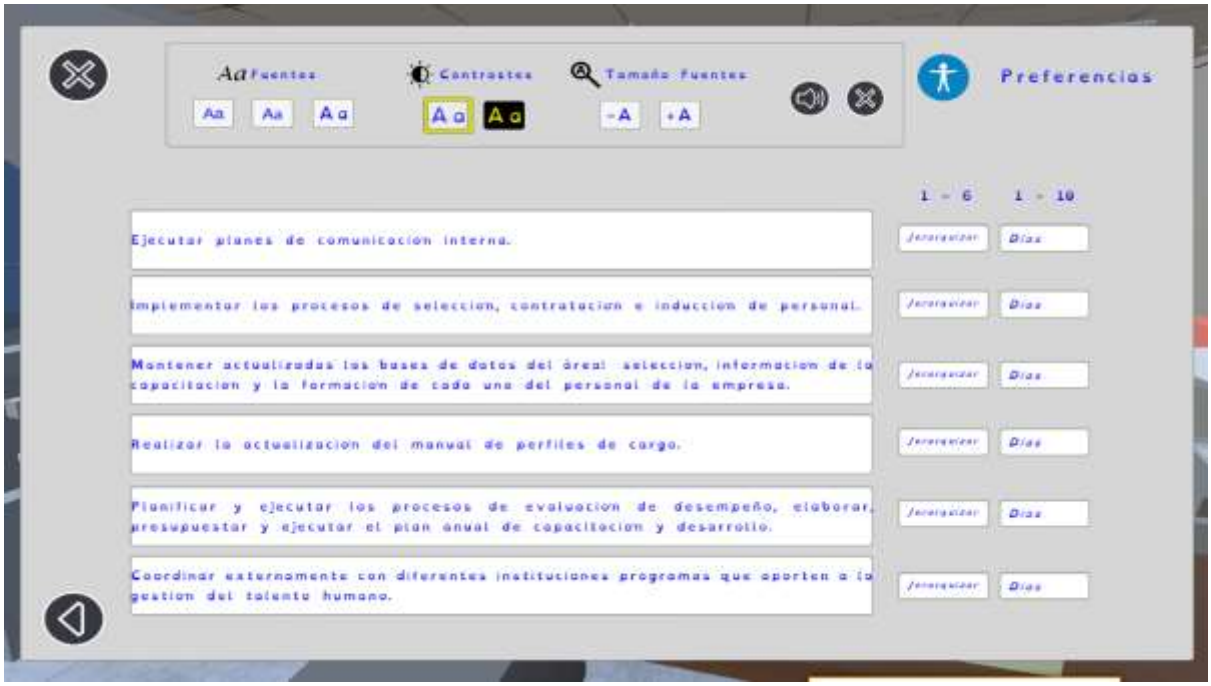


Figura 169. Vista en donde el participante interactúa con el teclado y selecciona un contraste claro.

Figura 170. Vista en donde el participante interactúa con el teclado y selecciona un contraste obscuro.

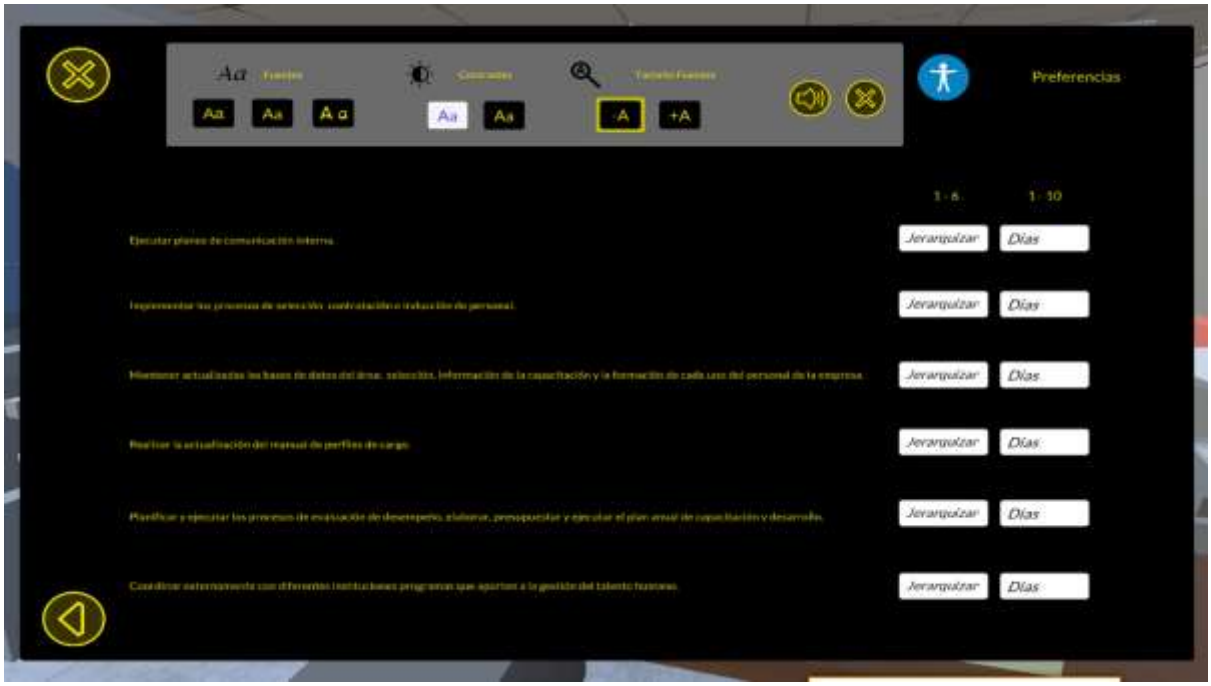


Figura 171. Vista en donde el participante interactúa con el teclado y selecciona el botón de reducir el tamaño de texto.

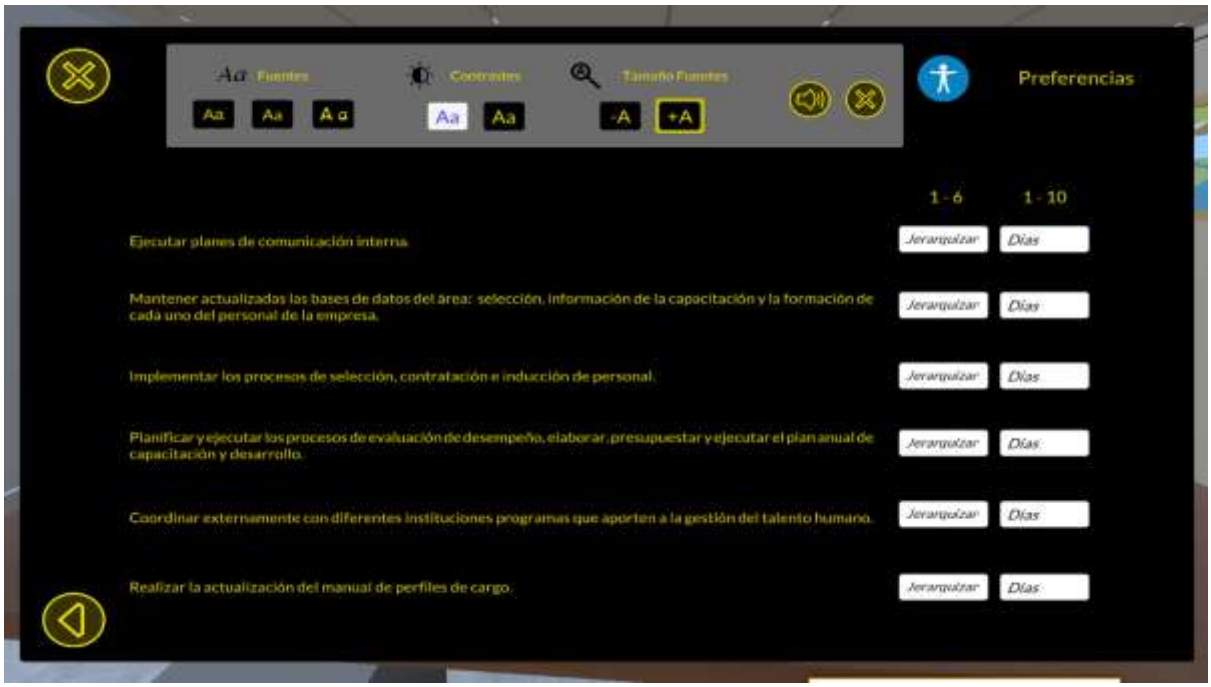


Figura 172. Vista en donde el participante interactúa con el teclado y selecciona el botón de aumentar el tamaño de texto.

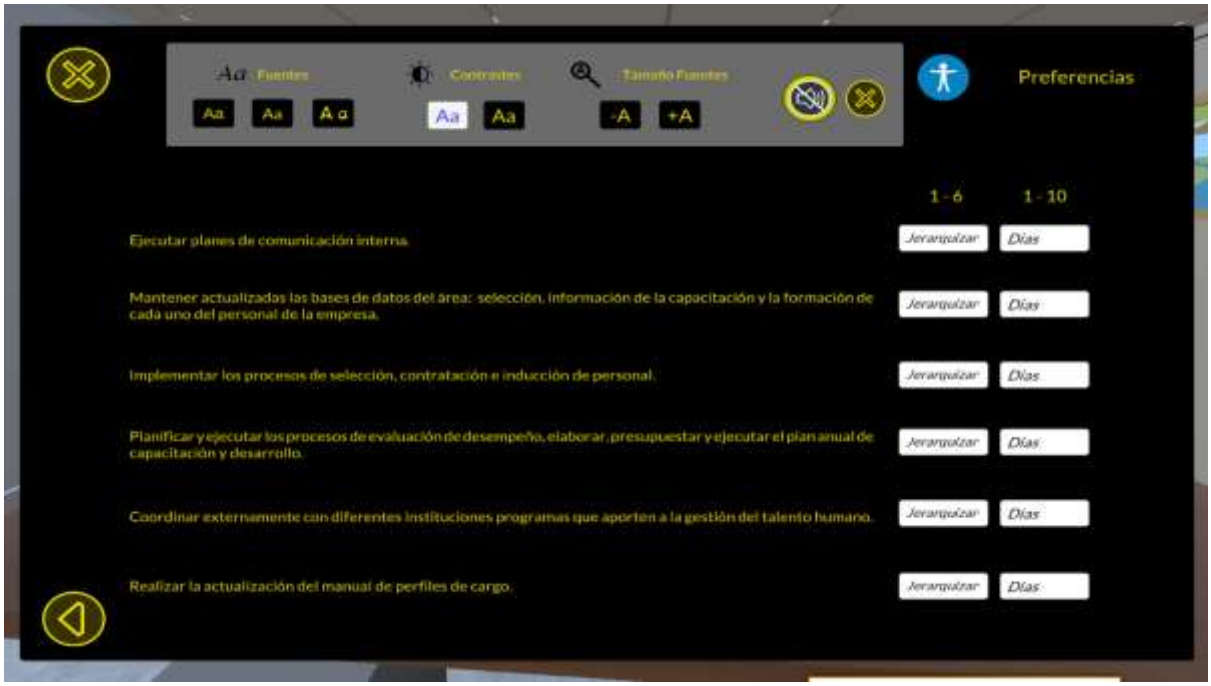


Figura 173. Vista en donde el participante interactúa con el teclado y selecciona el botón desactivar el audio.

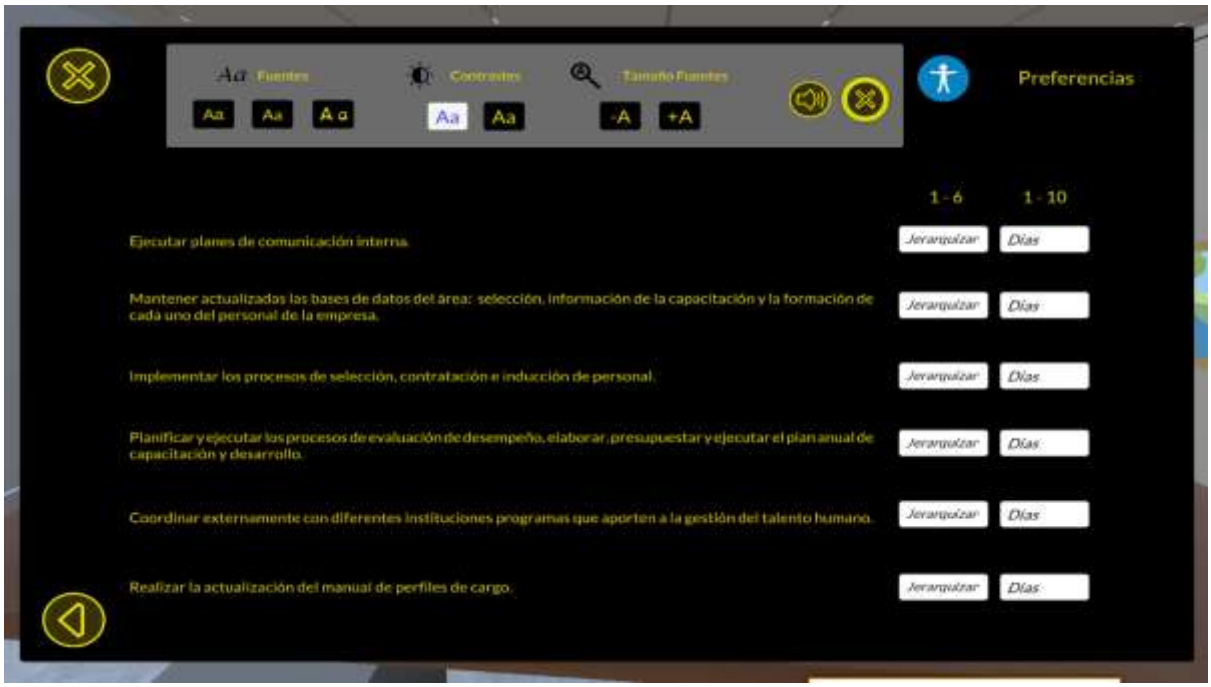


Figura 174. Vista en donde el participante interactúa con el mouse y selecciona el botón salir de la barra de preferencias.

Anexos F

Manual Técnico

“ DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE SIMULADORES LABORALES 3D, BASADO EN EJERCITARIOS EL TIEMPO E INFORMACIÓN DIFÍCIL, DENTRO DEL MARCO DEL PROYECTO EDUTECH”

Manual técnico guiará a los usuarios que darán soporte y modificaciones a los simuladores laborales, el cual les dará a conocer la estructura de la construcción de los simuladores laborales y su funcionalidad.

Est. Narcisa de Jesús Araujo Pérez
naraujop@est.ups.edu.ec

1. Requerimientos del sistema

- Equipo, teclado, mouse, monitor.
- Ram mínimo de 16 GB.
- Sistema operativo Windows 10.
- CPU: compatibilidad con el conjunto de instrucciones SSE2.
- GPU: Tarjeta gráfica con capacidades DX10 (shader model 4.0).
- UNITY: versión 2020.3.21f1.
- WebGL: cualquier versión de escritorio reciente de Firefox, Chrome, Edge o Safari.

2. Funcionamiento de simuladores laborales

2.1 Animación de Personaje

Luego de crear un nuevo proyecto y agregar el personaje a la escena procedemos a crear la respectiva animación, desde nuestro personaje agregamos un componente Animator Controler desde la ventana inspector, luego desde la ventana Project nos dirigimos a la carpeta en donde vamos a crear nuestras animaciones, en este caso nos dirigimos a Assets/Animations una vez dentro damos clic derecho créate/Animator Controler le asignamos un nombre y procedemos a crear nuestra animación.

Para las animaciones del pesonaje se hace uso de varias animaciones descargadas de la página mixamo, dichas animaciones las agregamos a nuestro Animator que creamos *CaminadoInicial*.

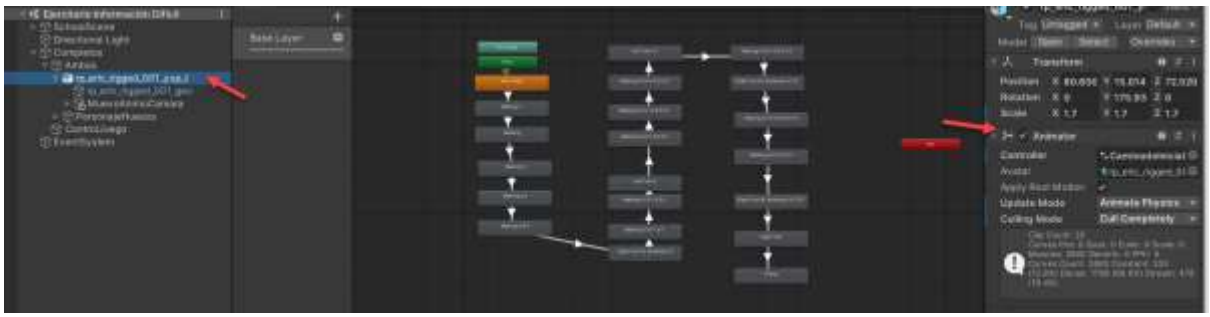


Figura 175. Animación inicial de personaje en Animator Controller de Unity.

Se crea otra animación en donde el personaje cambia de expresión facial al llegar a su lugar de trabajo. Desde el GameObject *Ambos* se crea un *Animator Controller*, creamos un clip

de animación modificamos las propiedades *is Active* del prefab de los huesos del personaje; la propiedad *transform* del GameObject MuevoAnimoCamara en donde se modifica la *posición* y *rotación* de la cámara y la propiedad IsActive del personaje. Dichas propiedades las modificamos en la línea de tiempo según a las necesidades del proyecto.

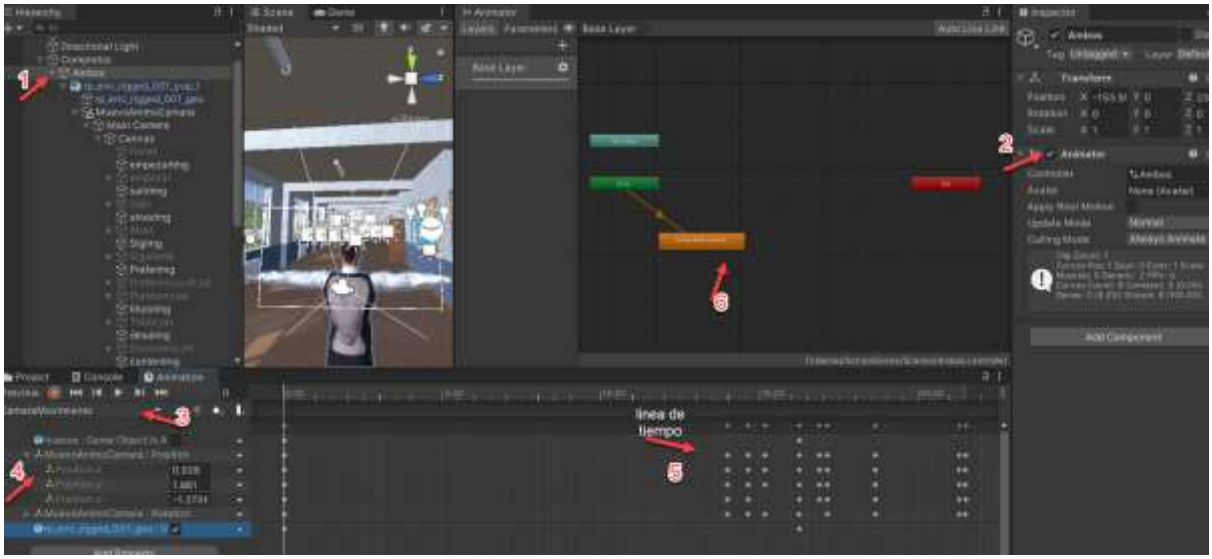


Figura 176. Animación de personaje en la oficina.

2.2 Animación Canvas de Interfaz de usuario

Luego del diseño de la interfaz de usuario se procedió a darle sentido al mismo, se creó una animación para que luego de que termine la animación del personaje aparezca en pantalla el Canvas es decir la interfaz de usuario creada.

A continuación desde nuestro Canvas agregamos un componente Animator Controller llamado *Canvas*, desde la ventana *Animation* creamos una nueva animación damos clic en *create new clip* y se asignamos un nombre a la animación en este caso se llama *Empezar*, luego damos clic en *Add Property* nos aparece un menú del cual seleccionamos el botón a animar y finalmente seleccionamos la propiedad *is Active*. La animación que se hace es modificar la linea

de tiempo de la propiedad *Is Active*, que al ejecutar el proyecto pase 21 segundos(tiempo en que termina la animación del personaje) para que el botón empezar se active.

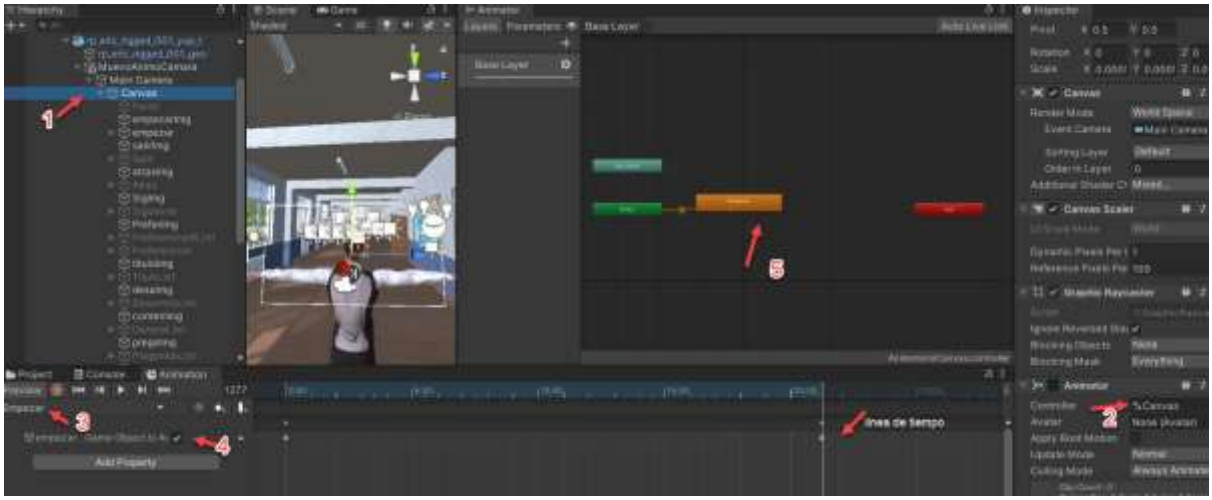


Figura 177. Animación de Canvas usando un Animator Controller.

2.3 Vista de Jerarquía del Proyecto

En la ventana *Inspector* asociamos los componentes creados con las variables declaradas, para ello arrastramos cada componente hacia la variable con la que se desea asociar.

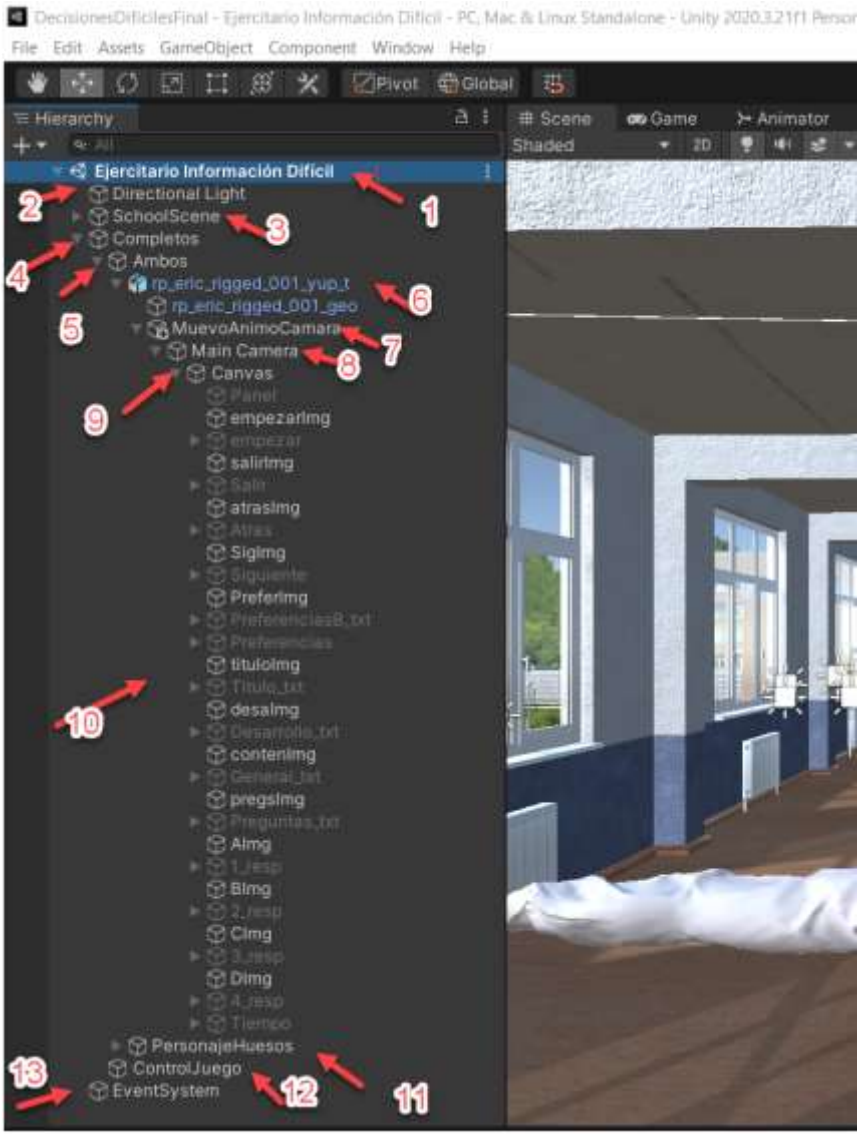


Figura 178. Vista jerarquía del proyecto la cual contiene todos los objetos utilizados.

1. Escena del proyecto.
2. Iluminación de escena.
3. Escenario de simulador laboral.
4. Se crea el GameObject *Completos* y dentro de este se crea dos GameObjects.

5. El GameObject *Ambos* contiene el personaje de la escena, dentro del personaje arrastramos la cámara principal para que esta se mueva con el personaje cuando inicie la animación.
6. Personaje de la escena.
7. GameObject que contiene la cámara principal.
8. Cámara principal de la escena.
9. Se crea un Canvas el cual contiene todos los componentes de la interfaz de usuario.
10. Corresponde a todos los componentes de la interfaz de usuario a simular.
11. Personaje que se muestra después que termina la animación del personaje inicial.
12. Gameobject que contiene el script principal *GeneralEmpezar.cs* y el script *WebData.cs* que consume los servicios web.
13. Es la manera en la que se envían los eventos a objetos basados en input (mouse, teclado).

1.4. Programación Simulador Información Difícil.

1.4.1 Script GeneralEmpezar.cs

```
using System;
using System.Collections;
using System.Collections.Generic;//libreria para manipular con listas
using UnityEngine;
using System.Linq;//libreria para realizar consultas
using System.Text;//libreria para manipular texto
using UnityEngine.UI;// libreria para acceder a las propiedades de los elementos de la interfaz de usuario.
using System.IO;//libreria para escribir y leer datos.
using System.Threading.Tasks;
using UnityEngine.EventSystems;
using TMPro; //importamos la libreria TextMeshPro.
using Random = UnityEngine.Random;
using UnityEngine.Networking;//importamos la libreria para la comunicacion con el servidor web.
```

Figura 179. Librerías utilizadas.

1.2.1.1 Declaración de variables y relación con objetos de interfaz de Unity

Desde la ventana de jerarquía en el GameObject ControlJuego una vez posicionados aquí nos dirigimos a la ventana inspector y procedemos a relacionar los objetos con las variables declaradas para ello arrastramos cada objeto y lo colocamos en la variable correspondiente.

```
// Se declaran las variables publicas a las cuales las asociamos a los gameobjects correspondientes.
public GameObject juego, botSalir, botPref, botSig, botAtras, tit, detall, conten, pregu, botEmpe, personaje, personajeHuesos;
/*para mostrar texto declaramos variables públicas  de tipo TextMeshProUGUI y asociamos dichas variables
con los gameobjects creados en el paso anterior las cuales contienen los componentes <<text>>*/
public TextMeshProUGUI textoEmpiezo,textoTit,textPreg, textodetall, textoconten, tiempoText, textoSig;
//Se declara las variables públicas de tipo Image las cuales las asociamos al botón empezar y al botón siguiente.
public Image empiezo, nextB;
```

Figura 180. Fragmento de código de declaración de variables.

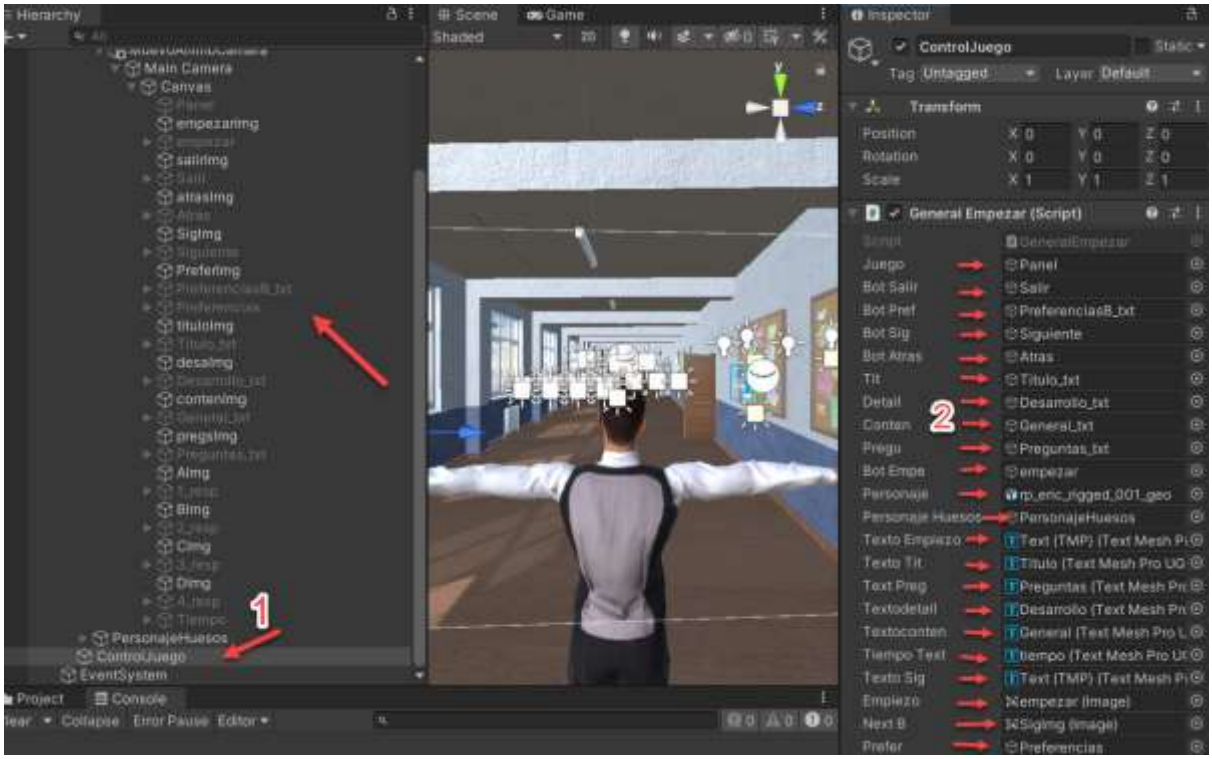


Figura 181. Relacionamos o arrastramos cada objeto con la variable correspondiente.

```
//Se declara las variables públicas y las asociamos con los gameobjects correspondientes, dichas variables
// se utilizan para dar animaciones a objetos del escenario.
public GameObject prefer, fondoPC, papeles, papeles1, calcu, mouse;
```

Figura 182. Fragmento de código de declaración de variables.

Figura 183. Relacionamos o arrastramos cada objeto con la variable correspondiente.

```
//Se declara la variable estática booleana para controlar cuando la barra de preferencias este activada o desactivada.
public static bool prefBool = false;
/*Se declara la variable de tipo string, en la cual cargamos las preguntas con sus respectivas opciones de respuestas;
las preguntas son separadas mediante " ;; " y las opciones de respuestas las separamos mediante " ||| "*/
string documento = "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará.;;1.- Análisis de ev
//Variable en la que se cargara una a una las líneas de texto que estoy separando y así controlar que no este vacía.
string line;
// array en donde se guardaran los tiempos de respuesta utilizado por el participante.
string[] tiemposResp;
// la variable pos se utiliza como contador de las actividades y utilizamos la variable contRes como contador de las respuestas correctas.
int pos = 0, contRes = 0;
/* Se declara las siguientes listas de string en donde:
    contenPreg, alternaPregs, contenPregFinal, alternaPregsFinal
*/
List<string> contenPreg, alternaPregs, contenPregFinal, alternaPregsFinal;
```

Figura 184. Fragmento de código de declaración de variables.

```
//Se declara un array de gameobjects en donde se cargarán los gameobjects que contienen las alternativas a elegir por el participante.
public GameObject[] altern;
//Se declara un array de tipo texto el cual contiene el texto correspondiente a cada alternativa a elegir.
public TextMeshProUGUI[] textalter;
```

Figura 185. Fragmento de código de declaración de variables.

El array *altern* hace referencia a los botones que contienen las alternativas de respuestas, el array *textalter* corresponde a espacio en donde se cargará el texto de las alternativas; ya que cada pregunta cuenta con cuatro alternativas se define un tamaño de 4.

Figura 186. Relacionamos los objetos que contienen las alternativas de la pregunta.

```
//Se declara array de string en donde cargamos las respuestas correctas.
public string[] rescor;
```

Figura 187. Fragmento de código de declaración de variables.

A continuación, le damos un tamaño de 2 al array ya en este caso son dos preguntas, especificamos las respuestas correctas de cada pregunta, en este caso las respuestas correctas de las preguntas del ejercitario Información difícil son la respuesta 1.

Figura 188. Se registra las respuestas correctas de las actividades.

```
    //variable que utilizamos para mostrar texto en pantalla.
    string retFin;
    int puntajeFinal = 0;
    private float StartTime;
    //variable es utilizada para controlar en que panel o menú nos encontramos.
    public static string menu = "Main";
```

Figura 189. Fragmento de código de declaración de variables.

Para la navegación mediante el teclado se crearon arreglos seleccionables, arreglos de imágenes y de audios, por tanto, es importante recalcar que el orden en que se cargue los objetos en cada arreglo será el orden que se seleccione al presionar el teclado, es decir si al presionar el teclado se activa el botón en la posición 0 entonces también se activará la imagen y sonido en la posición 0.

```
/* Array de objetos seleccionables, cada objeto que se va seleccionar con el teclado.
    // nextFieldMain: botones panel principal
    // nextFieldPrefer: botones panel de preferencias
    // nextFieldDiag: botones panel de instrucciones
    // nextFieldPreg: botones panel de preguntas
    // nextFieldFinA: botones panel de puntuación
    // nextFieldFin: botones panel de retroalimentación
 */
public Selectable[] nextFieldMain, nextFieldPrefer, nextFieldDiag, nextFieldPreg, nextFieldFin, nextFieldFinA;
/* Array de imágenes correspondientes a cada boton del juego.
    // main: imagenes panel principal
    // prefImg: imagenes panel de preferencias
    // dial: imagenes panel de instrucciones
    // juegI: imagenes panel de preguntas
    // finI: imagenes panel de retroalimentación
    // alternaIma: imagenes de las alternativas a elegir
    // finAI: imagenes panel de puntuación */
public Image[] main, prefImg, dial, juegI, finI, alternaIma, finAI;
```

Figura 190. Fragmento de código de declaración de variables.

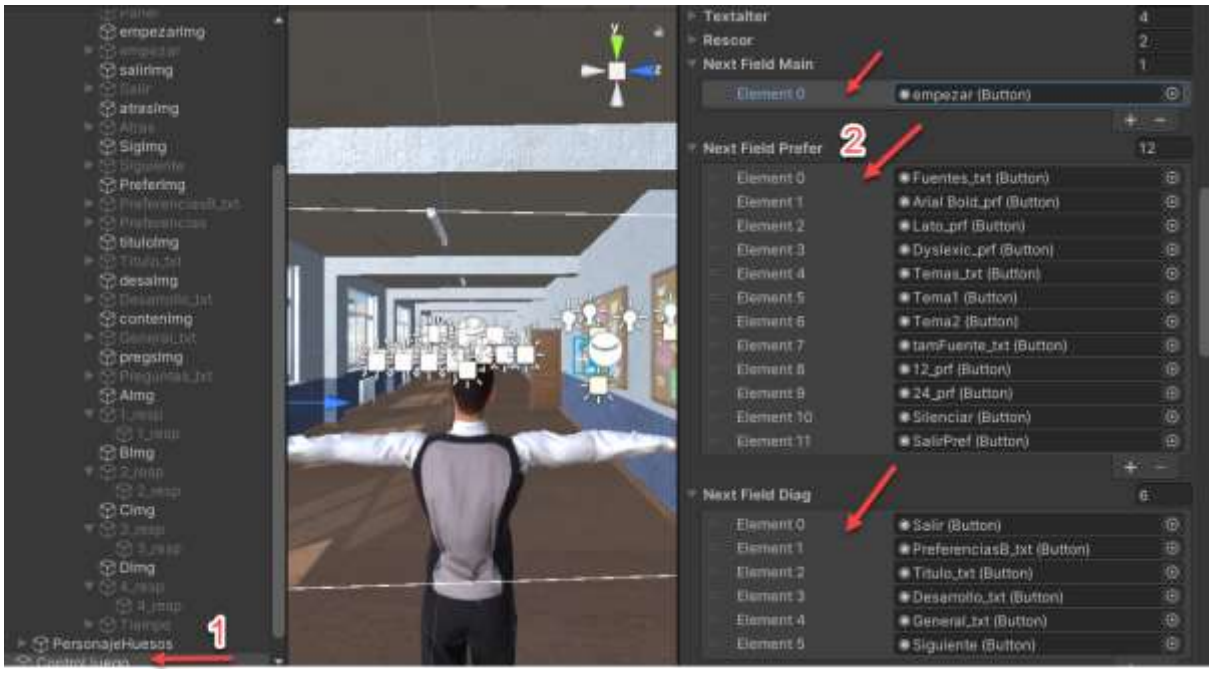


Figura 191. Relacionamos los objetos con la variable correspondiente.

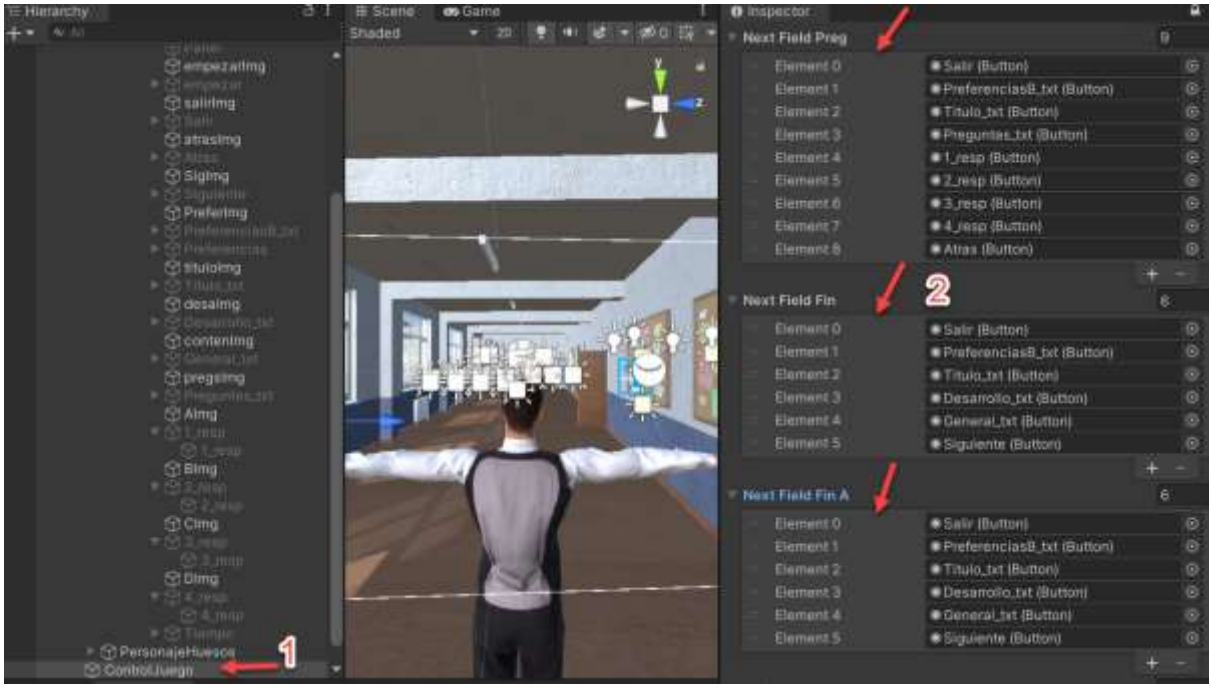


Figura 192. Relacionamos los objetos con la variable correspondiente.

```
/* Array de imágenes correspondientes a cada boton del juego.
    // main: imagenes panel principal
    // prefImg: imagenes panel de preferencias
    // dial: imagenes panel de instrucciones
    // juegI: imagenes panel de preguntas
    // finI: imagenes panel de retroalimentación
    // alternaIma: imagenes de las alternativas a elegir
    // finAI: imagenes panel de puntuación */
public Image[] main, prefImg, dial, juegI, finI, alternaIma, finAI;
```

Figura 193. Fragmento de código de declaración de variables.

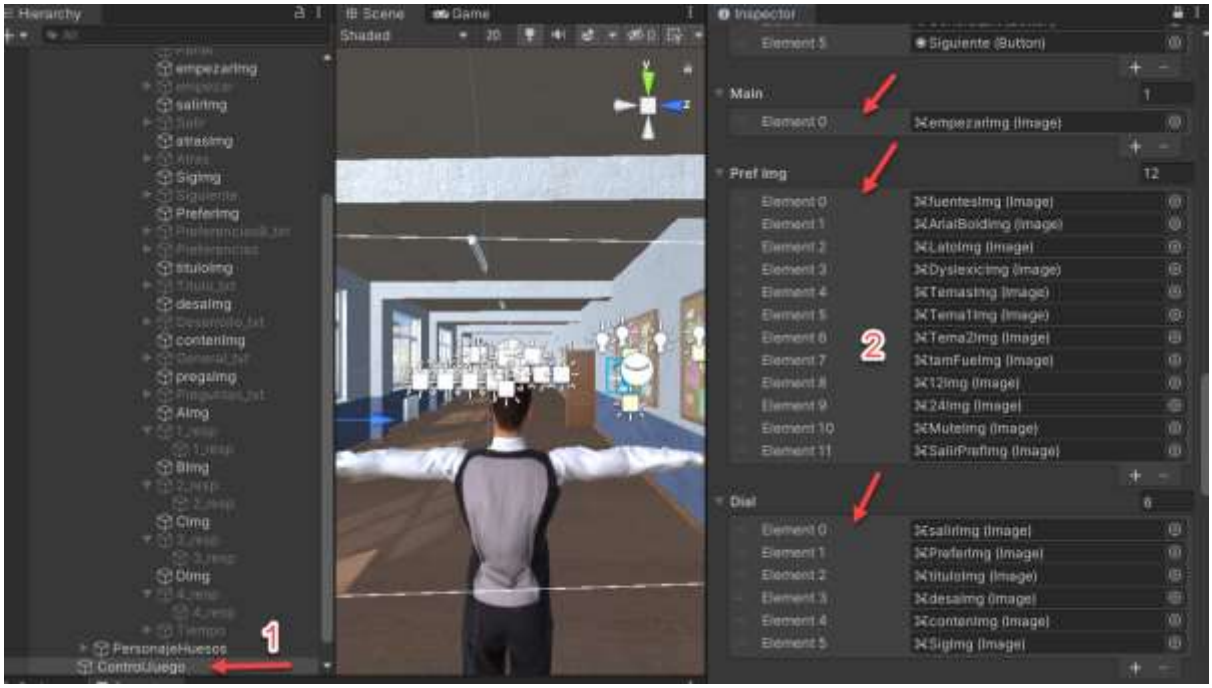


Figura 194. Relacionamos los objetos con la variable correspondiente.

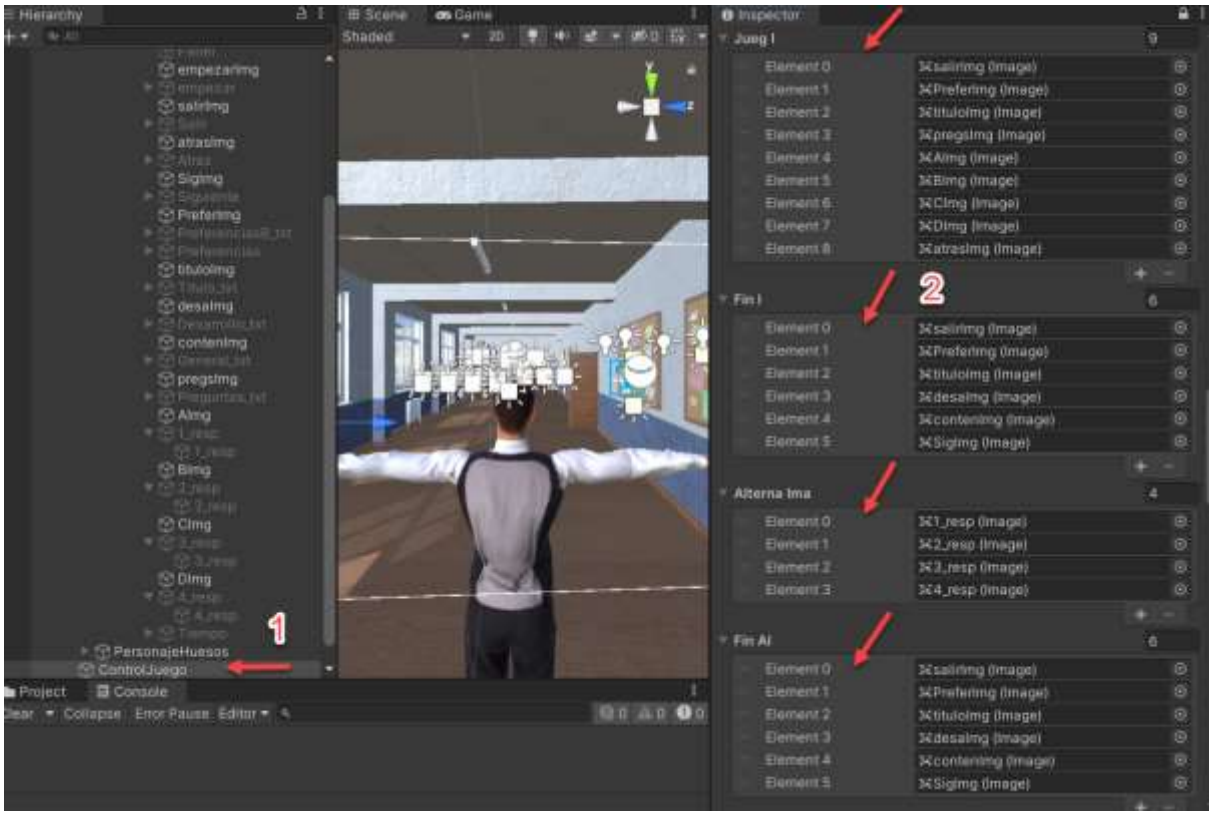


Figura 195. Relacionamos los objetos con la variable correspondiente.

Es importante resaltar que el arreglo de objetos seleccionables y el arreglo de imágenes son los que van a funcionar con el teclado, por ejemplo, si al presionar el teclado nos posicionamos en la tecla salir se activará también la imagen como se muestra a continuación.

Figura 196. Ejemplo de selección de botón salir acompañado de una imagen que lo resalta.

A continuación, se declaran contadores para recorrer los arreglos de botones, audios, imágenes utilizadas para la navegación por teclado.

```
/* numC: Controlador para saber en que boton del panel de instrucciones estamos
   numI: Controlador para saber en que IMAGEN estamos */
int numC = 0, numI = 0;
/* numCP: Controlador para saber en que boton del panel de preferencias estamos
   numFP: Controlador para saber en que imagen del panel de preferencias estamos */
int numCP = 0, numFP = 0;
/* numPP: Controlador para saber en que pregunta estamos
   numFPP: Controlador para saber en que imagen de la pregunta estamos   */
int numPP = 0, numFPP = 0;
/* numFF: Controlador para saber en que boton del panel de puntuacion estamos
   numIFF: Controlador para saber en que imagen del panel de puntuacion estamos */
int numFF = 0, numIFF = 0;
/* numFFA: Controlador para saber en que boton del panel de retroalimentación estamos
   numIFFA: Controlador para saber en que imagen del panel de retroalimentación estamos  */
int numFFA = 0, numIFFA = 0;
/* variables para recorrer en que pregunta estamos
    // ap1: alternativas de la pregunta 1
    // ap2: alternativas de la pregunta 2   */
int ap1 = 0, ap2 = 0;
/* variable que controla en que numero de pregunta estoy*/
int numPreg;
```

Figura 197. Fragmento de código de declaración de variables.

Se declara una variable de tipo AudioSource , esta será quien reproduzca todos los audios.

```
// variable de tipo AudioSource que representa el sonido principal, que es donde va a salir todos los sonidos al interactuar con el teclado
public AudioSource sonidoAS;
```

Figura 198. Fragmento de código de declaración de variables.

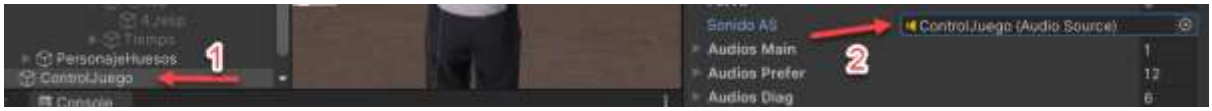


Figura 199. Relacionamos el objeto con la variable correspondiente.

```
/* Array de audios, aqui uno a uno los vamos que voy a cargar  */
public AudioClip[] audiosMain, audiosPrefer, audiosDiag, audiosPreg, audiosP1, audiosP2, audiosFin, audiosFinA;
```

Figura 200. Fragmento de código en donde se declaran varios arreglos de audios.

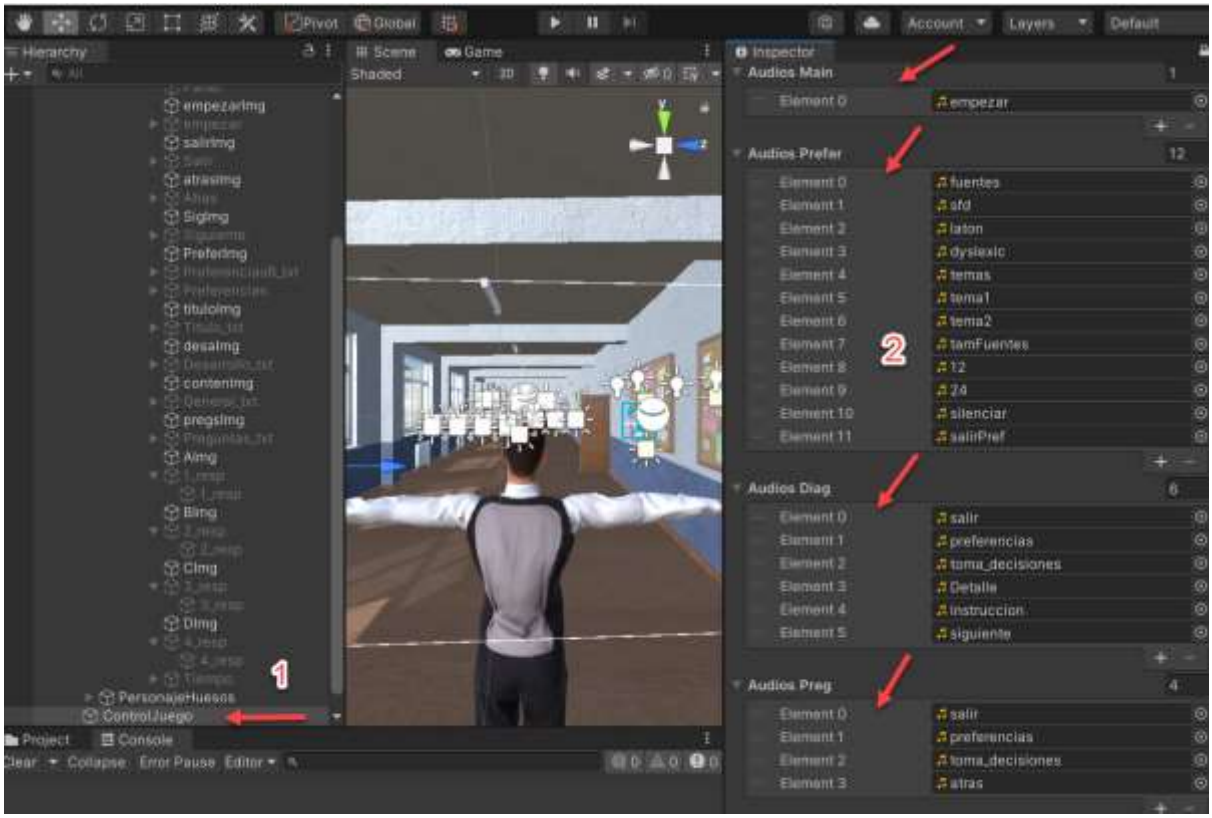


Figura 201. Relacionamos los objetos con la variable correspondiente.

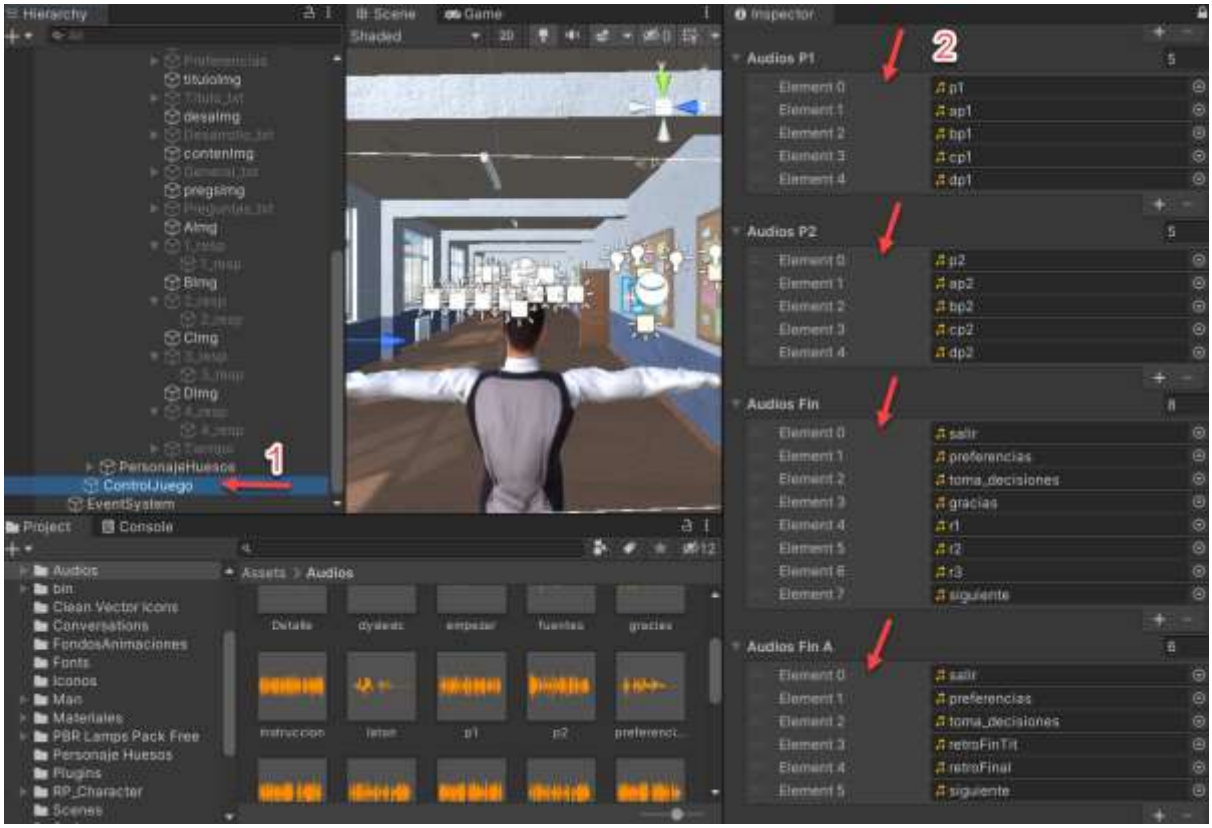


Figura 202. Relacionamos los objetos con la variable correspondiente.

A continuación, se declaran arreglos de tipo AudioSource los cuales guardarán todos los audios a reproducir mediante el mouse.

```
/* sive para detener cualquier sonido que esté dentro de un objeto cuando se presione el ratón. */
public AudioSource[] audiosMainMouse, audiosPreferMouse, audiosDiagMouse, audiosPregMouse, audiosFinMouse, audiosFinAMouse;
```

Figura 203. Se declaran arreglos de GameObjects.

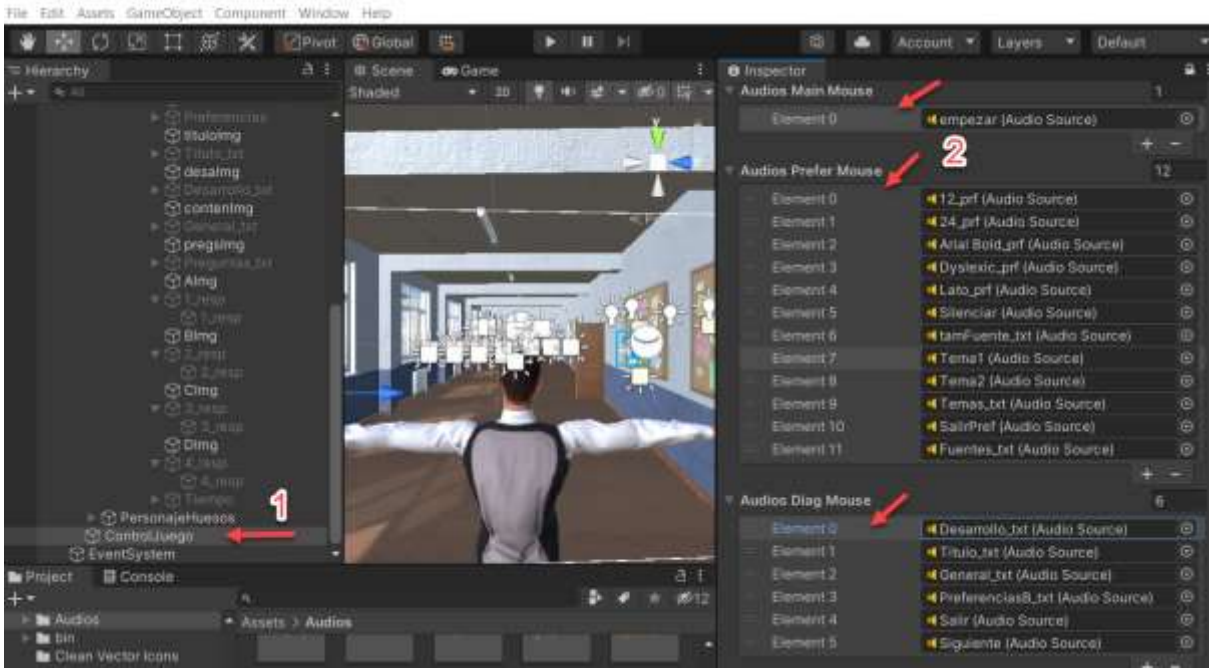


Figura 204. Relacionamos los objetos con la variable correspondiente.

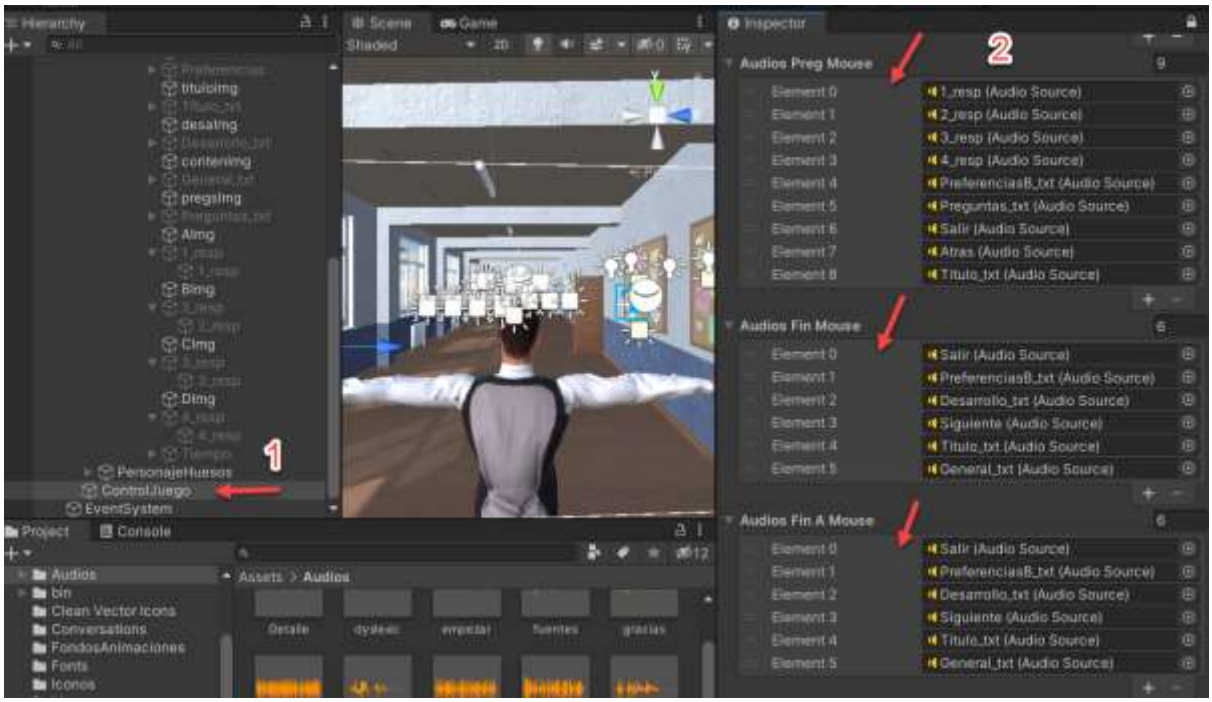


Figura 205. Relacionamos los objetos con la variable correspondiente.

```
/* variable que será true o false cuando se mueva la cámara */
public static bool move = true;
/* variable cámara de tipo gameobject a la cual se le relaciona la cámara principal de la escena.*/
public GameObject camara;
//Lista de tipo entero en donde se guardará los números aleatorios
List<int> numerosGuardados;
/* lista en donde se guardan las preguntas. */
List<int> numPares;
/* lista de enteros, en donde se guardan las alternativas. */
List<int> listaFinalPregs;
// variable que se usa de contador de las preguntas.
int contadorPR;
/* variable que obtiene la fecha actual. */
string datetime;
```

Figura 206. Fragmento de código de declaración de variables.

```
/* boton que se usa para indicar que se ha llegado al final del juego, a esta variable la asociamos al boton siguiente */
public Button sigoFin;
/* variables de tipo Sprite
    // ImagenCasa: imagen para reiniciar juego
    // ImagenSiguiente: imagen para boton siguiente
    */
public Sprite ImagenCasa, ImagenSiguiente;
```

Figura 207. Fragmento de código de declaración de variables.

Figura 208. Relacionamos los objetos con la variable correspondiente.

```
/* declaro el array de preguntas. */
string arrayPreguntas;
 /* creación de una variable de la clase WebData,
 a la cual se pasa los campos de datos para consumir los servicios web,
 luego relacionamos esta variable con el gameobject que contiene el script WebData.cs. */
public WebData webData;
/* instancia de la clase SimuladorData */
public SimuladorData eje = new SimuladorData();
/* variable de la clase Pregunta, esta variable irá agregando dentro del JSON una nueva pregunta*/
public Pregunta ppre;
// variable que es utilizada para controlar el estado de las preguntas.
string TextoAgregoModif = "";
```

Figura 209. Fragmento de código de declaración de variables.

1.2.1.2 Declaración de funciones

A continuación, se muestra el código fuente utilizado con su respetiva documentación.

```
void Start()
{
    nuevoJuego();//llamamos a la función para que antes que nada los valores empiecen en 0
}
```

Figura 210. Fragmento de código de la función Start.

```
void Update()
{
    //si menu es igual a Preguntas
    if ( GeneralEmpezar.menu == "Preguntas")
    {
        correTiempo();//empieza a tomar el tiempo
    }
    //si el botón empezar esta activo y el movimiento de la cámara esta activo
    if (botEmpe.activeSelf == true && move == true)
    {
        /*podremos mover la vista de la cámara mediante el mouse*/
        float pointer_x = Input.GetAxis("Mouse X");
        float pointer_y = Input.GetAxis("Mouse Y");
        camara.transform.Rotate(0, pointer_x * 1f, 0);
    }
    if (botEmpe.activeSelf == true)//si el botón empezar esta activo
    {
        /*se desactiva los personajes*/
        personaje.SetActive(false);
        personajeHuesos.SetActive(false);
    }
    Tabulando();//llamamos a la función tabulando, para que siempre este escuchando a los eventos del teclado
}
```

Figura 211. Fragmento de código de la función Update() en donde se implementa el movimiento de la cámara con el mouse.

```
//esta función permite salir del simulador laboral
public void Salir()
{
    Application.Quit();
}
```

Figura 212. Fragmento de código de la función Salir.

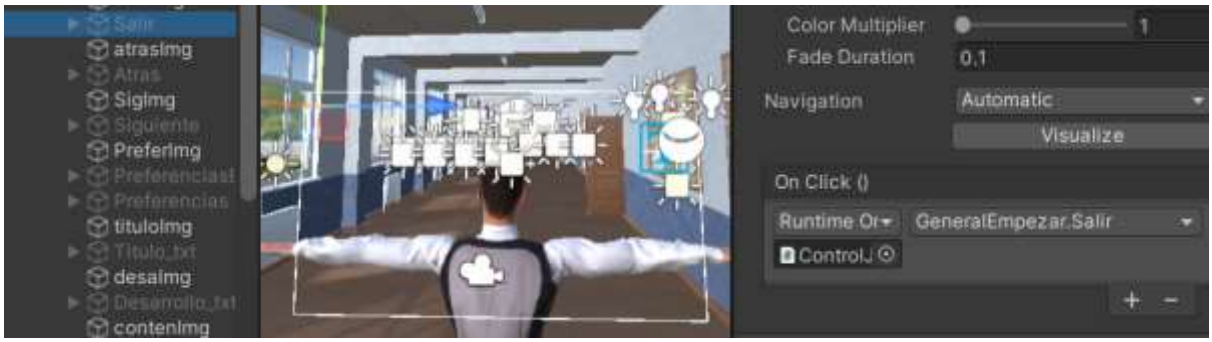


Figura 213. Agregamos el evento On Click y llamamos a la función Salir.

La función *nuevoJuego()* permite al estudiante volver a la interfaz principal en donde podrá iniciar de nuevo la interacción con el simulador laboral, inicializando todos los valores de las variables en cero.

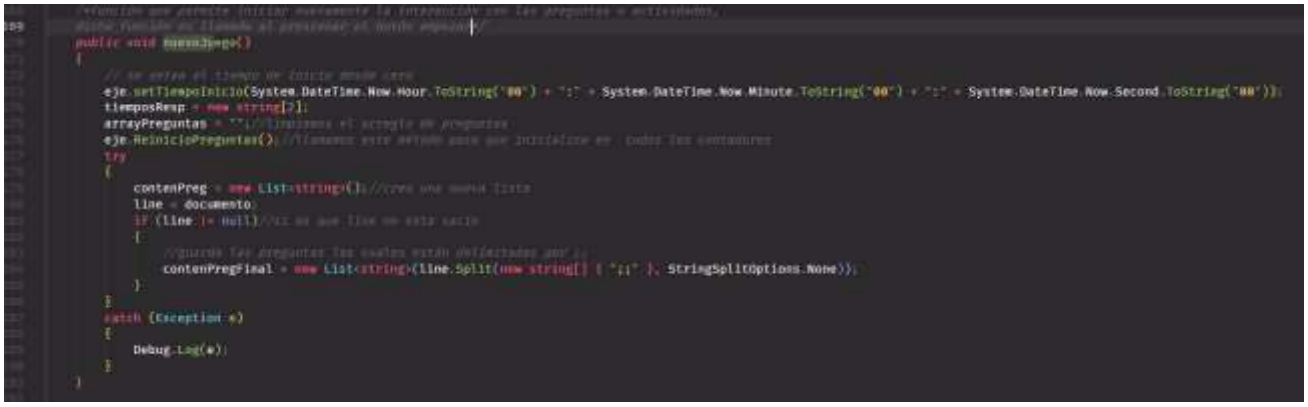


Figura 214. Fragmento de código de la función nuevoJuego.

La función *Tabulando ()* principalmente se hace cargo de la navegación mediante el teclado en donde para navegar se ha definido las teclas: navegación hacia atrás(alt, fecha izquierda, flecha arriba) y para navegar hacia adelante(tab, flecha derecha y flecha abajo).

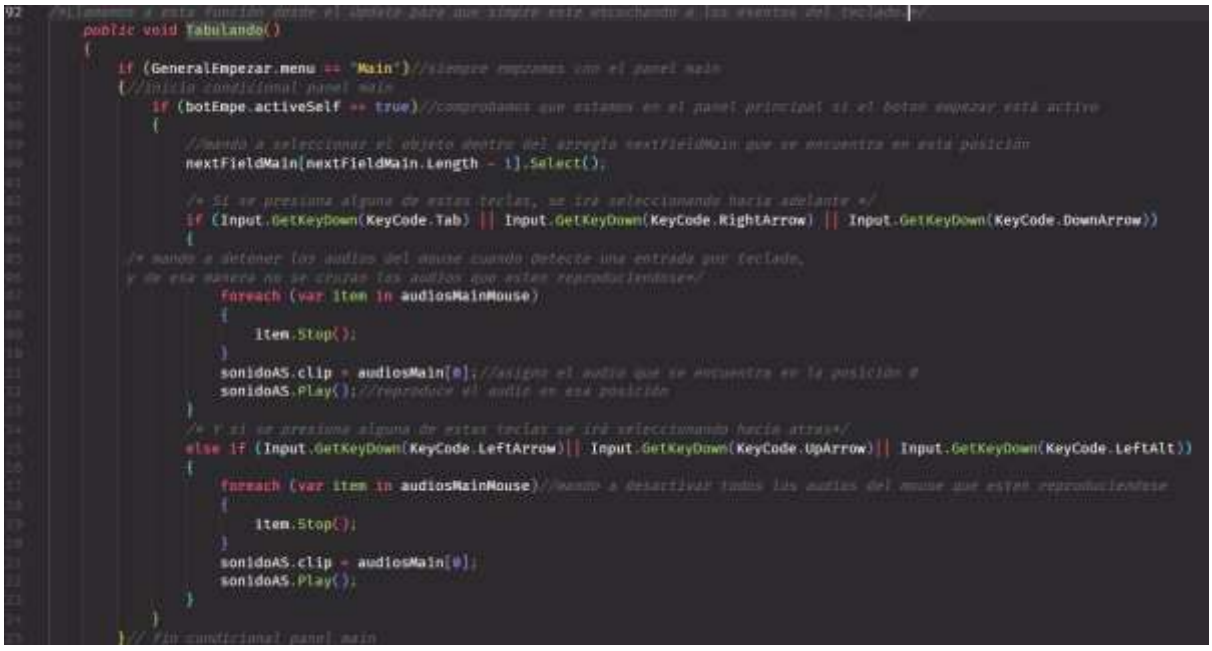


Figura 215. Fragmento de código de la función Tabulando parte 1.

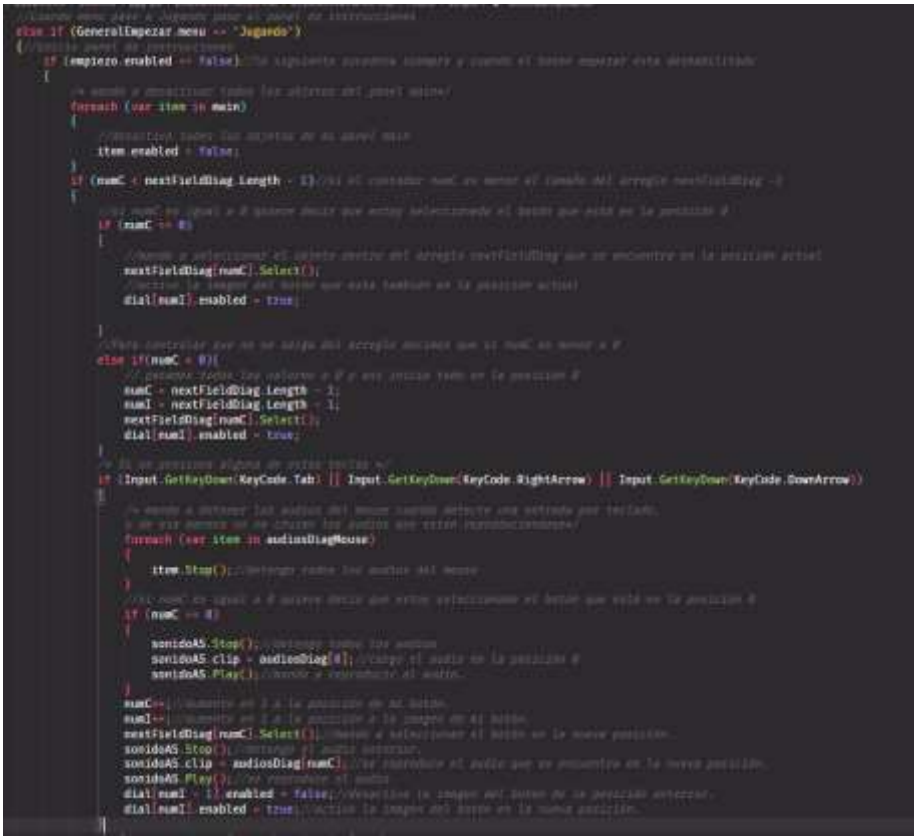


Figura 216. Fragmento de código de la función Tabulando parte 2.

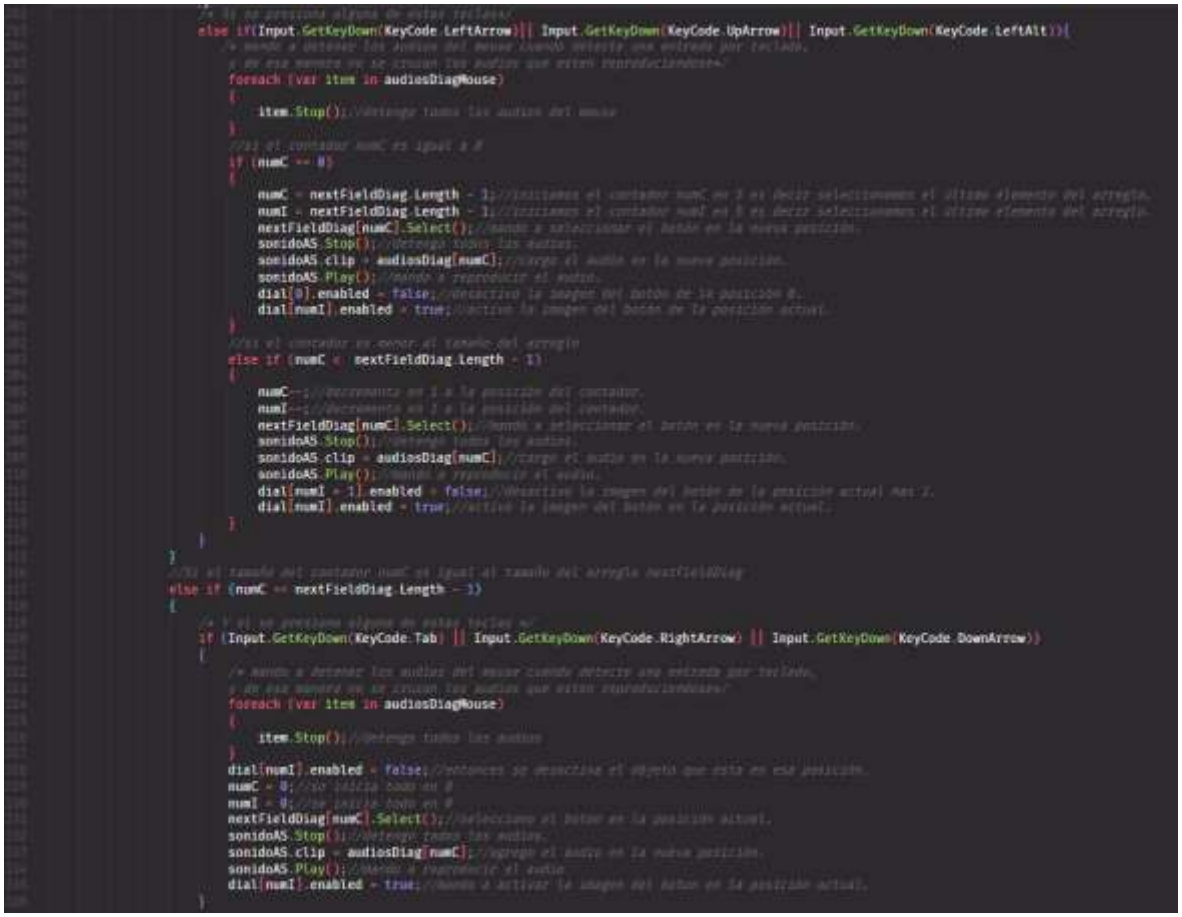


Figura 217. Fragmento de código de la función Tabulando parte 3.

```
                /* Y si se presiona alguna de estas teclas */
                else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){

                    numC--;//decremento en 1 a la posición del contador.
                    numI--;//decremento en 1 a la posición del contador.

                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                    y de esa manera no se cruzan los audios que esten reproduciendose*/
                    foreach (var item in audiosDiagMouse)
                    {
                        item.Stop();//detengo todos los audios.
                    }
                    dial[numI+1].enabled = false;//desactivo la imagen del botón de la posición actual más 1.
                    nextFieldDiag[numC].Select();//selecciono el boton en la posición actual.
                    sonidoAS.Stop();//detengo todos los audios que esten reproduciendose.
                    sonidoAS.clip = audiosDiag[numC];//cargo el audio en la nueva posición.
                    sonidoAS.Play();//mando a reproducir el audio.
                    dial[numI].enabled = true;//activo la imagen del botón de la posición actual.
                }
            }
        }
        //si el botón siguiente está desactivado significa que llegué al panel de preguntas
        if (botSig.activeSelf == false)
        {
            //al método MenuSel el cual se utiliza para controlar en que menu nos encontramos le paso el nombre Preguntas.
            GeneralEmpezar.MenuSel("Preguntas");
            /* mando a desactivar todos los objetos del panel de instrucciones*/
            foreach (var item in dial)
            {
                item.enabled = false;//desactivo todos los objetos.

            }
            numC = 0;//inicializo el contador en 0.
            numI = 0;//inicializo el contador en 0.
        }

    }//fin condicional panel de instrucciones

    //si menu es igual a preguntas
    else if (GeneralEmpezar.menu == "Preguntas")
    {//inicio condicional panel de preguntas
        //si el botón siguiente está desactivado
        if (botSig.activeSelf == false)
        {
            nextB.enabled = false;//desactivo la imagen del boton siguiente.

            //Si el contador numPP es menor al tamaño del arreglo nextFieldPreg.
            if (numPP < nextFieldPreg.Length - 1)
            {
                //Y si el contador es igual a 0
                if (numPP == 0)
                {

                    nextFieldPreg[numPP].Select();//selecciona el botón en la posición actual.
                    juegI[numFPP].enabled = true;//activa la imagen del botón en la posición actual.
                }
```

Figura 218. Fragmento de código de la función Tabulando parte 4.

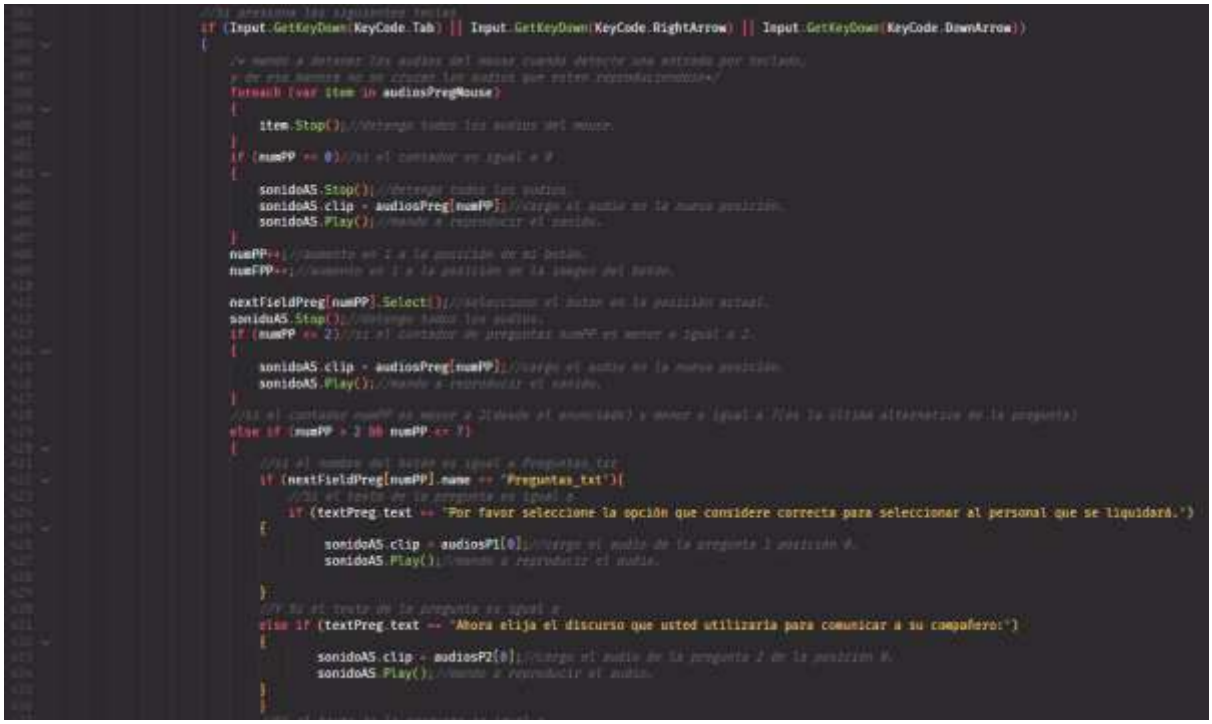


Figura 219. Fragmento de código de la función Tabulando parte 5.

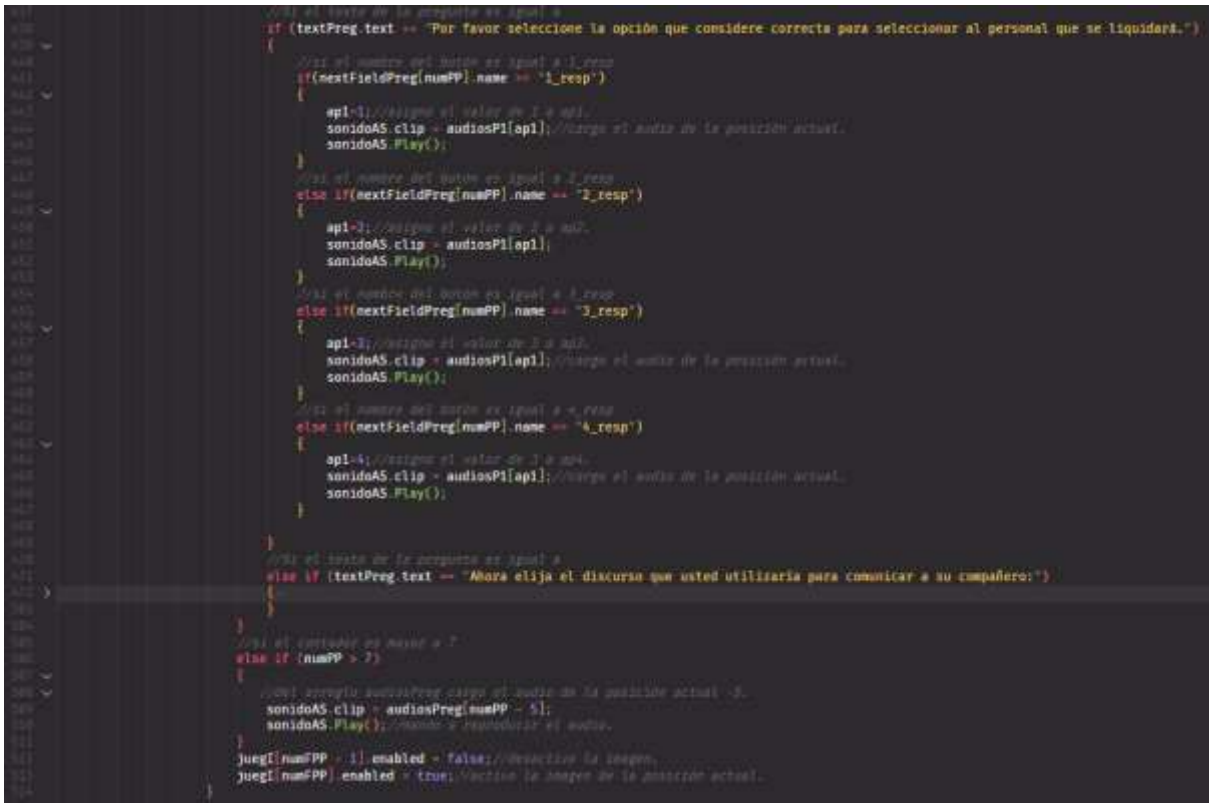
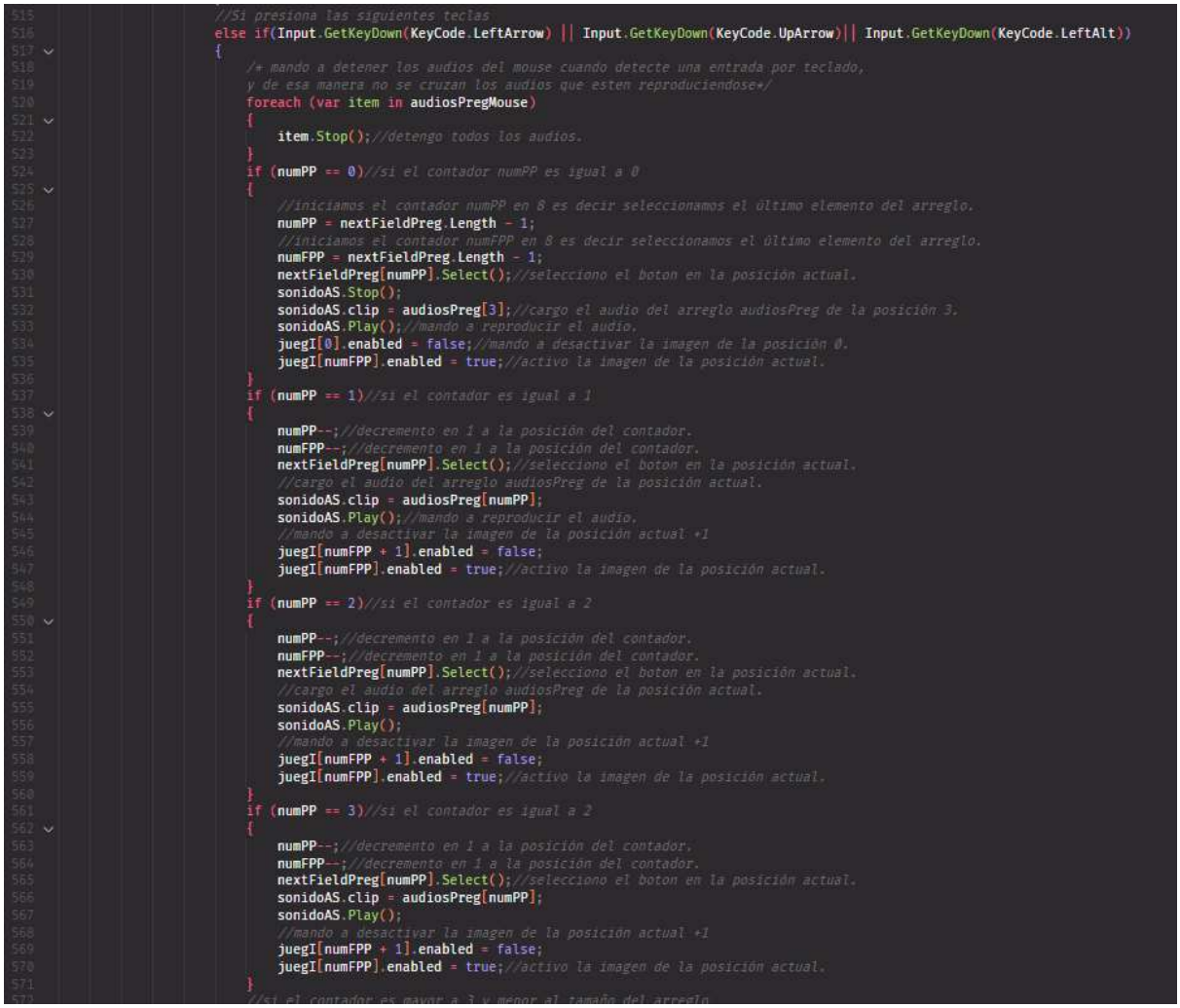


Figura 220. Fragmento de código de la función Tabulando parte 6.



```
//Si presiona las siguientes teclas
else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
{
    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
    y de esa manera no se cruzan los audios que esten reproduciendose*/
    foreach (var item in audiosPregMouse)
    {
        item.Stop();//detengo todos los audios.
    }
    if (numPP == 0)//si el contador numPP es igual a 0
    {
        //iniciamos el contador numPP en 8 es decir seleccionamos el último elemento del arreglo.
        numPP = nextFieldPreg.Length - 1;
        //iniciamos el contador numFPP en 8 es decir seleccionamos el último elemento del arreglo.
        numFPP = nextFieldPreg.Length - 1;
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        sonidoAS.Stop();
        sonidoAS.clip = audiosPreg[3];//cargo el audio del arreglo audiosPreg de la posición 3.
        sonidoAS.Play();//mando a reproducir el audio.
        juegI[0].enabled = false;//mando a desactivar la imagen de la posición 0.
        juegI[numFPP].enabled = true;//activo la imagen de la posición actual.
    }
    if (numPP == 1)//si el contador es igual a 1
    {
        numPP--;//decremento en 1 a la posición del contador.
        numFPP--;//decremento en 1 a la posición del contador.
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        //cargo el audio del arreglo audiosPreg de la posición actual.
        sonidoAS.clip = audiosPreg[numPP];
        sonidoAS.Play();//mando a reproducir el audio.
        //mando a desactivar la imagen de la posición actual +1
        juegI[numFPP + 1].enabled = false;
        juegI[numFPP].enabled = true;//activo la imagen de la posición actual.
    }
    if (numPP == 2)//si el contador es igual a 2
    {
        numPP--;//decremento en 1 a la posición del contador.
        numFPP--;//decremento en 1 a la posición del contador.
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        //cargo el audio del arreglo audiosPreg de la posición actual.
        sonidoAS.clip = audiosPreg[numPP];
        sonidoAS.Play();
        //mando a desactivar la imagen de la posición actual +1
        juegI[numFPP + 1].enabled = false;
        juegI[numFPP].enabled = true;//activo la imagen de la posición actual.
    }
    if (numPP == 3)//si el contador es igual a 2
    {
        numPP--;//decremento en 1 a la posición del contador.
        numFPP--;//decremento en 1 a la posición del contador.
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        sonidoAS.clip = audiosPreg[numPP];
        sonidoAS.Play();
        //mando a desactivar la imagen de la posición actual +1
        juegI[numFPP + 1].enabled = false;
        juegI[numFPP].enabled = true;//activo la imagen de la posición actual.
    }
    //si el contador es mayor a 3 y menor al tamaño del arreglo
```

Figura 221. Fragmento de código de la función Tabulando parte 7.

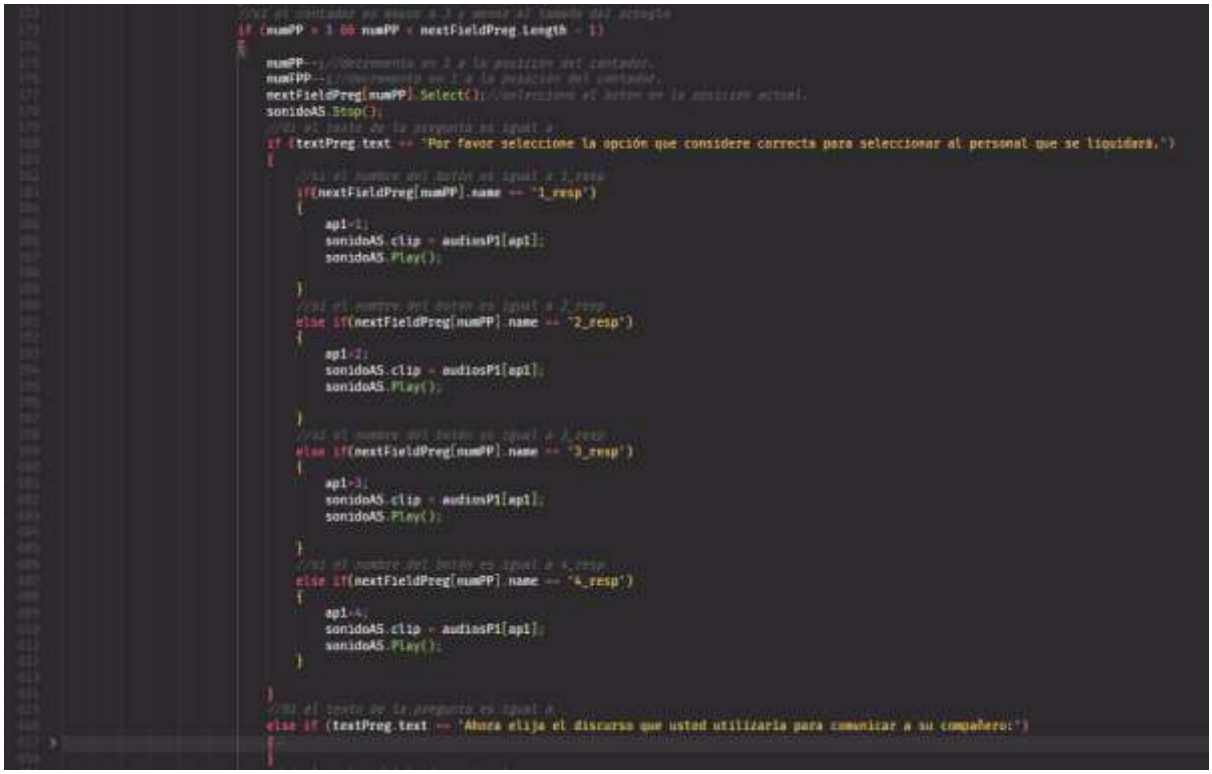


Figura 222. Fragmento de código de la función Tabulando parte 8.

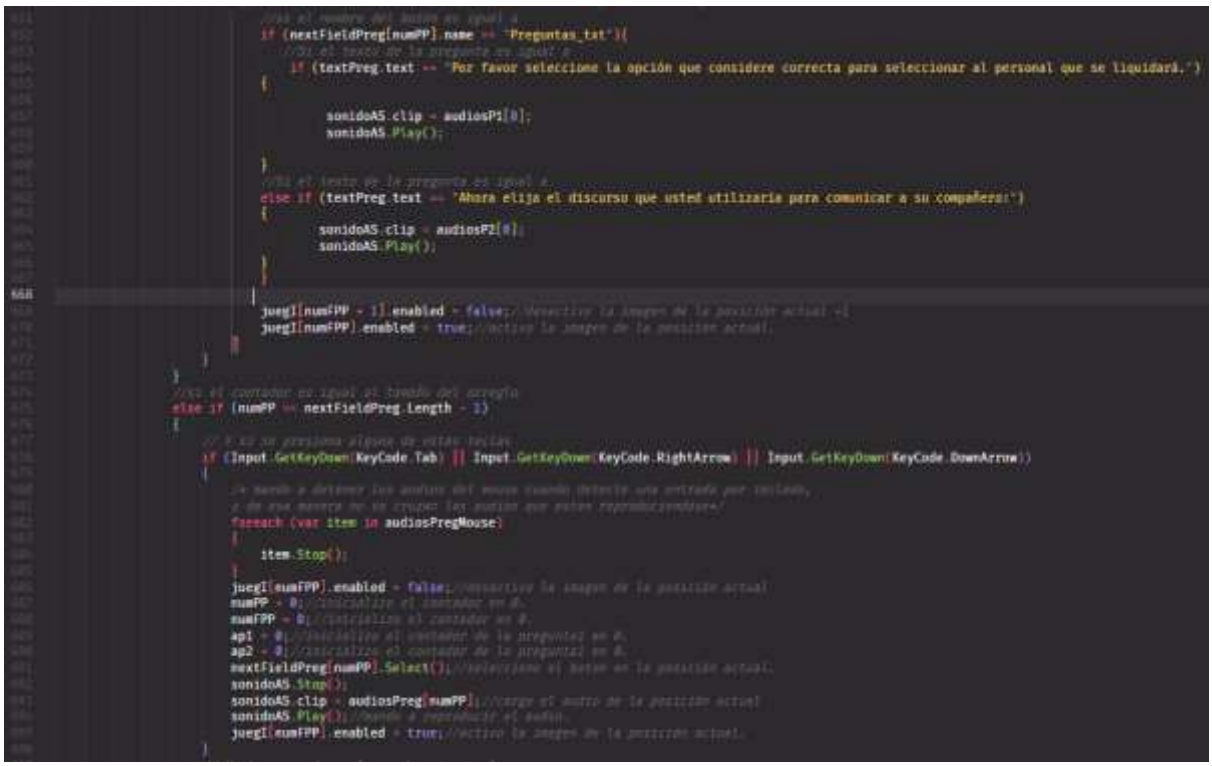


Figura 223. Fragmento de código de la función Tabulando parte 9.

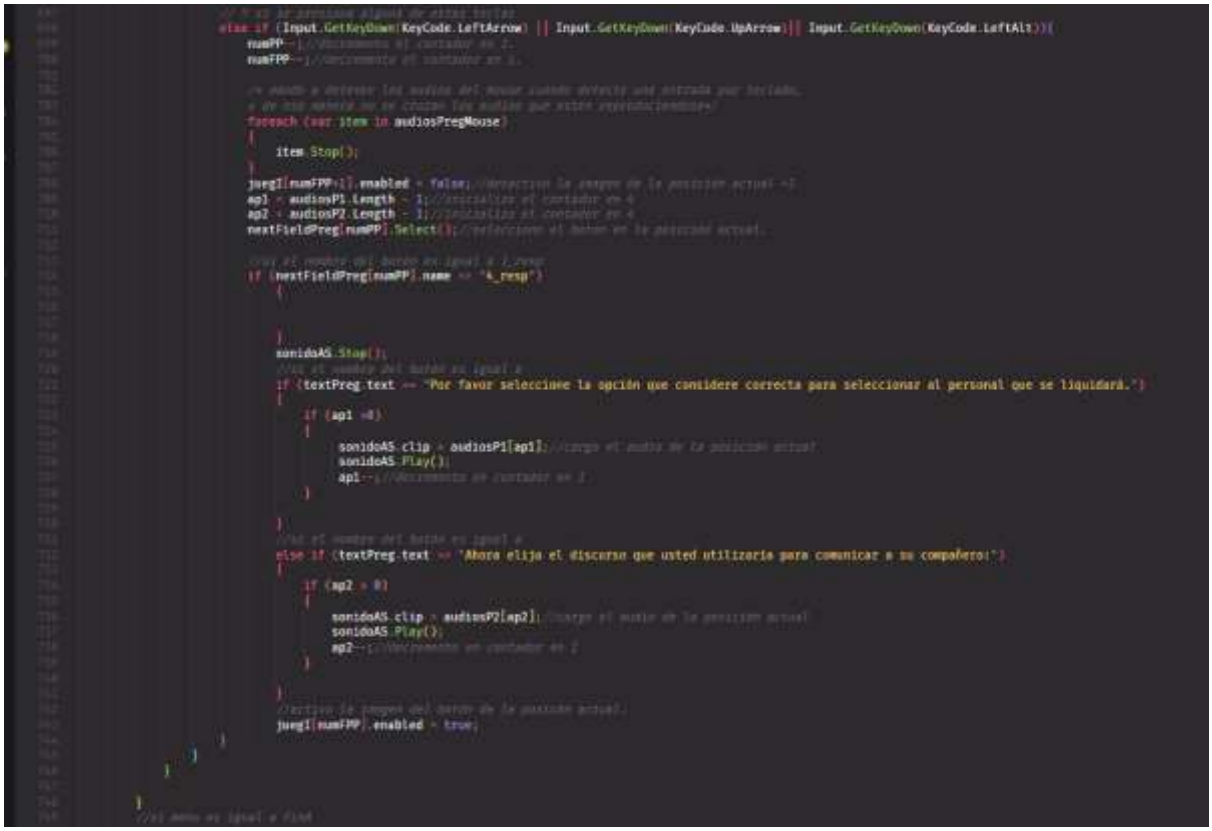
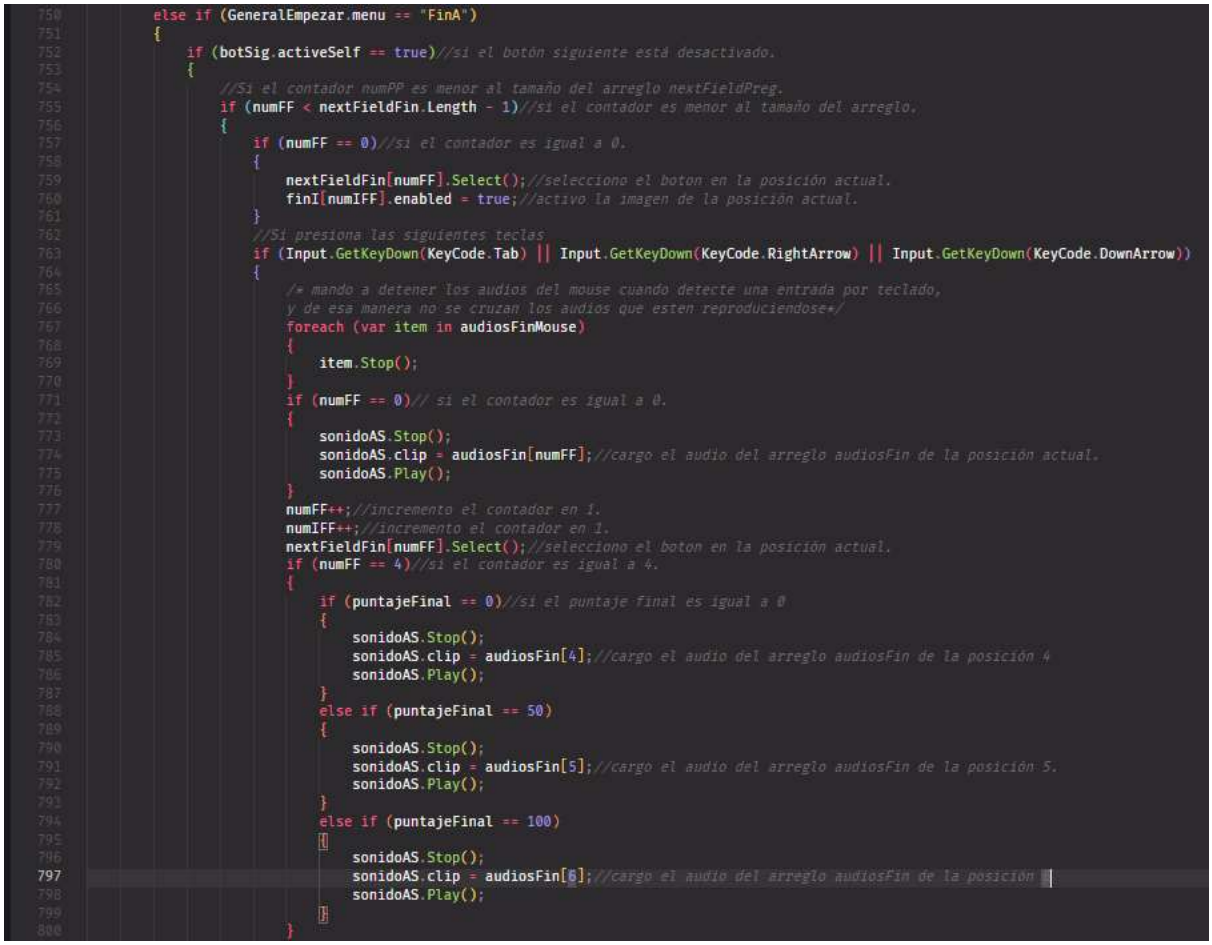


Figura 224. Fragmento de código de la función Tabulando parte 10.



```
else if (GeneralEmpezar.menu == "FinA")
{
    if (botSig.activeSelf == true)//si el botón siguiente está desactivado.
    {
        //Si el contador numPP es menor al tamaño del arreglo nextFieldPreg.
        if (numFF < nextFieldFin.Length - 1)//si el contador es menor al tamaño del arreglo.
        {
            if (numFF == 0)//si el contador es igual a 0.
            {
                nextFieldFin[numFF].Select();//selecciono el boton en la posición actual.
                finI[numIFF].enabled = true;//activo la imagen de la posición actual.
            }
            //Si presiona las siguientes teclas
            if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
            {
                /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                y de esa manera no se cruzan los audios que esten reproduciendose*/
                foreach (var item in audiosFinMouse)
                {
                    item.Stop();
                }
                if (numFF == 0)// si el contador es igual a 0.
                {
                    sonidoAS.Stop();
                    sonidoAS.clip = audiosFin[numFF];//cargo el audio del arreglo audiosFin de la posición actual.
                    sonidoAS.Play();
                }
                numFF++;//incremento el contador en 1.
                numIFF++;//incremento el contador en 1.
                nextFieldFin[numFF].Select();//selecciono el boton en la posición actual.
                if (numFF == 4)//si el contador es igual a 4.
                {
                    if (puntajeFinal == 0)//si el puntaje final es igual a 0
                    {
                        sonidoAS.Stop();
                        sonidoAS.clip = audiosFin[4];//cargo el audio del arreglo audiosFin de la posición 4
                        sonidoAS.Play();
                    }
                    else if (puntajeFinal == 50)
                    {
                        sonidoAS.Stop();
                        sonidoAS.clip = audiosFin[5];//cargo el audio del arreglo audiosFin de la posición 5.
                        sonidoAS.Play();
                    }
                    else if (puntajeFinal == 100)
                    {
                        sonidoAS.Stop();
                        sonidoAS.clip = audiosFin[6];//cargo el audio del arreglo audiosFin de la posición
                        sonidoAS.Play();
                    }
                }
```

Figura 225. Fragmento de código de la función Tabulando parte 11.

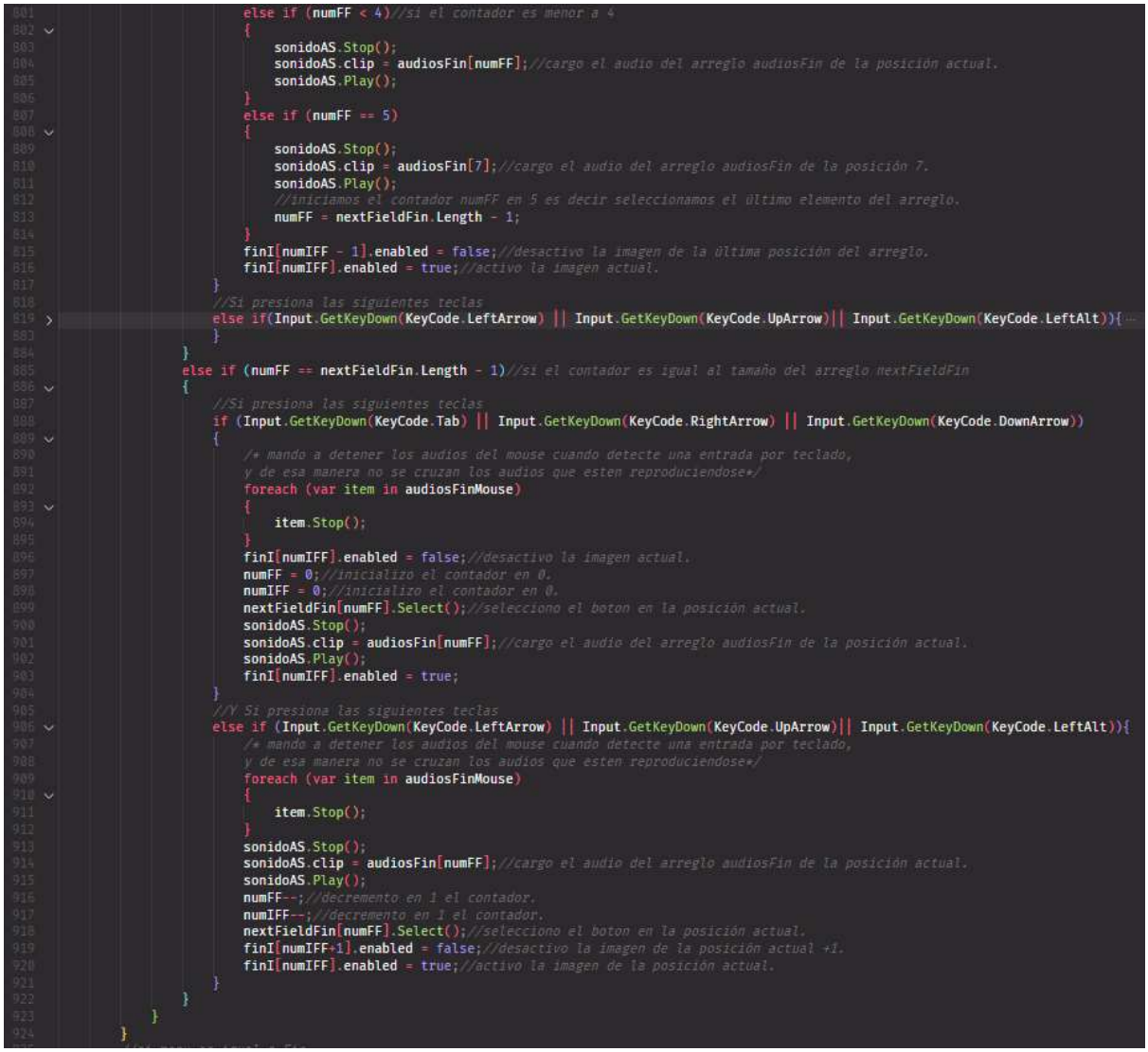
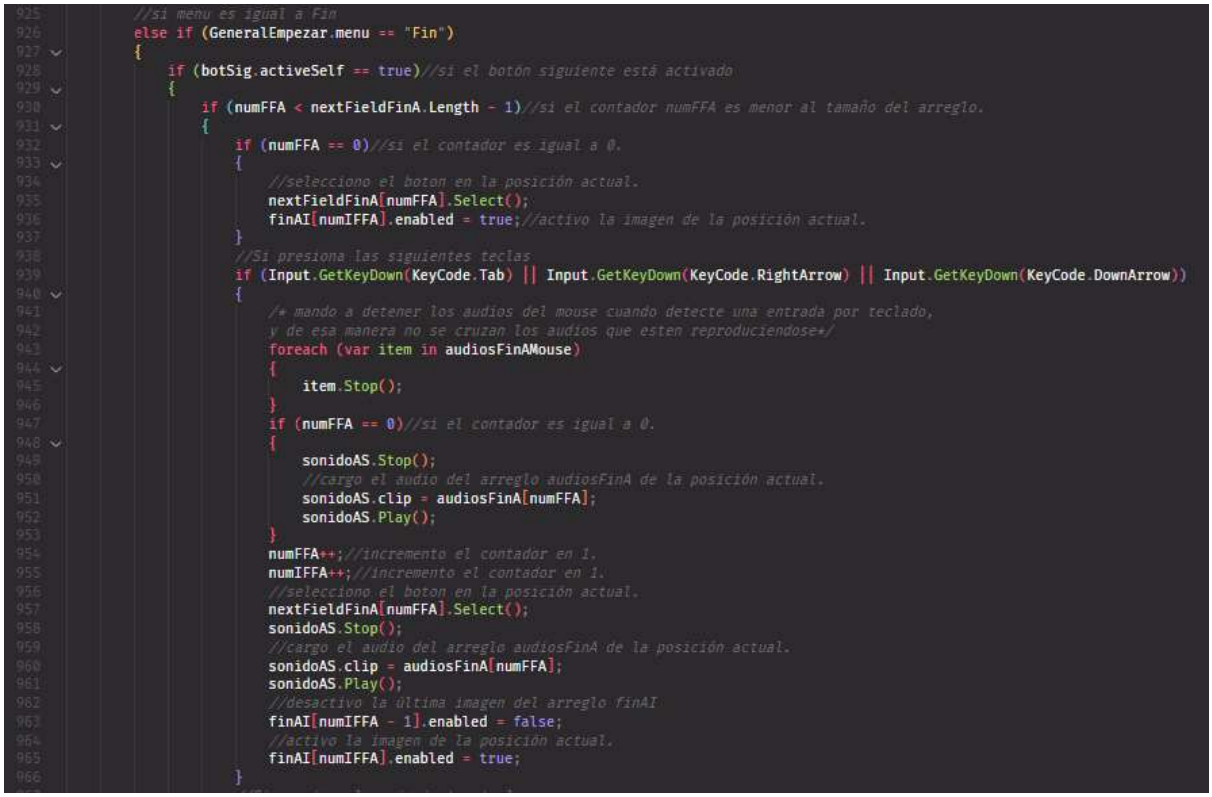


Figura 226. Fragmento de código de la función Tabulando parte 12.



```
        //si menu es igual a Fin
        else if (GeneralEmpezar.menu == "Fin")
        {
            if (botSig.activeSelf == true)//si el botón siguiente está activado
            {
                if (numFFA < nextFieldFinA.Length - 1)//si el contador numFFA es menor al tamaño del arreglo.
                {
                    if (numFFA == 0)//si el contador es igual a 0.
                    {
                        //selecciono el boton en la posición actual.
                        nextFieldFinA[numFFA].Select();
                        finAI[numIFFA].enabled = true;//activo la imagen de la posición actual.
                    }
                    //Si presiona las siguientes teclas
                    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
                    {
                        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                        y de esa manera no se cruzan los audios que esten reproduciendose*/
                        foreach (var item in audiosFinAMouse)
                        {
                            item.Stop();
                        }
                        if (numFFA == 0)//si el contador es igual a 0.
                        {
                            sonidoAS.Stop();
                            //cargo el audio del arreglo audiosFinA de la posición actual.
                            sonidoAS.clip = audiosFinA[numFFA];
                            sonidoAS.Play();
                        }
                        numFFA++;//incremento el contador en 1.
                        numIFFA++;//incremento el contador en 1.
                        //selecciono el boton en la posición actual.
                        nextFieldFinA[numFFA].Select();
                        sonidoAS.Stop();
                        //cargo el audio del arreglo audiosFinA de la posición actual.
                        sonidoAS.clip = audiosFinA[numFFA];
                        sonidoAS.Play();
                        //desactivo la última imagen del arreglo finAI
                        finAI[numIFFA - 1].enabled = false;
                        //activo la imagen de la posición actual.
                        finAI[numIFFA].enabled = true;
                    }
```

Figura 227. Fragmento de código de la función Tabulando parte 13.

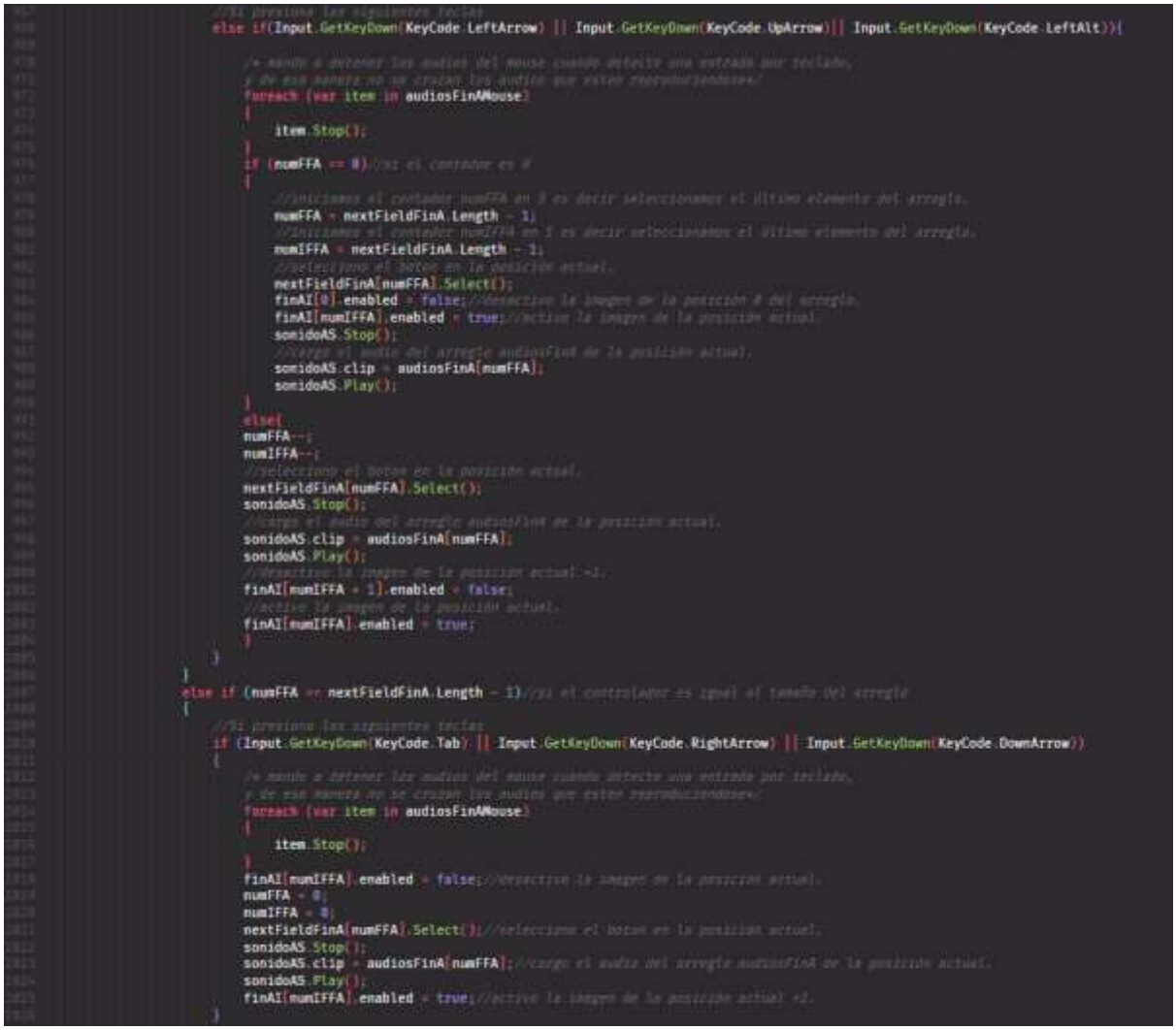


Figura 228. Fragmento de código de la función Tabulando parte 14.

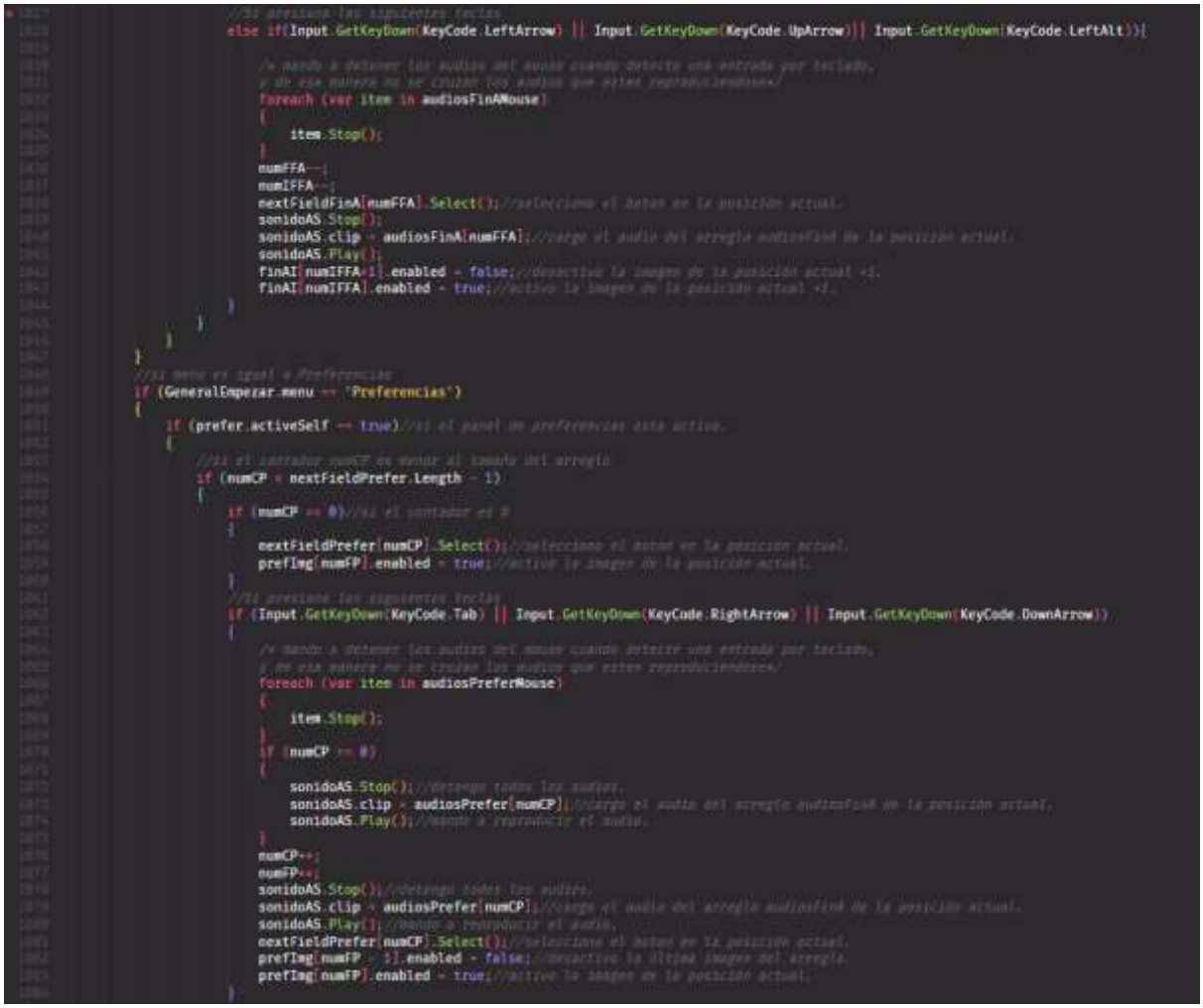


Figura 229. Fragmento de código de la función Tabulando parte 15.

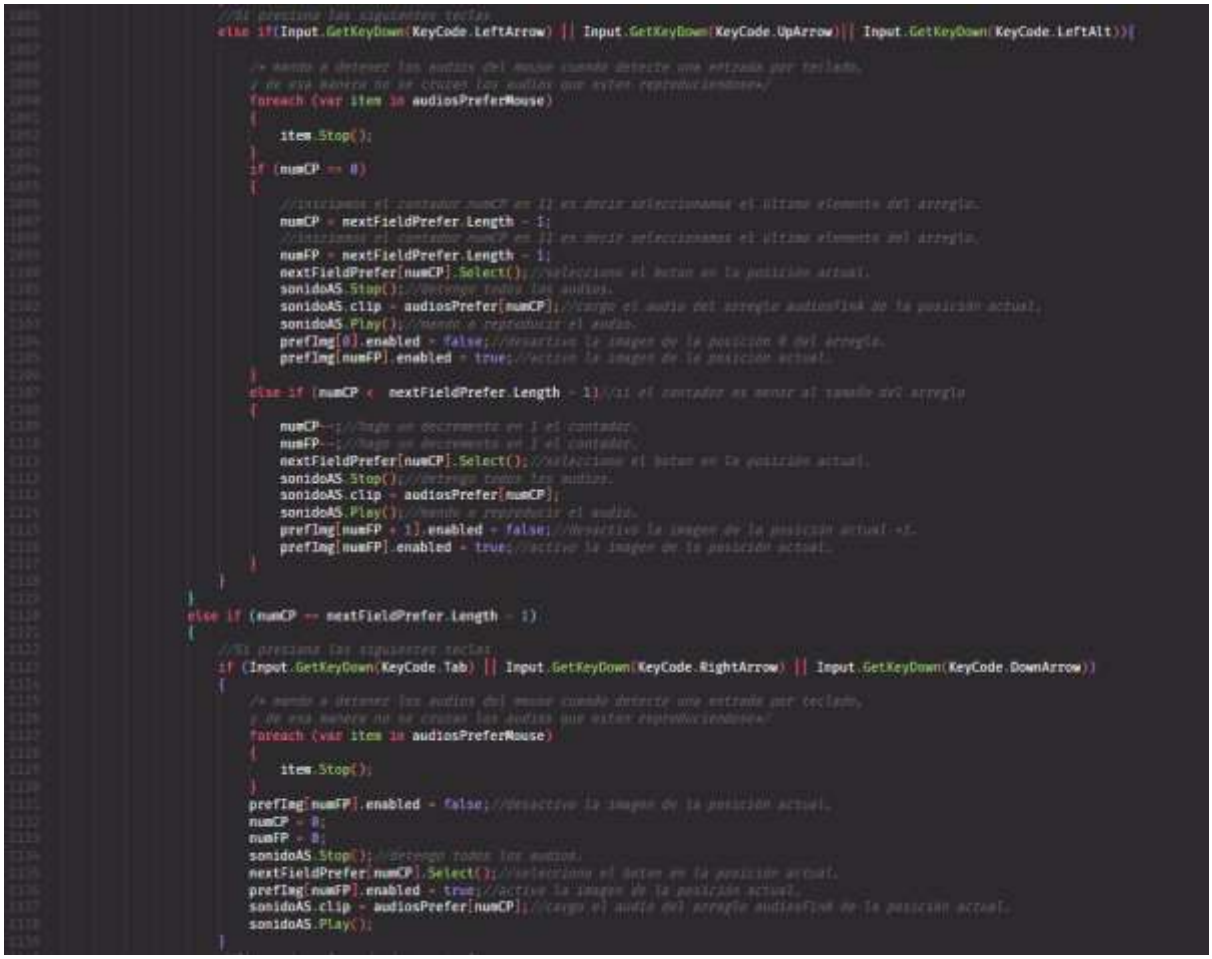


Figura 230. Fragmento de código de la función Tabulando parte 16.

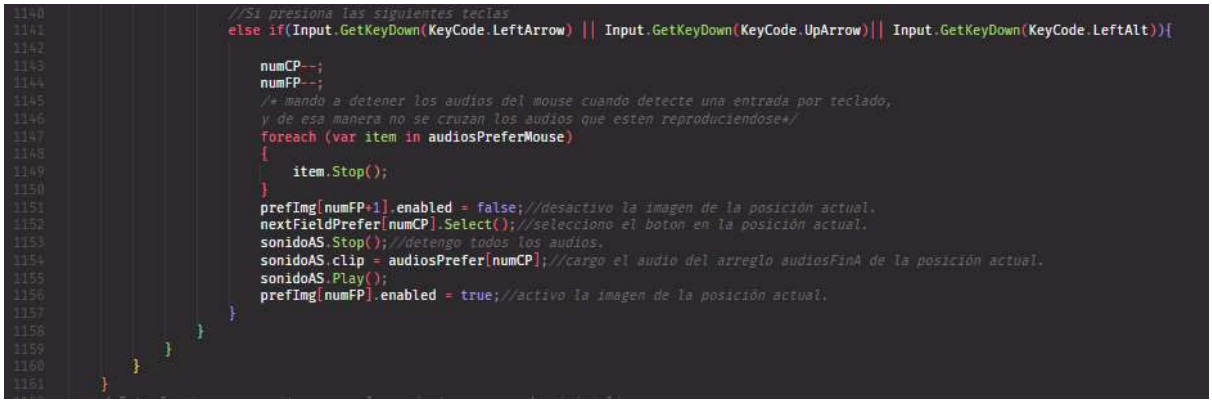


Figura 231. Fragmento de código de la función Tabulando parte 17.

La función *correTiempo()* cuando al navegar ya sea mediante el mouse o teclado y este posicionado en los campos de texto iniciará la captura del tiempo de respuesta de cada actividad presentada al estudiante, dicha función es llamada en cada campo te texto de ingreso de

respuestas.

```
1162    /*Esta funcion me permite poner el cronómetro para poder inicializar
1163    cada vez que respondo las preguntas*/
1164    public void correTiempo()
1165    {
1166        float TimerControl = Time.time - StartTime;
1167        string mins = ((int)TimerControl / 60).ToString("00");
1168        string segs = (TimerControl % 60).ToString("00");
1169        string milisegs = ((TimerControl * 100) % 100).ToString("00");
1170        string TimerString = string.Format("{00}:{01}:{02}", mins, segs, milisegs);
1171        tiempoText.text = TimerString;//en esta variable se guarda el tiempo de respuesta empleado por pregunta.
1172    }
```

Figura 232. Fragmento de código de la función correTiempo.

La función *Empiezo()* permite iniciar la interacción del participante con el simulador laboral, esta función es llamada al presionar el botón empezar del panel principal, ésta a la vez llama la función nuevoJuego() para inicializar todas las variables en cero, también a la función GenerarAleatorisSinRepetir() para que se generen números randomicos y mostrar las preguntas de manera aleatoria.

```
1173    //Esta función es llamada cuando se presiona el botón empezar
1174    public void Empiezo()
1175    {
1176        nuevoJuego();//inicializa todo como un juego nuevo.
1177        textoTit.text = "EJERCITARIO INFORMACIÓN DIFÍCIL";//se muestra en el panel main
1178        textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE";//se muestra en el panel main
1179        datetime = DateTime.Now.ToString("hh:mm:ss");//registra la hora de inicio
1180        //la imagen de la casa la cual corresponde al botón de reiniciar cambia por la imagen del boton siguiente.
1181        sigoFin.image.sprite = ImagenSiguiente;
1182        GenerarAleatoriosSinRepetir();//llamo al metodo de generar números aleatorios.
1183        GeneralEmpezar.move = false;//deshabilito el movimiento de la cámara.
1184        empiezo.enabled = false;//desactivo el botón empezar.
1185        textoEmpiezo.gameObject.SetActive(false);//desactivo el texto del botón empezar
1186        /*Activo todos los objetos del panel de instrucciones*/
1187        juego.SetActive(true);
1188        botSig.SetActive(true);
1189        botPref.SetActive(true);
1190        botSalir.SetActive(true);
1191        tit.SetActive(true);
1192        detall.SetActive(true);
1193        conten.SetActive(true);
1194        GeneralEmpezar.menu = "Jugando";//menu es igual a Jugando
1195        contadorPR = 0;//inicializo el contador en 0.
1196    }
```

Figura 233. Fragmento de código de la función Empiezo.

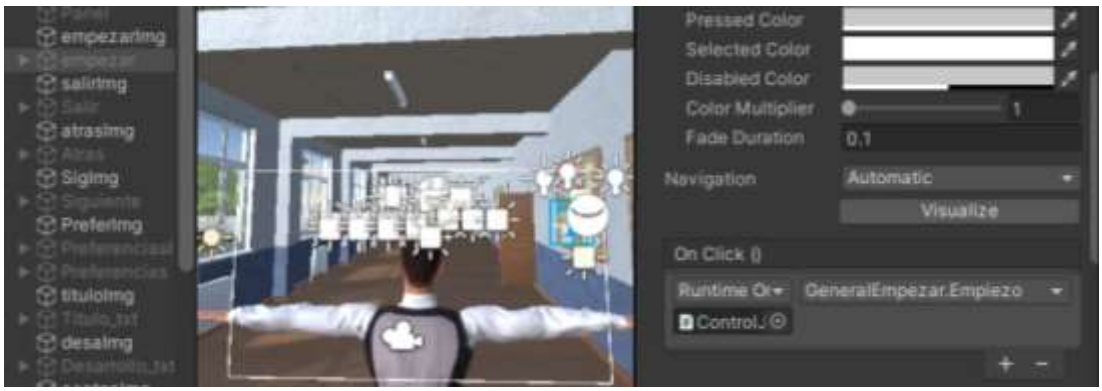


Figura 234. Agregamos el evento On Click y llamamos a la función Empiezo.

La siguiente función es llamada dentro de la función Empezar.

```
1197    /*Funcion para generar números aleatorios*/
1198    void GenerarAleatoriosSinRepetir()
1199    {
1200
1201        numerosGuardados = new List<int>();//crea una lista de numeros aleatorios guardados
1202        numPares = new List<int>();//se crea una lista de numeros pares correspondiente a las preguntas
1203        listaFinalPregs = new List<int>();//se crea lista de preguntas randomicas.
1204        int posicionAleatoria, numpar = 0;
1205        //si el contador es menor al tamaño de la lista contenPregFinal dividimos para dos ya que son dos el número de preguntas.
1206        for (int i = 0; i < (contenPregFinal.Count / 2); i++)
1207        {
1208            do
1209            {
1210                /*hacer que la posicionAleatoria es igual al rango de 0 a 1, en este caso*/
1211                posicionAleatoria = Random.Range(0, (contenPregFinal.Count / 2));
1212              //mientras numerosGuardados contenga un número de posicionAleatoria
1213            } while (numerosGuardados.Contains(posicionAleatoria));
1214
1215            numerosGuardados.Add(posicionAleatoria);//agrego a mi lista el número aleatorio generado.
1216            listaFinalPregs.Add(0);//agrego a mi lista  en la posición 0 el número que generé.
1217            numPares.Add(numpar);//agrego 0 a la lista
1218            numpar = numpar + 2;//Inicializo numpar y digo que es igual a numpar+2;
1219        }
1220        //recorro con un buble y digo si i es menor al número de numerosGuardados +1
1221        for (int i = 0; i < numerosGuardados.Count; i++)
1222        {
1223            //mi listaFinalPregs en la posición i  es igual a el numero numero de preguntas
1224            listaFinalPregs[i] = numPares[numerosGuardados[i]];
1225        }
1226        //recorro la lista y digo si i es menor al número de mi lista numeroGuardados +1.
1227        for (int i = 0; i < listaFinalPregs.Count; i++)
1228        {
1229            //se agregan mis preguntas con sus alternativas.
1230            contenPreg.Add(contenPregFinal[listaFinalPregs[i]]);
1231            //se agregan mis preguntas con sus alternativas +1.
1232            contenPreg.Add(contenPregFinal[listaFinalPregs[i] + 1]);
1233        }
1234    }
```

Figura 235. Fragmento de código de la función GenerarAleatoriosSinRepetir.

La siguiente función es llamada al presionar el botón de preferencias y cuando ésta esta activa se manda a desactivar todas las imágenes del resto de paneles para evitar errores.

```
/*Función que al llamarla mediante el boton de preferencias activa el panel de preferencias*/
public void ActivoPrefer()
{
    if (pefBool == false)//si el botón de preferencias está en false
    {
        prefer.SetActive(true);//mando a activar el panel de preferencias
        GeneralEmpezar.menu = "Preferencias";//menu es igual a Jugando
        tit.SetActive(false);//desactivo el titulo para evitar que se choquen con el panel de preferencias.

        GeneralEmpezar.pefBool = true;//se activa el botón de preferencias.
        if (pregu.activeSelf == true)//si mis preguntas estan activadas
        {
            /* mando a desactivar todas las imagenes del panel de preguntas*/
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            /* mando a desactivar todas las imagenes del panel de instrucciones*/
            foreach (var item in dial)
            {
                item.enabled = false;
            }
        }
        else if (
            //Si el texto de la pregunta es igual a
            textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
        )
        {
            /* mando a desactivar todas las imagenes del panel FinA*/
            foreach (var item in finAI)
            {
                item.enabled = false;
            }
        }
         //Si el texto de la pregunta es igual a
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            /* mando a desactivar todas las imagenes del panel Fin*/
            foreach (var item in finI)
            {
                item.enabled = false;
            }
        }
    }
}
```

Figura 236. Fragmento de código de la función ActivoPrefer.

Figura 237. Agregamos el evento On Click y llamamos a la función ActivoPrefer.

La siguiente función es llamada al presionar el botón salir del panel de preferencias.

```
/*Funcion que al llamarla mediante el boton salir de preferencias desactiva el panel de preferencias*/
public void DesctivoPrefer()
{
    if (pefBool == true)//si el boton de preferencias está activo
    {
        prefer.SetActive(false);//desactiva el panel de preferencias
        tit.SetActive(true);//activa el titulo
        GeneralEmpezar.pefBool = false;//desactiva botón de
        if (pregu.activeSelf == true)//si las preguntas estan activadas
        {
            GeneralEmpezar.menu = "Preguntas";
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            GeneralEmpezar.menu = "Jugando";
        }
        else if (
            textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"
        )
        {
            GeneralEmpezar.menu = "FinA";
        }
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            GeneralEmpezar.menu = "Fin";
        }
        //Cada vez que cierre el panel de preferencias me direccionará hacia el panel que este activado.
    }
}
```

Figura 238. Fragmento de código de la función DesctivoPrefer.

Figura 239. Agregamos el evento On Click y llamamos a la función DesctivoPrefer.

La siguiente función es llamada dentro de la función Siguiente() y la función Atrás(), se usa para reiniciar los valores de los contados en cero cada vez que se presione le botón atrás o siguiente.

```
1314    /*Funcion que al llamarla desde la funcion siguiente o atras reinicia sus valores a cero*/
1315    public void ReinicioSeleccion()
1316    {
1317        /*Inicializo todos los contadores en cero*/
1318        numC = 0;
1319        numI = 0;
1320        numCP = 0;
1321        numFP = 0;
1322        numPP = 0;
1323        numFPP = 0;
1324        numFF = 0;
1325        numIFF = 0;
1326        numFFA = 0;
1327        numIFFA = 0;
1328
1329        nextFieldDiag[numC].Select();//selecciono el objeto que se encuentra en la posición
1330        nextFieldPrefer[numCP].Select();//selecciono el objeto que se encuentra en la posición
1331        nextFieldPreg[numPP].Select();//selecciono el objeto que se encuentra en la posición
1332        nextFieldFin[numFF].Select();//selecciono el objeto que se encuentra en la posición
1333        nextFieldFinA[numFFA].Select();//selecciono el objeto que se encuentra en la posición
1334
1335        /* mando a desactivar todas las imagenes del panel de preferencias/**/
1336        foreach (var item in prefImg)
1337        {
1338            item.enabled = false;
1339        }
1340        foreach (var item in dial)
1341        {
1342            item.enabled = false;
1343        }
1344        foreach (var item in juegI)
1345        {
1346            item.enabled = false;
1347        }
1348        foreach (var item in finI)
1349        {
1350            item.enabled = false;
1351        }
1352        foreach (var item in finAI)
1353        {
1354            item.enabled = false;
1355        }
1356
1357
1358    }
```

Figura 240. Fragmento de código de la función ReinicioSeleccion.

La siguiente función es llamada desde la función *Respondo ()* y al presionar el botón siguiente, En esta función también presentan las actividades a resolver para él participante, para ello primero consultamos si *menu* es igual a *Preguntas*, si es que si entonces se muestra al participante la actividad junto a las posibles respuestas, cuando seleccione el botón la opción que crea correcta internamente se estará llamando la función Respondo() quien tomará las respuestas seleccionadas por el participante y las comparará contra el arreglo de respuestas correctas, una vez que el participante conteste la última pregunta pasará al panel de calificación eso lo controlamos por medio de un condicional en el que decimos si el contador de preguntas es igual al número de preguntas que tiene la lista de preguntas, significará que ha terminado de resolver las preguntas lo cual lo llevará al panel de calificación, una vez ahí se mostrará el puntaje obtenido para ello se consulta si la variable *menu* es igual a *FinA*, si es que si, entonces dice, si el puntaje obtenido es 0 la calificación es 0%, si el puntaje obtenido es 50 entonces la calificación es 50% y si el puntaje obtenido es 100 entonces tendrá el 100%; finalmente cuando

la variables *menu* sea igual a Fin entonces estaremos en el último panel, una vez que estemos internamente se llama a la función *Post()* para consumir el servicio web.

Figura 241. Fragmento de código de la función Siguiente parte 1.

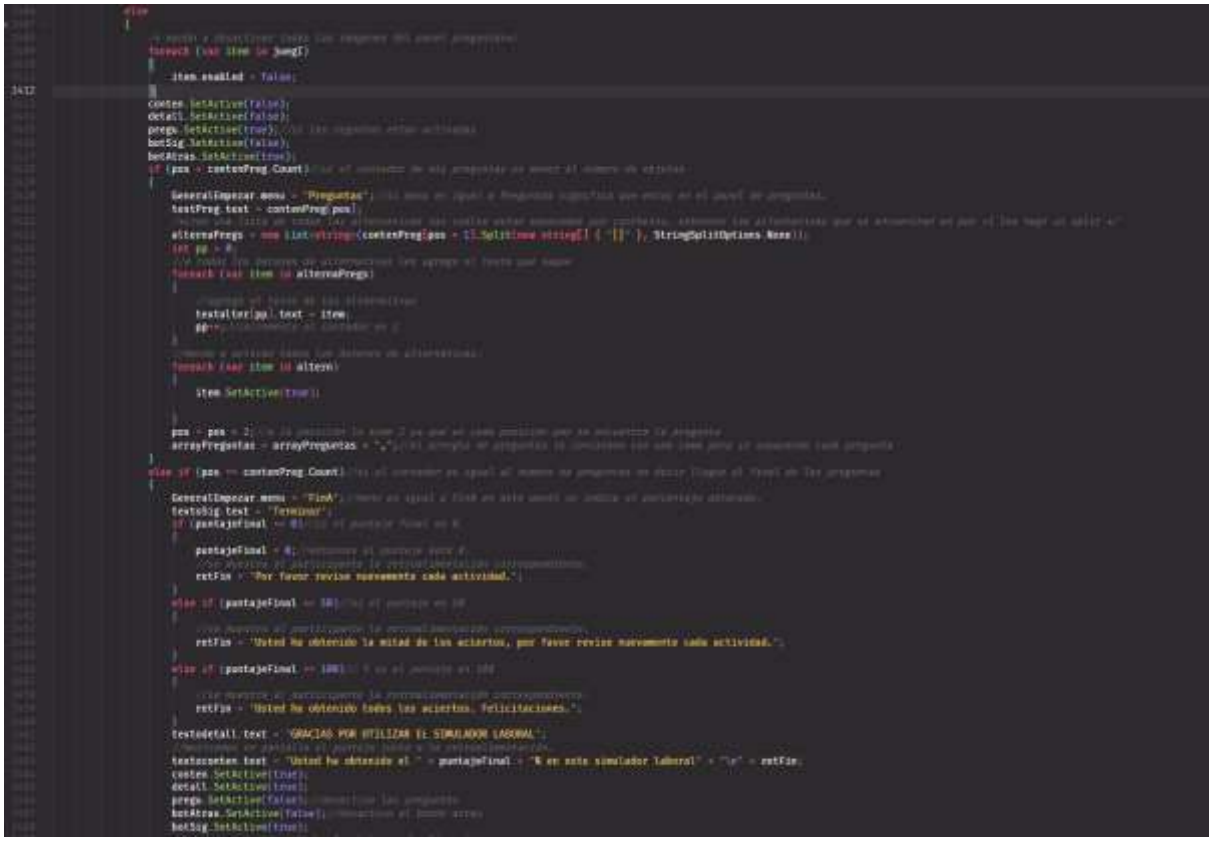


Figura 242. Fragmento de código de la función Siguiente parte 2.

Figura243. Fragmento de código de la función Siguiente parte 3.

Figura 244. Agregamos el evento On Click y llamamos a la función Siguiente.

Se crea la función *Post()* para consumir el servicio web, pasamos la url del servidor web indicando el tipo de método en este caso POST ya que se va a insertar datos en el servidor web, generamos el archivo json de mis preguntas, envío el json y finalmente hago la petición al servidor web mediante SendWebRequest().

```
public void Post()
{
    var request = new UnityWebRequest("https://simulab.edutech-project.org/api/registrarActividad", "POST");
    string auxJSON = JsonUtility.ToJson(eje);
    byte[] bodyRaw = Encoding.UTF8.GetBytes(auxJSON);
    request.uploadHandler = (UploadHandler)new UploadHandlerRaw(bodyRaw);
    request.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
    request.SetRequestHeader("Content-Type", "application/json");
    request.SendWebRequest();
}
```

Figura 245. Fragmento de código de la función Post().

```
public void Atras()
{
    ReiniciarSeleccion();
    if (prefer.activeSelf == false)
    {
        TextoAgregarModif = "Actu";
        sonidoAS.Stop();
        contan.SetActive(false);
        detall.SetActive(false);
        pregu.SetActive(true);
        pos = pos - 2;
        numPreg = 0;
        puntajeFinal = 0;
        if (pos > 1)
        {
            GeneralEmpezar.menu = "Preguntas";
            textPreg.text = contenPreg[pos - 1];
            alternaPregs = new List<string>(contenPreg[pos - 1].Split(new string[] { "][" }, StringSplitOptions.None));
            int pp = 0;
            foreach (var item in alternaPregs)
            {
                textalter[pp].text = item;
                pp++;
            }
            foreach (var item in altern)
            {
                item.SetActive(true);
            }
            contRes--;
            contadorPR--;
            if (contadorPR == 0)
            {
                arrayPreguntas = ",";
            }
            else
            {
                arrayPreguntas = arrayPreguntas.Substring(0, arrayPreguntas.IndexOf(",") + 1);
            }
        }
```

Figura 246. Fragmento de código de la función Atrás parte 1.

```
        else
        {
            if (pos == 0)//si el contador es 0 quiere decir que estamos en el panel de instrucciones
            {
                puntajeFinal = 0;//inicializamos el puntajeFinal en 0.
                /*Activamos los siguientes objetos de la interfaz*/
                juego.SetActive(true);
                botSig.SetActive(true);
                botPref.SetActive(true);
                botSalir.SetActive(true);
                botAtras.SetActive(false);//desactivo el botón atras
                tit.SetActive(true);
                detall.SetActive(true);
                conten.SetActive(true);
                arrayPreguntas = "";//limpio la variable de preguntas.
                GeneralEmpezar.menu = "Jugando";
                pregu.SetActive(false);//desactivo las preguntas
                //mando a desactivar todos los botones de alternativas.
                foreach (var item in altern)
                {
                    item.SetActive(false);
                }
            }
        }
    }
```

Figura 247. Fragmento de código de la función Atras() parte 2.

Figura 248. Agregamos el evento OnClick() y llamamos a la función Atras().

```
public void Respondo(TextMeshProUGUI textoBot)
{
    if (textoBot.name == rescor[contRes])//si el nombre del texto es igual a la respuesta correcta regitrada .
    {
        puntajeFinal = puntajeFinal + 50;//obtiene un puntaje de 50%.
    }

    if (tiemposResp[contadorPR] == null)//si el tiempo de respuesta de mi contador es igual a null.
    {
        tiemposResp[contadorPR] = tiempoText.text;//si es igual a null le asigno un nuevo tiempo.
    }
    else
    {//caso contrario
        //creo un arreglo de string en el cual se guarda el tiempo de respuesta anterior
        string[] tiemResAnt = tiemposResp[contadorPR].Split(':');//separo hora,minuto y segundo por :
        //creo un arreglo de string en el cual se guarda el tiempo de respuesta nuevo
        string[] tiemResNuevo = tiempoText.text.Split(':');
        /*Para obtener el tiempo total de respuesta sumamos el tiemResNuevo + tiempoResAnt*/
        int segundos = Int32.Parse(tiemResNuevo[2]) + Int32.Parse(tiemResAnt[2]);
        int minutos = Int32.Parse(tiemResNuevo[1]) + Int32.Parse(tiemResAnt[1]);
        int horas = Int32.Parse(tiemResNuevo[0]) + Int32.Parse(tiemResAnt[0]);
        int auxH = 0;//auxiliar para horas
        int auxM = 0;//auxiliar para minutos
        string horF = "", minF = "", segF = "";
        while (segundos > 60)//mientras segundos sea menor a 60
        {
            segundos = segundos - 60;//segundos es igual a segundos - 60
            auxM++;//incremento en uno el contador.
        }
        minutos = minutos + auxM;//minutos es igual a minutos más la auxM
        while (minutos > 60)//mientras minutos sea menor a 60
        {
            minutos = minutos - 60;//minutos es igual a minutos -60
            auxH++;//incremento en uno el contador.
        }
        horas = horas + auxH;//entonces horas es igual a horas mas auxH
        if (horas < 10)//si horas es menor a 10
        {
            horF = "0" + horas;//horF es igual a 0 + mas horas.
        }
        else
        {//caso contrario
            horF = "" + horas;//se queda igual
        }
        if (minutos < 10)//si minutos es menor a 10
        {
            minF = "0" + minutos;
        }
        else
        {
            minF = minutos + "";//se queda igual
        }
        if (segundos < 10)//si segundos es menor a 10
        {
            segF = "0" + segundos;
        }
        else
        {
            segF = segundos + "";//se queda igual
        }
        tiemposResp[contadorPR] = horF + ":" + minF + ":" + segF;//seteo el nuevo tiempo que me demore respondiendo esa pregunta.
    }
```

Figura 249. Fragmento de código de la función Respondo parte 1.

```
        /*cada vez que respondo una pregunta guarda lo siguiente*/
        ppre = new Pregunta();//declaro una nueva pregunta
        ppre.setRespuestaIngresada(textoBot.name);//seteo la respuesta ingresada
        ppre.setTiempoRespuesta(tiemposResp[contadorPR]);//seteo el tiempo de respuesta
        ppre.setNumeroPregunta(numerosGuardados[contadorPR] + 1);//seteo el numero de pregunta
        if (TextoAgregoModif == "Nuevo")//si estoy ingresando la respuesta textoAgregoModif es Nuevo.
        {
            eje.addPregunta(ppre);//agrego la pregunta al archivo json
        }
        else if (TextoAgregoModif == "Actu")//si presioné el boton atras textoAgregoModif es Actu
        {
            eje.updatePregunta(ppre, eje.preguntas[contadorPR]);//actualizo la respuesta solo en esa posición
        }

        sonidoAS.Stop();//como voy a pasar al siguiente panel detengo todos los audios.
        contRes++;//el contador de las respuestas aumenta en 1.
        numPreg++;//el numero de la pregunta aumenta en 1.
        numPP = 0;//inicializo en 0 el contador de preguntas.
        numFPP = 0;//inicializo en 0 el contador de imagenes de las preguntas.
        contadorPR++;//incremento en 1 el contador.
        Siguiente();//lamo a la función siguiente para que me de paso a la siguiente pregunta.
    }
```

Figura 250. Fragmento de código de la función Respondo parte 2.

La función *MenuSel()* permite controlar en que pantalla estamos en ese momento, en donde la variable *menu* va cambiando según el panel en el que esté, es decir cuando menu es igual a Main significa que está en el panel principal o main, cuando menu sea igual Jugando es que estamos en el panel de instrucciones, cuando menu es igual a Preguntas es que esta en el panel de Preguntas, cuando menu es igual a FinA es que esta en el panel de calificación y cuando menu es igual a Fin es que está en último panel en donde se muestra la retroalimentación del ejercitario resuelto.

```
    }//método para controlar en que menu o panel me encuentro.
    public static void MenuSel(string tipoMen)
    {
        GeneralEmpezar.menu = tipoMen;
    }
}//fin de clase GeneralEmpezar
```

Figura 251. Fragmento de código de la función MenuSel().

1.2.1.3 Declaración de clases serializadas

- **Clase Serializada Pregunta**

La clase serializada Pregunta hace referencia a la estructura de cada pregunta, es decir cada vez que se responda una pregunta, se guardaran los campos de datos correspondiente a la pregunta.

```
[Serializable]
/*Clase que hace referencia a la estructura de cada pregunta*/
public class Pregunta
{
    public string respuestaIngresada;
    public string tiempoRespuesta;
    public int numeroPregunta;
    public DateTime inicio;
    public DateTime fin;
    public string getRespuetaIngresada()
    {
        return respuestaIngresada;
    }
    public string getTiempoRespuesta()
    {
        return tiempoRespuesta;
    }
    public void setNumeroPregunta(int numeroPregunta)
    {
        this.numeroPregunta = numeroPregunta;
    }
    public int getPregunta()
    {
        return numeroPregunta;
    }
    public void setInicio()
    {
        this.inicio = System.DateTime.Now;
    }
    public void setFin()
    {
        this.fin = System.DateTime.Now;
    }
    public void setTiempoRespuesta(string tiempoRespuesta)
    {
        this.tiempoRespuesta = tiempoRespuesta;
    }
    public void setRespuestaIngresada(string respt)
    {
        this.respuestaIngresada = respt;
    }
}
```

Figura 252. Fragmento de código de la clase serializada Pregunta.

La clase serializada SimuladorData hace referencia al json que se va a generar, ya que un simulador laboral puede tener n preguntas, se crea una lista de preguntas.

```
public class SimuladorData
{
    /*atributos de archivo json*/
    public string correo;
    public int numeroEjercitario;
    public string tiempoInicio;
    public string tiempoFin;
    public string fechaDeActividad;
    /* Ya que un simulador puede tener n preguntas se crea una lista de preguntas. */
    public List<Pregunta> preguntas = new List<Pregunta>();

    public void ReinicioPreguntas()
    {
        /* Se crea una lista del objeto preguntas. */
        preguntas = new List<Pregunta>();
    }
    public void setNumeroEjercitario(int numero)
    {
        this.numeroEjercitario = numero;
    }
    public int getNumeroEjercitario()
    {
        return this.numeroEjercitario;
    }
    public void setTiempoInicio(string tiempoInicio)
    {
        this.tiempoInicio = tiempoInicio;
    }
    public string getTiempoInicio()
    {
        return this.tiempoInicio;
    }
    public void setTiempoFin(string tiempoFin)
    {
        this.tiempoFin = tiempoFin;
    }
    public string getTiempoFin()
    {
        return this.tiempoFin;
    }
    public void setFechaActividad(string fechaDeActividad)
    {
        this.fechaDeActividad = fechaDeActividad;
    }
    public string getFechaActividad()
    {
        return this.fechaDeActividad;
    }
    public void setCorreo(string correo)
    {
        this.correo = correo;
    }
    public string getCorreo()
    {
        return this.correo;
    }
    public void addPregunta(Pregunta nuevaPreg)
    {
        this.preguntas.Add(nuevaPreg);
    }
    public void updatePregunta(Pregunta upPreg, Pregunta antPreg)
    {
        this.preguntas.Remove(antPreg);
        this.preguntas.Add(upPreg);
    }
    public Pregunta readPregunta(int indice)
    {
        return preguntas[indice];
    }
    public void setInicioPregunta(int numero)
    {
        this.preguntas[numero].setInicio();
    }

}
```

Figura 253. Fragmento de código de la clase serializada SimuladorData.

```
{"correo":"naraujop@gmail.com",
"numeroEjercitario":10,
"tiempoInicio":"03:17:03",
"tiempoFin":"03:17:12",
"fechaDeActividad":"2022-10-22 03:17:12"
,"preguntas":[{"respuestaIngresada":"1",
                        "tiempoRespuesta":"00:03:86",
                        "numeroPregunta":1},
                        {"respuestaIngresada":"1",
                        "tiempoRespuesta":"00:02:68",
                        "numeroPregunta":2}]}
```


Figura 254. Estructura de preguntas que se genera en el archivo json.

1.4.2 Script MouseOver

1.4.2.1 Declaración de variables y relación con objetos de interfaz de Unity

El siguiente script permite que cuando el mouse está sobre un botón se detengan todos los sonidos que hayan estado reproduciéndose y solo me reproduce el audio que esté en dicho botón y detiene todo cuando el mouse sale del botón.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;

public class MouseOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
/*Además de la extensión MonoBehaviour se debe agregar la extensión
IPointerEnterHandler(mientras el mouse esté sobre el objeto) y
IPointerExitHandler(cuando el mouse salga del objeto)*/
{
    /*se declara variable booleana para controlar la reproducción
    de audio al pasar el mouse sobre el botón*/
    public bool isOver = false;
    /*Audiosource audio del objeto que va a sonar y Audiosource del
    objeto generalTextos para que todos los audios se detengan*/
    public AudioSource audio, generalTextos;
    //y el sonido que se carga para que suene
    public AudioClip soni;

/*Se hace uso del método por defecto el cual permite reproducir el
audio de un botón cuando este esté sobre él*/
    public void OnPointerEnter(PointerEventData eventData)
    {   //detiene todos los audios
        audio.Stop();
        //detiene los sonidos generales
        generalTextos.Stop();
        //va a decir que esta sobre el objeto
        isOver = true;
        audio.clip = soni;//agrego el audio

        audio.Play();//reproduce el audio
    }
/*Se hace uso del método por defecto el cual permite dejar de reproducir cuando el mouse ya no esté sobre el botón*/
    public void OnPointerExit(PointerEventData eventData)
    {
        isOver = false;
        audio.Stop();//detiene todos los audios.
        generalTextos.Stop();//detiene los sonidos generales
    }
}
```

Figura 255. Código fuente de script MouseOver.

Figura 256. Relación de objeto con la variable declarada.

1.4.3 Script OverTextos.cs

Este script permite detener todos los audios que estén reproduciéndose y reproduce el audio de un botón cuando el mouse está sobre él, y cuando el mouse sale del botón se detiene todo.

1.4.4.1 Declaración de variables

```
Assets > Scripts > OverTextos.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;

public class OverTextos : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false;
    public AudioClip[] audis;
    public AudioSource source, generalTextos;/*Audiosource audio del objeto que va a sonar y Audiosource del
    objeto generalTextos para que todos los audios se detengan*/
    [SerializeField]
    private TextMeshProUGUI texto;//declaro una variable de tipo texto
```

Figura 257. Declaración de variables.

1.4.4.2 Declaración de funciones

Esta función ocurre cuando el mouse está sobre el botón.

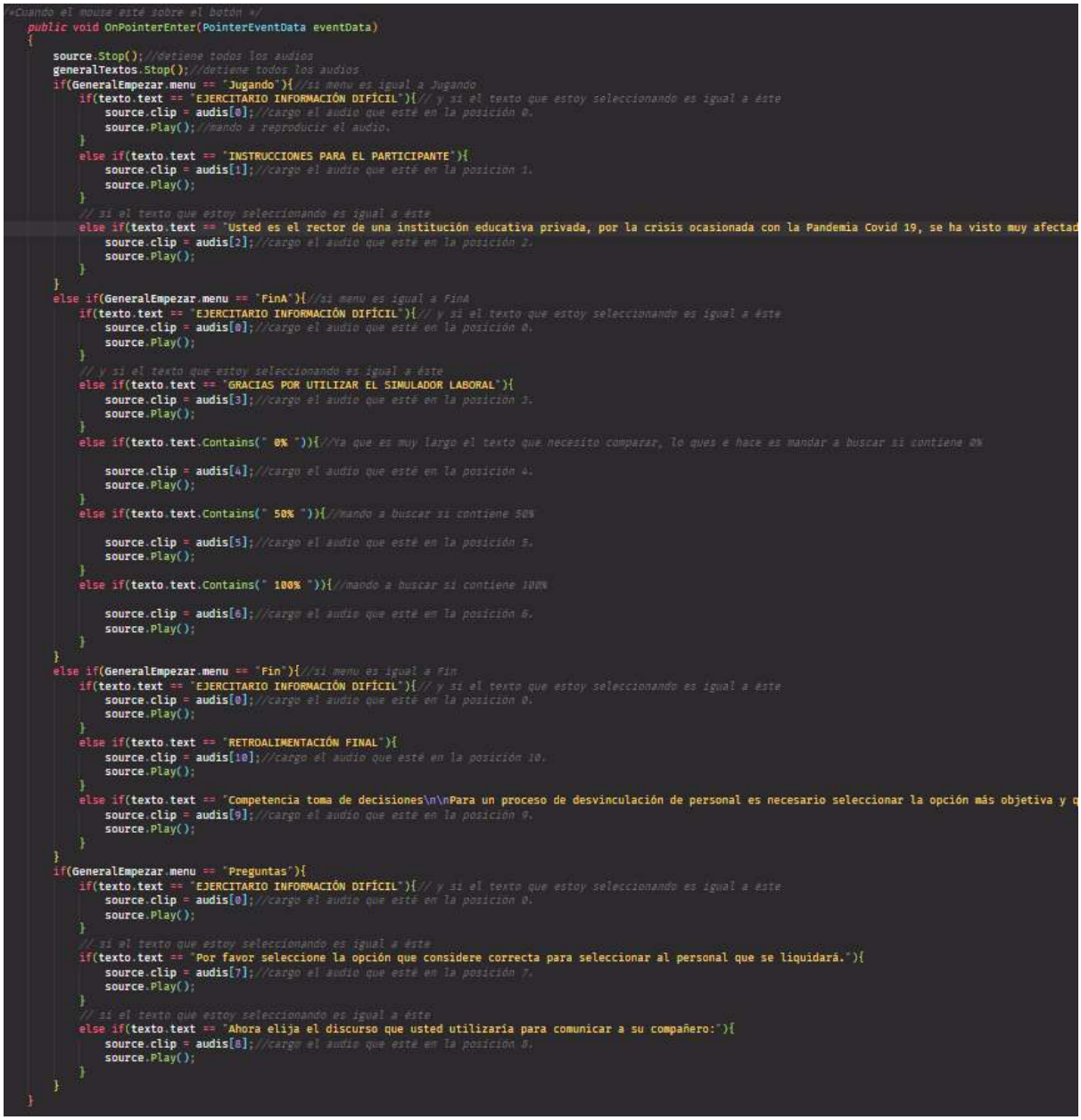


Figura 258. Funcion OnPointerEnter() cuando el mouse está sobre el botón.

Esta función ocurre cuando el mouse sale del botón.

```
    public void OnPointerExit(PointerEventData eventData)
    {
        //quito el mouse y se detiene todo
        isOver = false;
        source.Stop();//detendrá todos los audios.
        generalTextos.Stop();
    }
}
```

Figura 259. Función OnPointerExit() cuando el mouse sale del botón.

1.4.4 Script OpcionesOver.cs

El siguiente script permitirá que cuando el mouse esté sobre un botón reproduzca un audio, en este caso se utiliza los botones correspondientes a las alternativas de cada pregunta, para ello dentro de la función OnPointerEnter() usamos un condicional en donde le decimos que reproduzca el audio de dicho botón solo si el texto del enunciado de la pregunta es igual al texto que estamos mandando a comparar.

1.4.5.1 Declaración de variables y funciones

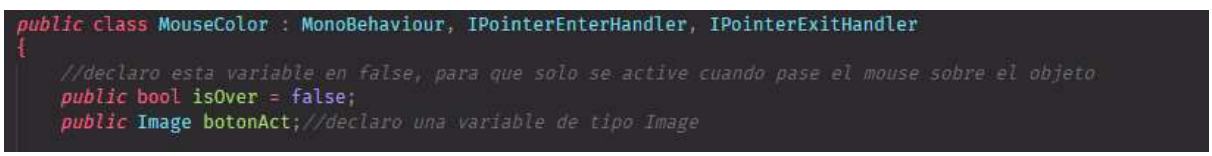
```
Scripts > OpcionesOver.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;

public class OpcionesOver : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false;//declaro esta variable en false
    public AudioClip[] p1, p2; //arreglos para cargar audios de las preguntas
    //source del objeto que va a sonar y generalTextos para que todos los audios se detengan
    public AudioSource source, generalTextos;
    [SerializeField]
    private TextMeshProUGUI texto;
    int numBoton=0;

    public void OnPointerEnter(PointerEventData eventData)
    {
       generalTextos.Stop();
       source.Stop();
       // si el texto que estoy seleccionando es igual a este
        if(texto.text == "Por favor seleccione la opción que considere correcta para seleccionar al personal que se liquidará."){
            source.clip = p1[numBoton];//cargo el audio del arreglo de la pregunta 1
            source.Play();//mando a reproducir el audio
        }
        // si el texto que estoy seleccionando es igual a este
        if(texto.text == "Ahora elija el discurso que usted utilizaría para comunicar a su compañero:"){
            source.clip = p2[numBoton];//cargo el audio del arreglo de la pregunta 2
            source.Play();//mando a reproducir el audio
        }
        isOver = true;//esta sobre el objeto

    }
    //sucede cuando el mouse sale del botón
    public void OnPointerExit(PointerEventData eventData)
    {
        //se detiene todo
        isOver = false;
        source.Stop();
        generalTextos.Stop();
    }
}
```

Figura 260. Script OpcionesOver().

1.4.5 Script MouseColor.cs

Este script permite resaltar los objetos cuando el mouse este sobre alguno de ellos, es decir, cuando el mouse esté sobre un botón éste cambiara de color, de esta manera el estudiante sabrá exactamente en donde está posicionado, para ello se ha implementado dos funciones, cambiar de color cuando el mouse esté sobre algún botón (función *OnPointerEnter ()*) y cuando el mouse salga de dicho botón se activará la función *OnPointerExit()* volviendo al color anterior del botón.

1.4.6.1 Declaración de variables y relación con objetos de interfaz de Unity



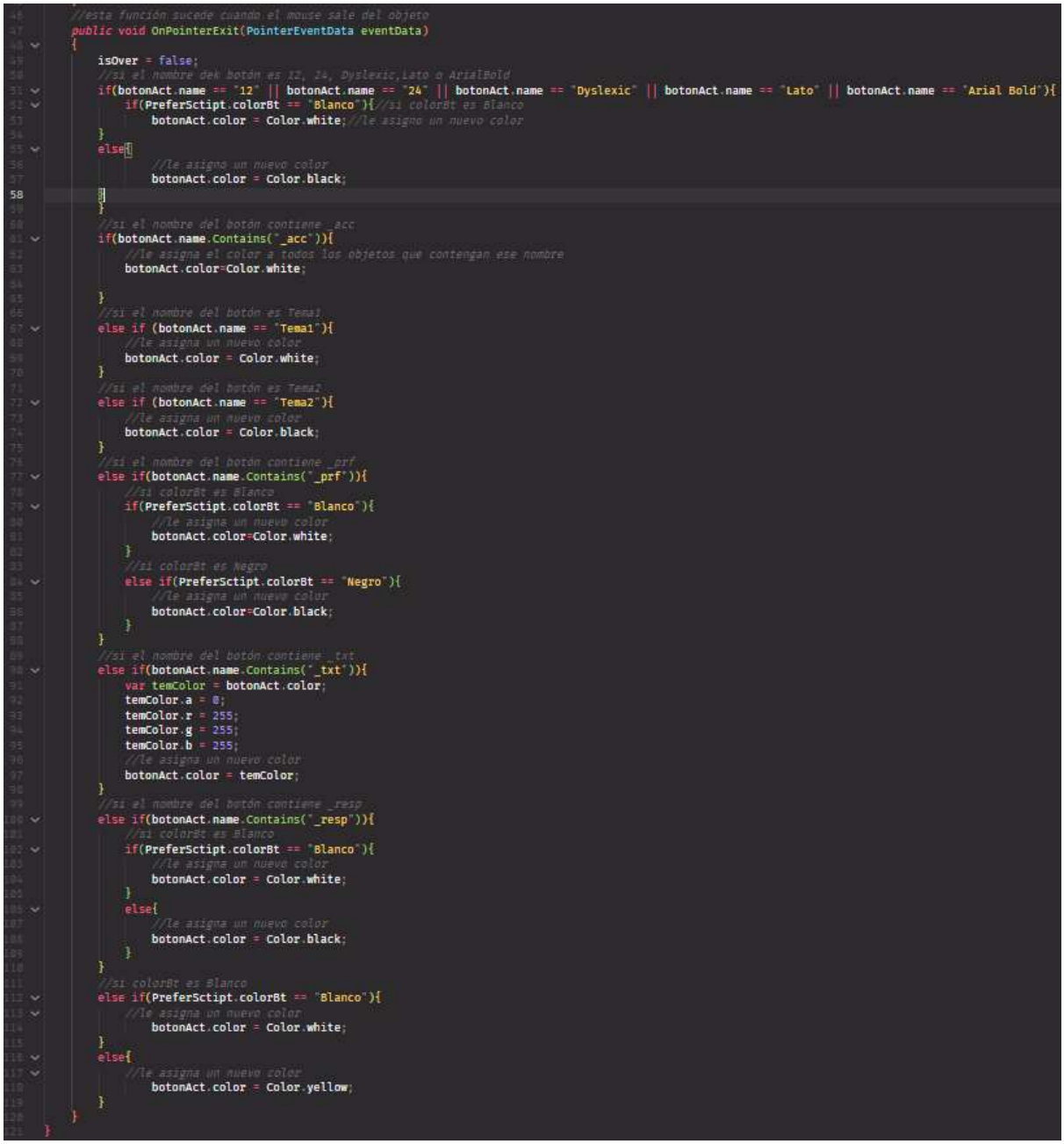
```
public class MouseColor : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    //declaro esta variable en false, para que solo se active cuando pase el mouse sobre el objeto
    public bool isOver = false;
    public Image botonAct;//declaro una variable de tipo Image
```

Figura 261. Declaración de variables.

Figura 262. Asignamos un botón a la variable BotonAct.

```
//esta función sucede cuando el mouse esta sobre el objeto
public void OnPointerEnter(PointerEventData eventData)
{
    isOver = true;
    //si colorBt es Blanco
    if(PreferSctipt.colorBt == "Blanco"){
        //accedo a la propiedad color del botón
        var temColor = botonAct.color;
        temColor.a = 0.5f;
        temColor.r = 0.52f;
        temColor.g = 0.5f;
        temColor.b = 0.5f;
        botonAct.color = temColor;//le asigna un nuevo color
    }
    //si el nombre del botón es Tema1
    else if (botonAct.name == "Tema1"){
        var temColor = botonAct.color;
        temColor.a = 0.5f;
        temColor.r = 0.52f;
        temColor.g = 0.5f;
        temColor.b = 0.5f;
        botonAct.color = temColor;//le asigna un nuevo color
    }
    else{
        //accedo a la propiedad color del botón
        var temColor = botonAct.color;
        temColor.a = 0.75f;
        temColor.r = 0.52f;
        temColor.g = 0.5f;
        temColor.b = 0.5f;
        botonAct.color = temColor;
    }
}
```

Figura 263. Función sucede cuando el mouse está sobre el objeto.



```
    //esta función sucede cuando el mouse sale del objeto
    public void OnPointerExit(PointerEventData eventData)
    {
        isOver = false;
        //si el nombre del botón es 12, 14, Dyslexic,Lato o ArialBold
        if(botonAct.name == "12" || botonAct.name == "24" || botonAct.name == "Dyslexic" || botonAct.name == "Lato" || botonAct.name == "Arial Bold"){
            if(PreferSctipt.colorBt == "Blanco"){//si colorBt es Blanco
                botonAct.color = Color.white;//le asigno un nuevo color
            }
            else{
                    //le asigno un nuevo color
                    botonAct.color = Color.black;
            }
        }
        //si el nombre del botón contiene _acc
        if(botonAct.name.Contains("_acc")){
            //le asigna el color a todos los objetos que contengan ese nombre
            botonAct.color=Color.white;

        }
        //si el nombre del botón es Tema1
        else if (botonAct.name == "Tema1"){
            //le asigna un nuevo color
            botonAct.color = Color.white;
        }
        //si el nombre del botón es Tema2
        else if (botonAct.name == "Tema2"){
            //le asigna un nuevo color
            botonAct.color = Color.black;
        }
        //si el nombre del botón contiene _prf
        else if(botonAct.name.Contains("_prf")){
            //si colorBt es Blanco
            if(PreferSctipt.colorBt == "Blanco"){
                //le asigna un nuevo color
                botonAct.color=Color.white;
            }
            //si colorBt es Negro
            else if(PreferSctipt.colorBt == "Negro"){
                //le asigna un nuevo color
                botonAct.color=Color.black;
            }
        }
        //si el nombre del botón contiene _txt
        else if(botonAct.name.Contains("_txt")){
            var temColor = botonAct.color;
            temColor.a = 0;
            temColor.r = 255;
            temColor.g = 255;
            temColor.b = 255;
            //le asigna un nuevo color
            botonAct.color = temColor;
        }
        //si el nombre del botón contiene _resp
        else if(botonAct.name.Contains("_resp")){
            //si colorBt es Blanco
            if(PreferSctipt.colorBt == "Blanco"){
                //le asigna un nuevo color
                botonAct.color = Color.white;
            }
            else{
                //le asigna un nuevo color
                botonAct.color = Color.black;
            }
        }
        //si colorBt es Blanco
        else if(PreferSctipt.colorBt == "Blanco"){
            //le asigna un nuevo color
                botonAct.color = Color.white;
        }
        else{
            //le asigna un nuevo color
                botonAct.color = Color.yellow;
        }
    }
}
```

Figura 264. Esta función sucede cuando el mouse sale del objeto.

1.4.6 Script PreferSctipt.cs

El siguiente script contiene opciones de preferencia con las cuales el participante puede personalizar el simulador laboral de acuerdo a sus necesidades y capacidades: para las opciones de contraste se emplea tres funciones las cuales trabajan en conjunto (*Cambio(), CambioColBot(), CambioColText()*) con las cuales podrá cambiar el color de fondo de la pantalla, el color de los botones y el color del texto; para la opción de activar o desactivar la

reproducción de audios se implementa la función *MuteAll()*, para la opción de tipos de fuentes se implementa la función *cambioFuente()* para ello se tiene tres opciones de fuentes; para la opción de aumentar o reducir el tamaño de texto se implementa la función *cambioTamaño()*.

1.4.3.1 Declaración de variables y relación con objetos de interfaz de Unity

Figura 265. Fragmento de código de declaración de variables.

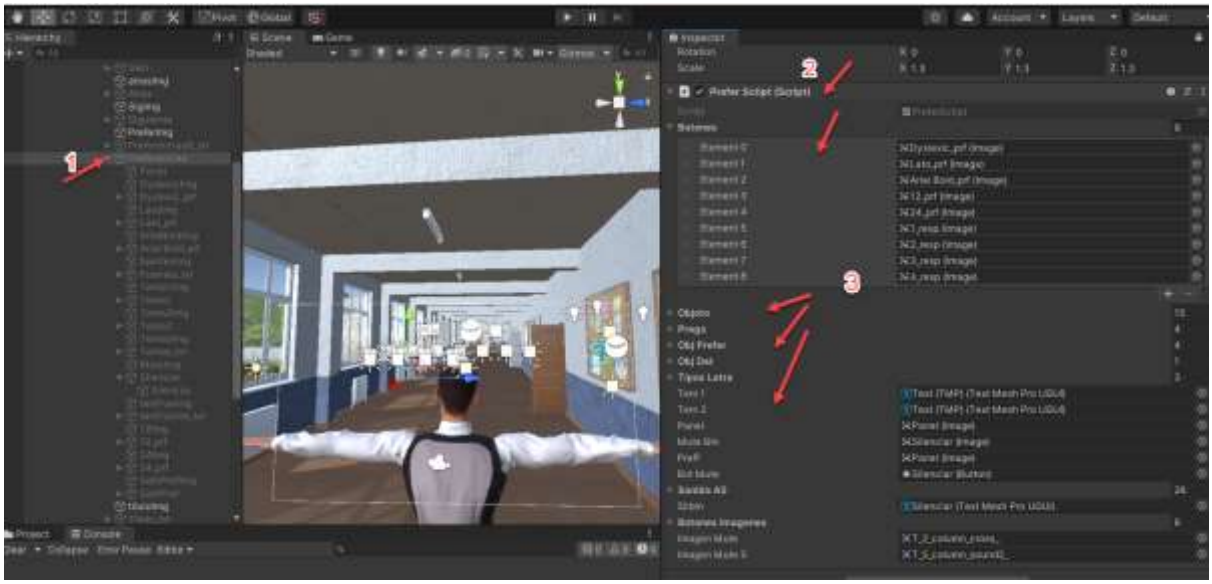


Figura 266. Asociamos los componentes con las variables declaradas.

```
/*Esta función recibe como parámetro un string en el cual indicamos el color del panel*/
    public void Cambio(string colpan){
        if (colpan == "Blanco"){//si el string que recibe es igual a Blanco
            panel.color = new Color(255,255,255,100);//cambiamos el color del panel a blanco
            preff.color=new Color(50,50,50,150);//cambiamos el color del panel de preferencias a oscuro.
        }//Y si el string que recibe es igual a Negro
        else if (colpan == "Negro"){
            panel.color = new Color(0,0,0,215);//cambiamos el color del panel a negro
            preff.color=Color.grey;//cambiamos el color del panel de preferencias a gris.
        }
    }
```

Figura 267. Función para cambiar el color del panel.

Las funciones *Cambio(), CambioColText() y CambioColBot()* son llamadas al presionar el botón tema1, para ello agregamos la propiedad OnClick() a nuestro botón y agregamos las funciones necesarias a las cuales les pasamos los parámetros que se necesita.

Figura 268. Combinación de funciones en donde se tiene como resultado un contraste claro.

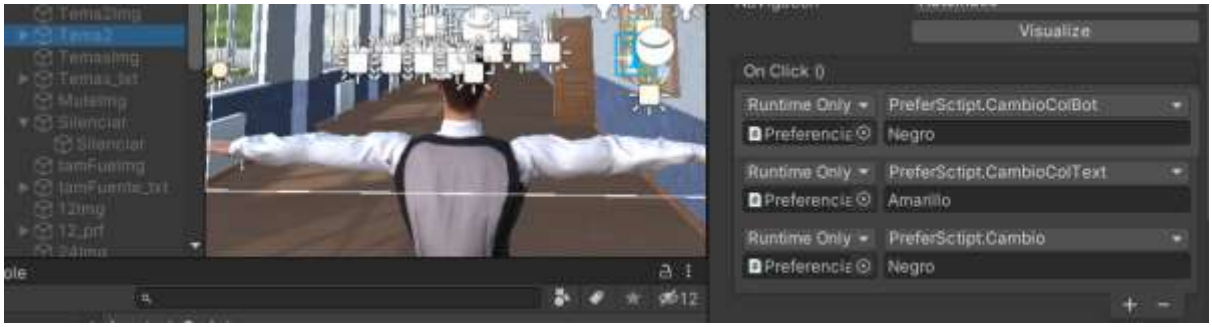


Figura 269. Combinación de funciones en donde se tiene como resultado un contraste obscuro.

```
/*Funcion que permite sileciar la salida de audio del simulador*/
    public void MuteAll(){
        //si la variable estatica es igual a false y colorBt es blanco
        if(isMute == false && PreferSctipt.colorBt == "Blanco"){
            //mandamos a sileciar todos los audios.
            foreach (var s in sonidoAS){
                s.volume = 0f;
            }
            isMute = true;
            //si esta desactivado el audio asignamos la imagen ImagenMute al botón
            botMute.image.sprite = ImagenMute;
            muteBtn.color = Color.yellow;//cambiamos a amarillo
        }
        //si la variable estatica es igual a false y colorBt es Negro
        else if(isMute == false && PreferSctipt.colorBt == "Negro"){
            //mandamos a sileciar todos los audios.
            foreach (var s in sonidoAS){
                s.volume = 0f;
            }
            isMute = true;
            botMute.image.sprite = ImagenMute;
            muteBtn.color = Color.white;
        }
        else {//caso contrario
            //mandamos a activar la salida de audios.
            foreach (var s in sonidoAS){
                s.volume = 1f;
            }
            isMute = false;
            //Si colorBt es blanco
            if(PreferSctipt.colorBt == "Blanco"){
                //cambiamos la imagen
                botMute.image.sprite = ImagenMuteS;
                //cambiamos a blanco
                muteBtn.color = Color.white;
            }
            else{
                botMute.image.sprite = ImagenMuteS;
                //caso contrario lo cambia a amarillo
                muteBtn.color = Color.yellow;
            }

        }
    }
```

Figura 270. Función para silenciar la salida de audios.

Figura 271. Asociamos a botón la función que permite silenciar los audios.

Función llamada desde el botón tema1 y tema2.

```
/*Función que recibe como parametro un string dicha función permite cambiar el color de los botones*/
    public void CambioColBot(string te){
        //si el string que recibe es igual a Blanco
        if (te == "Blanco"){
            //cambia todos los botones a blanco
            foreach(var b in botones){
                b.color = Color.white;
                PreferSctipt.ColB("Blanco");
            }
            foreach(var b in botonesImagenes){
                b.color = Color.white;
            }
            foreach(var b in pregs){
                b.color = Color.white;
            }
        }
        //si el string que recibe es igual a Negro
        else if (te == "Negro"){
            //cambia todos los botones a negro
            foreach(var b in botones){
                b.color = Color.black;
                PreferSctipt.ColB("Negro");
            }
            //los botones principales los cambiamos a amarillo
            foreach(var b in botonesImagenes){
                b.color = Color.yellow;
            }
        }
    }
```

Figura 272. Función para cambiar el color de los botones.

La siguiente función es llamada al presionar el botón tema1 y tema2.

```
/*Esta funcion recibe como parametro un string dicha función permite cambiar el color del texto*/
    public void CambioColText(string te){
        //si el string que recibe es igual a Negro
        if (te == "Negro"){
            /*cambia la letra a azul*/
            foreach (var obj in objeto) {
                obj.color= Color.blue;
            }
            foreach (var obj in tiposLetra) {
                obj.color= Color.blue;
            }
            foreach (var obj in pregs) {
                obj.color= Color.blue;
            }
            foreach (var obj in objPrefer) {
                obj.color= Color.blue;
            }
        }
        //si el string que recibe es igual a Amarillo
        if (te == "Amarillo"){
            /*cambia la letra a amarillo*/
            foreach (var obj in objeto) {
                obj.color = Color.yellow;
            }
            foreach (var obj in tiposLetra) {
                obj.color = Color.yellow;
            }
            foreach (var obj in pregs) {
                obj.color= Color.yellow;
            }
            foreach (var obj in objPrefer) {
                obj.color=Color.yellow;
            }
        }
    }

```

Figura 273. Función para cambiar el color del texto.

La siguiente función es llamada al presionar los botones para cambiar el tipo de fuente(Dyslexic, Lato y Arial Bold).

```
/*Esta función permite recibe como parámetro un string el cual se le pasa la fuente, esta función
permite personalizar el texto de la interfaz haciendo uso de tres tipos de fuente de texto*/
    public void cambioFuente(TMP_FontAsset fu){
        //si la fuente que recibe es igual a
        if(fu.ToString() == "OpenDyslexic-Regular SDF (TMPro.TMP_FontAsset)"){
            /*se define un tamaño para cuando esté activa esta fuente.*/
            foreach(var t in objeto){
                t.fontSize = 19f;
            }
            foreach(var t in pregs){
                t.fontSize = (18f);
            }
            foreach(var t in objPrefer){
                t.fontSize = (17f);
            }

            Disle=true;//si está activa
        }
        else{
            Disle=false;// si esta desactivada
        }
            /*cambia de fuente a todos los textos de la interfaz*/
            foreach(var t in objeto){
                t.font = fu;
            }
            foreach(var t in pregs){
                t.font = fu;
            }
            foreach(var t in objPrefer){
                t.font = fu;
            }
            //cambia el tipo de fuente
            tem1.font = fu;
            tem2.font = fu;
    }
```

*Figura 274. F*unción para cambiar el tipo de fuente.

Figura 275. Función que permite cambiar el tipo de fuente.

La siguiente función es llamada con los botones de aumentar y disminuir el tamaño de texto.

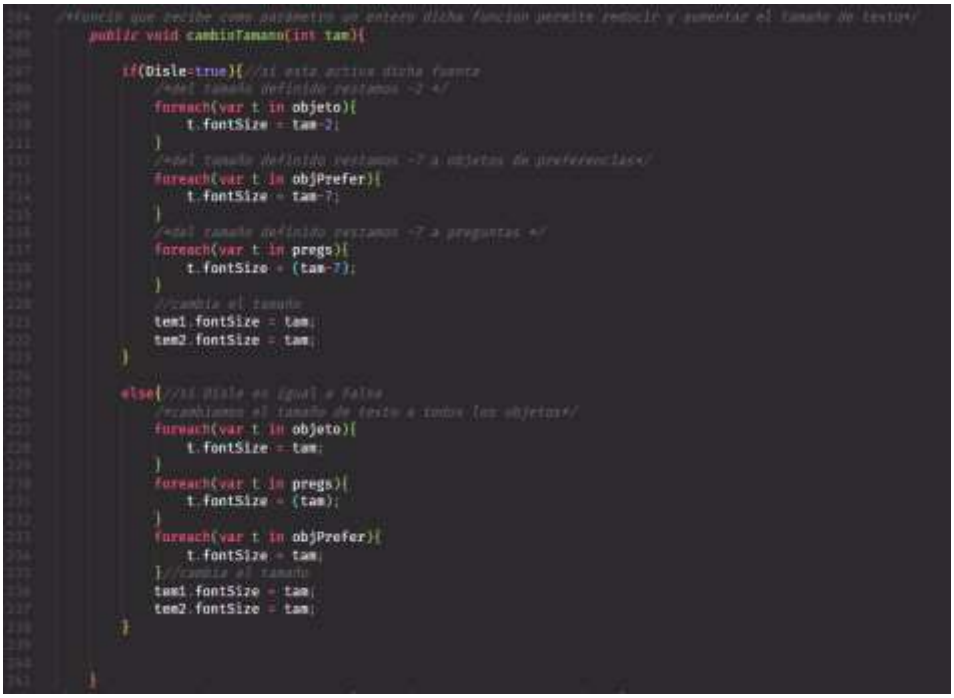


Figura 276. Función para cambiar aumentar y disminuir el tamaño de texto.

Figura 277. Funciona que permite cambiar el tamaño de texto.

Esta función es llamada al presionar el botón de preferencias.

```
/*Función que permite detener todos los sonidos cuando se descative el audio y
así evitar que algun audio se reproduzca hasta finalizar*/
    public void DetengoSonidos(){
        foreach(var t in sonidoAS){
            t.Stop();
        }
    }
```

Figura 278. Función para detener la salida de audio inmediatamente.

```
/*Funcion que recibe como parametro un string*/
    public static void MenuSel(string tipoMen){
        //controlamos en menu estamos
        PreferSctipt.menu = tipoMen;
    }
```

Figura 279. Función que ayuda a controlar el menú en el que estamos.

```
/*Esta función se utiliza para saber que color de botones y panel se esta asignando
y de esa manera combinar los colores de manera correcta*/
    public static void ColB(string col){
        PreferSctipt.colorBt = col;
    }
}
```

Figura 280. Esta función se usa para saber el color del panel y botones que se está asignando.

1.4.7. Script WebData.cs

Se crea la función *Post()* para consumir el servicio web, pasamos la url del servidor web indicando el tipo de método en este caso POST ya que se va a insertar datos en el servidor web, generamos el archivo json de mis preguntas, envío el json y finalmente hago la petición al servidor web mediante SendWebRequest().

```
using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {


    [DllImport("__Internal")]
    private static extern int GetUrlVal();

    [DllImport("__Internal")]
    private static extern int GetEjer();

    [DllImport("__Internal")]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {

    }
    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text="-- "+id;
        return  id;
    }

    public string Mail(){
        string mailV = GetMail();
        Debug.Log("Mail - Un: "+mailV);
        idMail.text="- "+mailV;
        return mailV;
    }

    public int Ejercitario(){
        int ejerV = GetEjer();
        Debug.Log("Ejer - Un: "+ejerV);
        idEjer.text="-- "+ejerV;
        return ejerV;

    }
}
```

Figura 281. Script WebData para los servicios web.

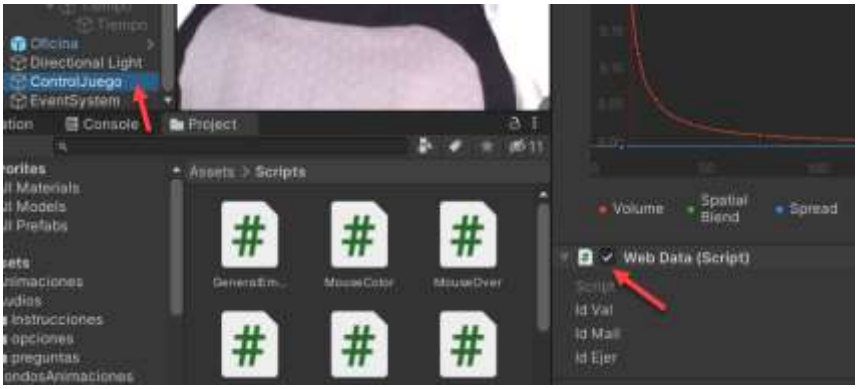


Figura 282. Añadimos el script WebData al proyecto, lo arrastramos al GameObject ControlJuego.

1.5. Programación Simulador Laboral El Tiempo

1.5.1. Script GeneralEmpezar.cs

```
using System;
using System.Collections;
using System.Collections.Generic;//libreria para manipular con listas
using UnityEngine;
using System.Linq;//libreria para realizar consultas
using System.Text;//libreria para manipular texto
using UnityEngine.UI;// libreria para acceder a las propiedades de los elementos de la interfaz de usuario.
using System.IO;//libreria para escribir y leer datos.
using System.Threading.Tasks;
using UnityEngine.EventSystems;
using TMPro; //importamos la libreria TextMeshPro.
using Random = UnityEngine.Random;
using UnityEngine.Networking;//importamos la libreria para la comunicacion con el servidor web.
```

Figura 283. Librerías utilizadas.

1.5.1.1 Declaración de variables y relación con objetos de interfaz de Unity

Figura 284. Declaración de variables parte 1.

Para la navegación por teclado se declararon arreglos seleccionables, arreglos de imágenes y de audios, por tanto, es importante recalcar que el orden en que se cargue los objetos en cada arreglo será el orden que se seleccione al presionar el teclado, es decir si al presionar el teclado se activa el botón en la posición 0 entonces también se activará la imagen y sonido en la posición 0.

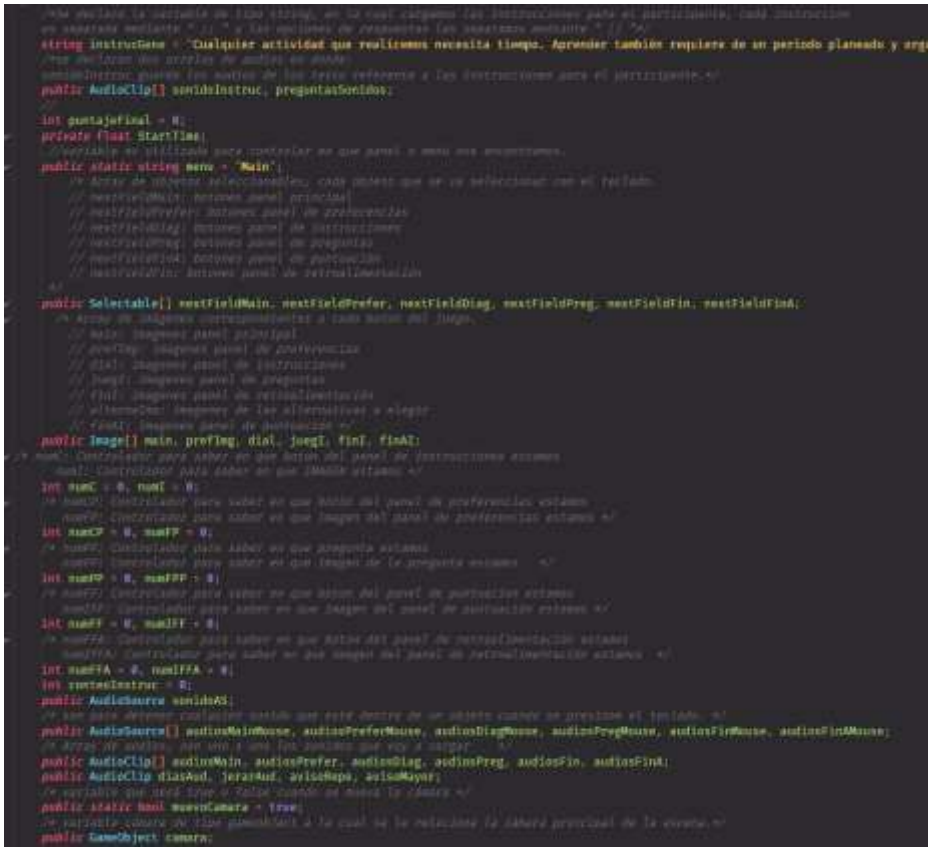


Figura 285. Declaración de variables parte 2.

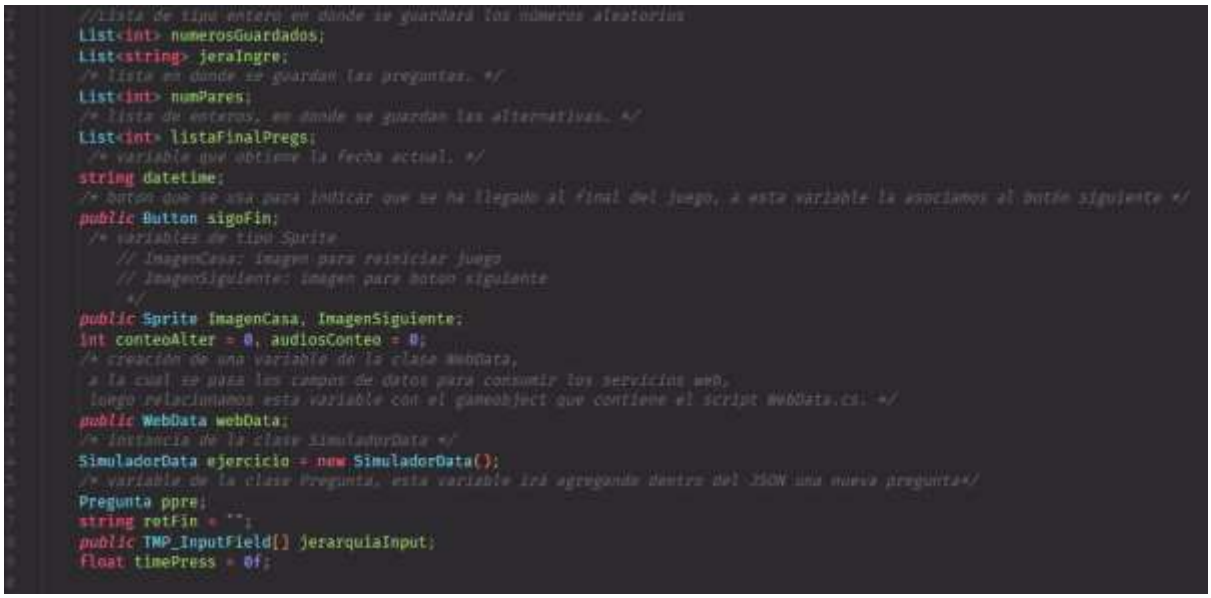


Figura 286. Declaración de variables parte 3.

Desde la ventana de jerarquía en el GameObject ControlJuego una vez posicionados aquí nos dirigimos a la ventana inspector y procedemos a relacionar los objetos con cada variable.

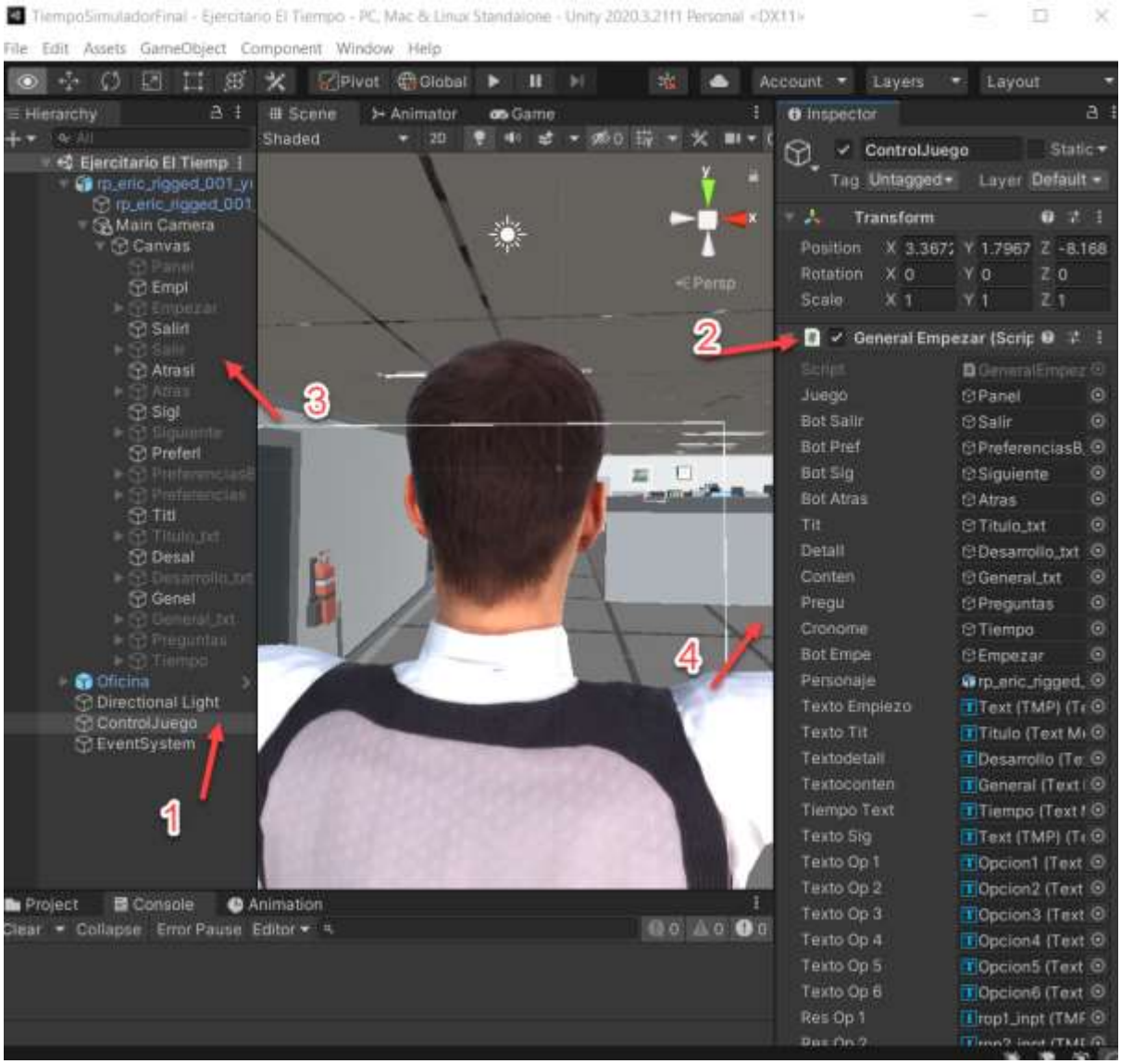


Figura 287. Relacionamos cada gamobject de la escena con la variable determinada.

1.5.1.2 Declaración de funciones

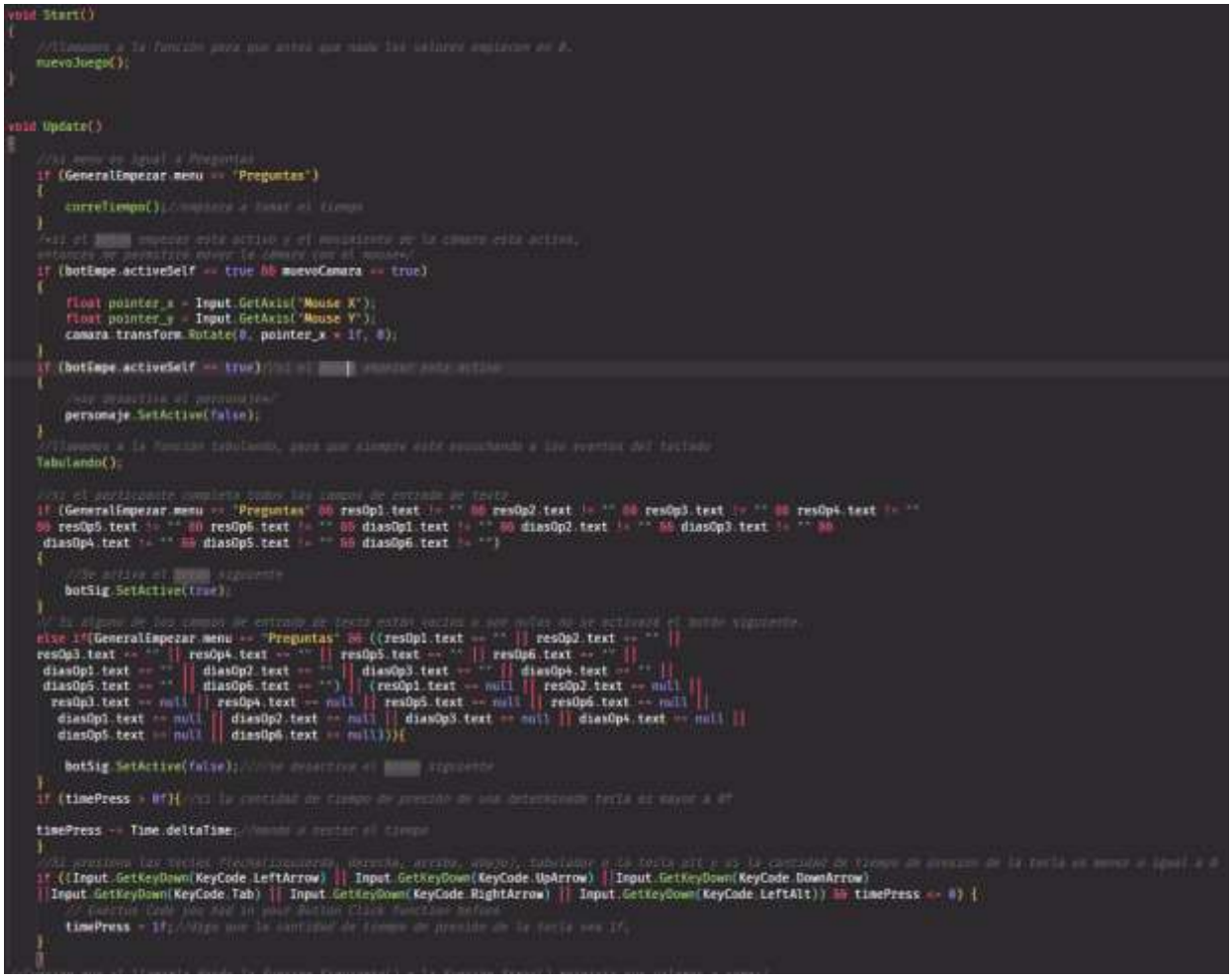


Figura 288. Funciones Start() y Update().

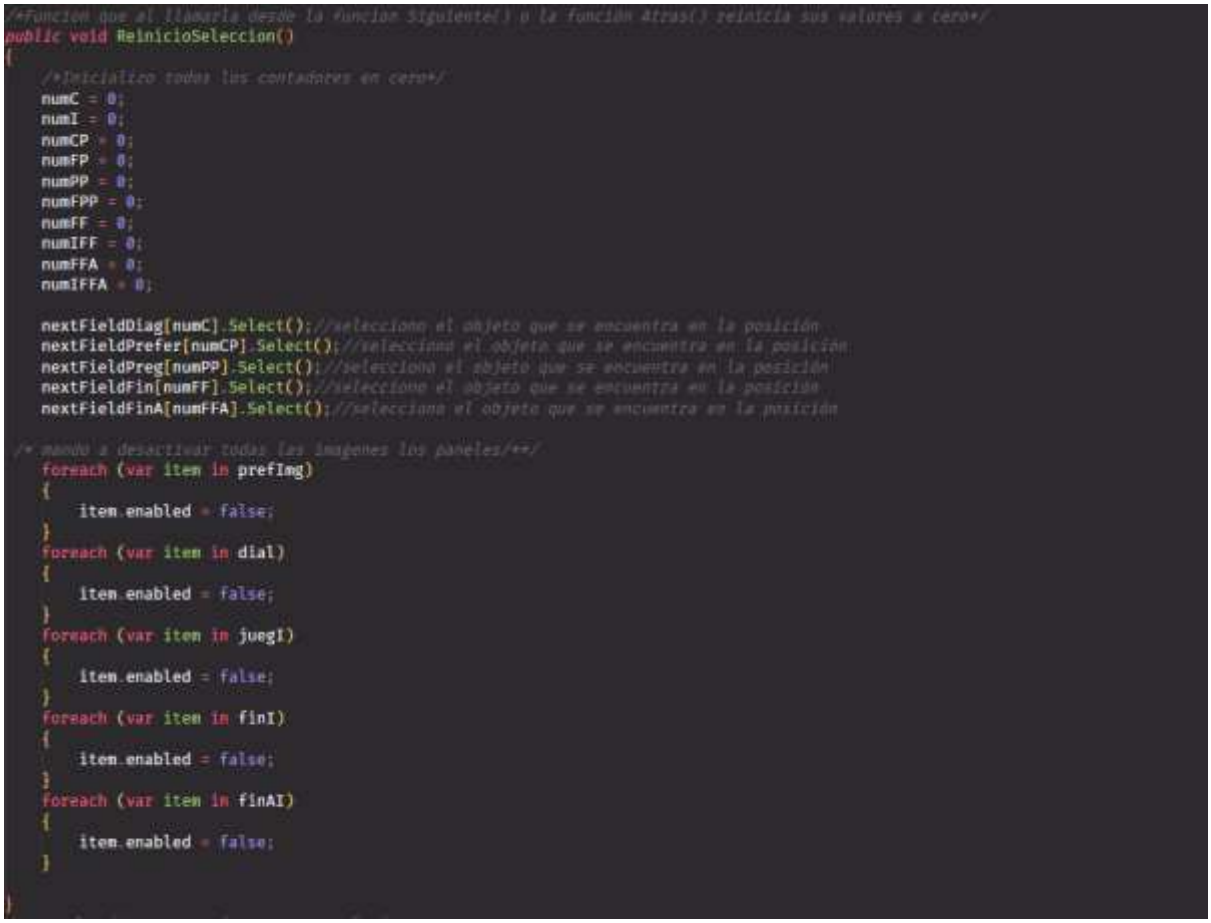


Figura 289. Función ReinicioSelección().

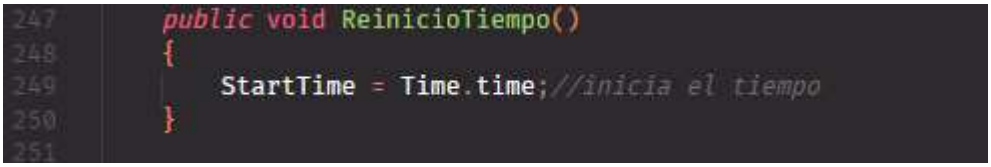


Figura 290. Función ReinicioTiempo().

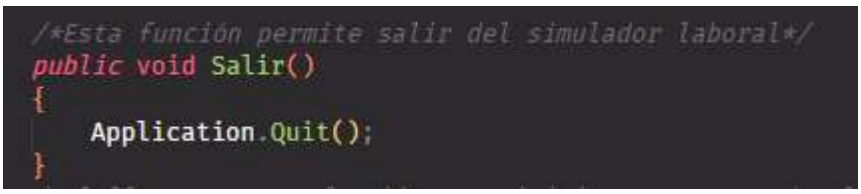


Figura 291. Función Salir().

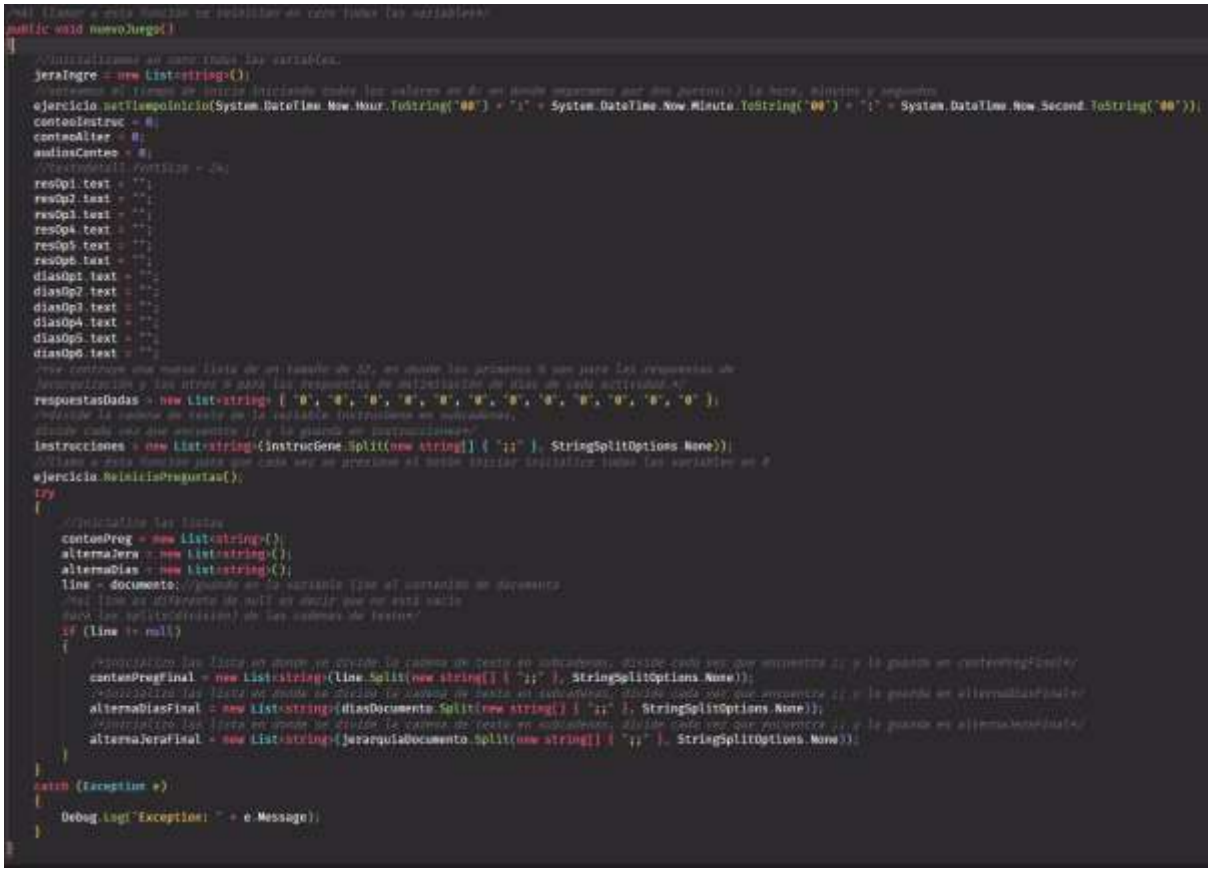


Figura 292. Función nuevoJuego().

La función *Tabulando()* controla los eventos del teclado, es decir la navegación en la interfaz al presionar ciertas teclas, para lograrlo primero se declaran arreglos de tipo Selectable, luego asociamos con cada GameObject de la escena, luego agregamos a nuestra sentencia de código el método *Select()* de Unity el cual hace que se seleccione el componente requerido cada vez que se navegue con el teclado; para controlar en que parte del simulador se encuentra el participante se hace uso de la variable estática *menu*, ésta variable va cambiando de nombre en cada panel que nos encontremos, por ejemplo para indicar que estamos en el panel principal decimos que menu es igual a *Main*, cuando estamos en el panel de instrucciones menu es igual a *Jugando*, cuando pasemos al panel de preguntas menu es igual a *Preguntas*, cuando pasemos

al panel de calificación menu es igual a *FinA* y cuando pasemos al último panel menu es igual a *Fin*.

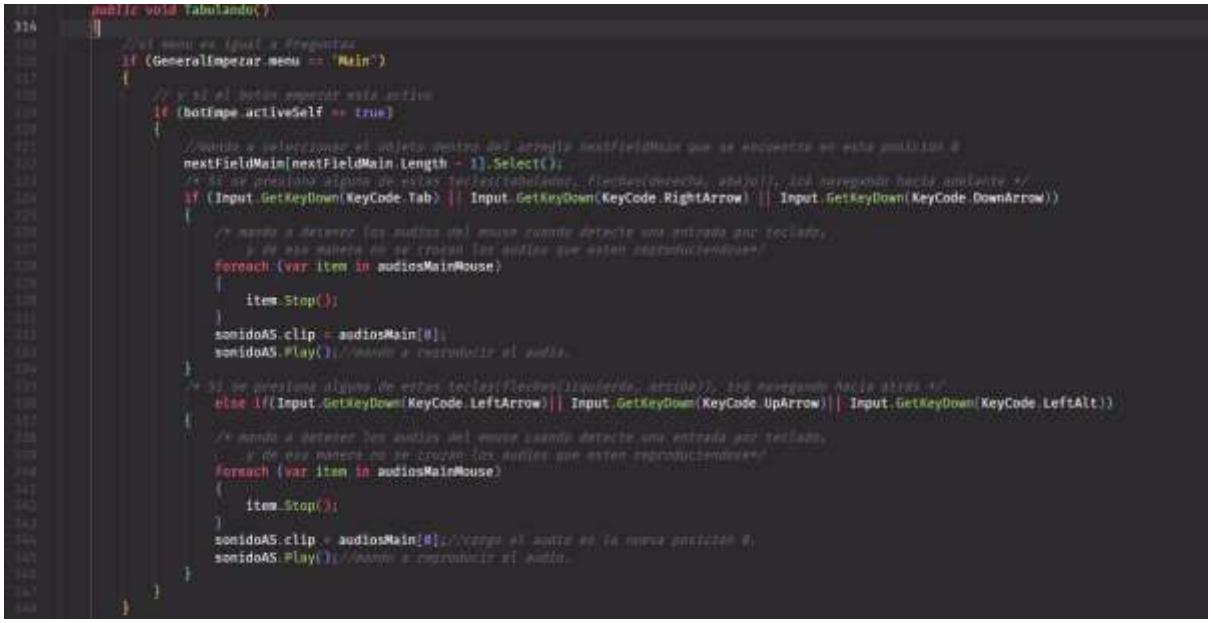


Figura 293. Control por teclado función Tabulando() parte 1.

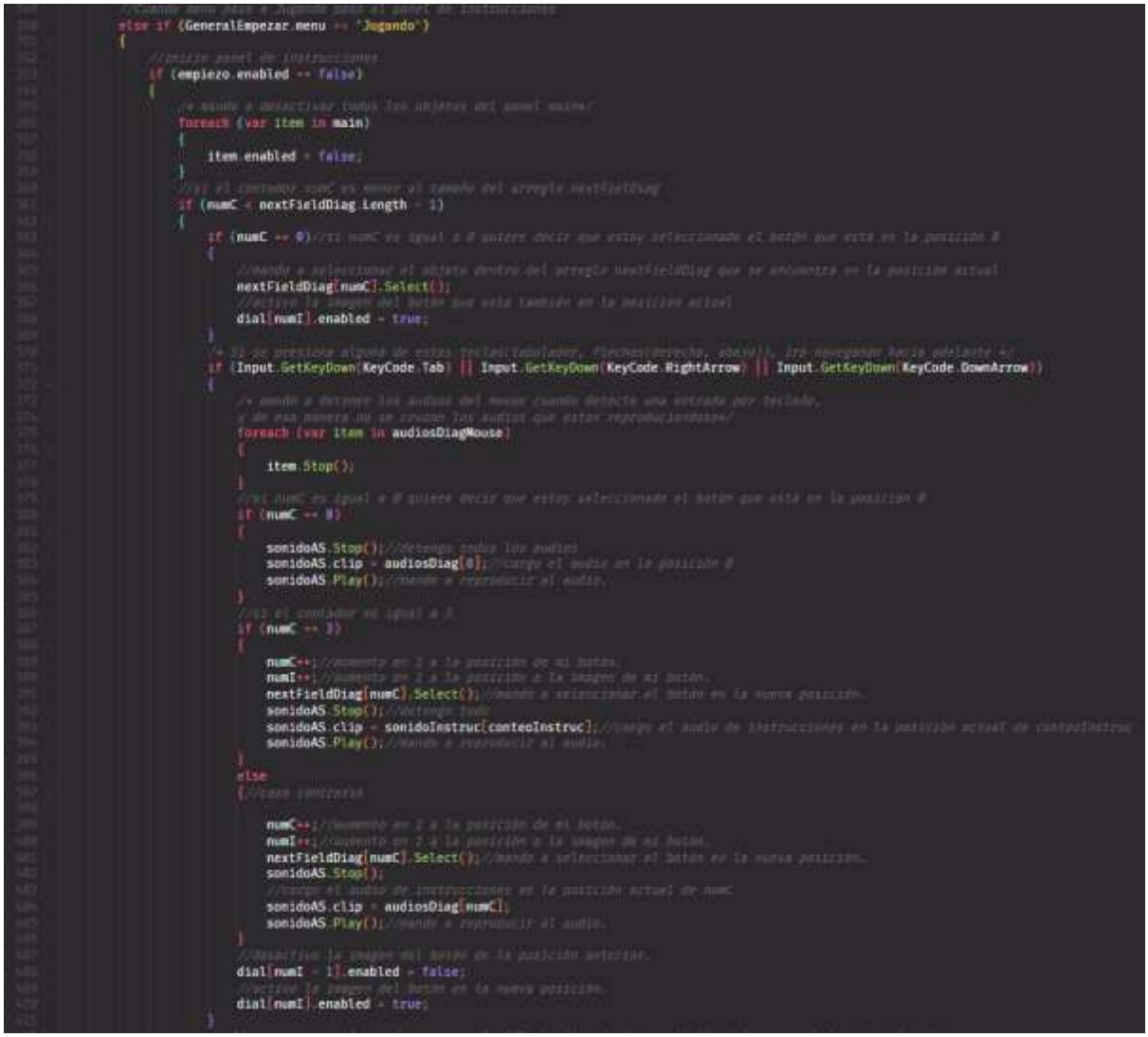


Figura 294. Control por teclado función Tabulando parte 2.

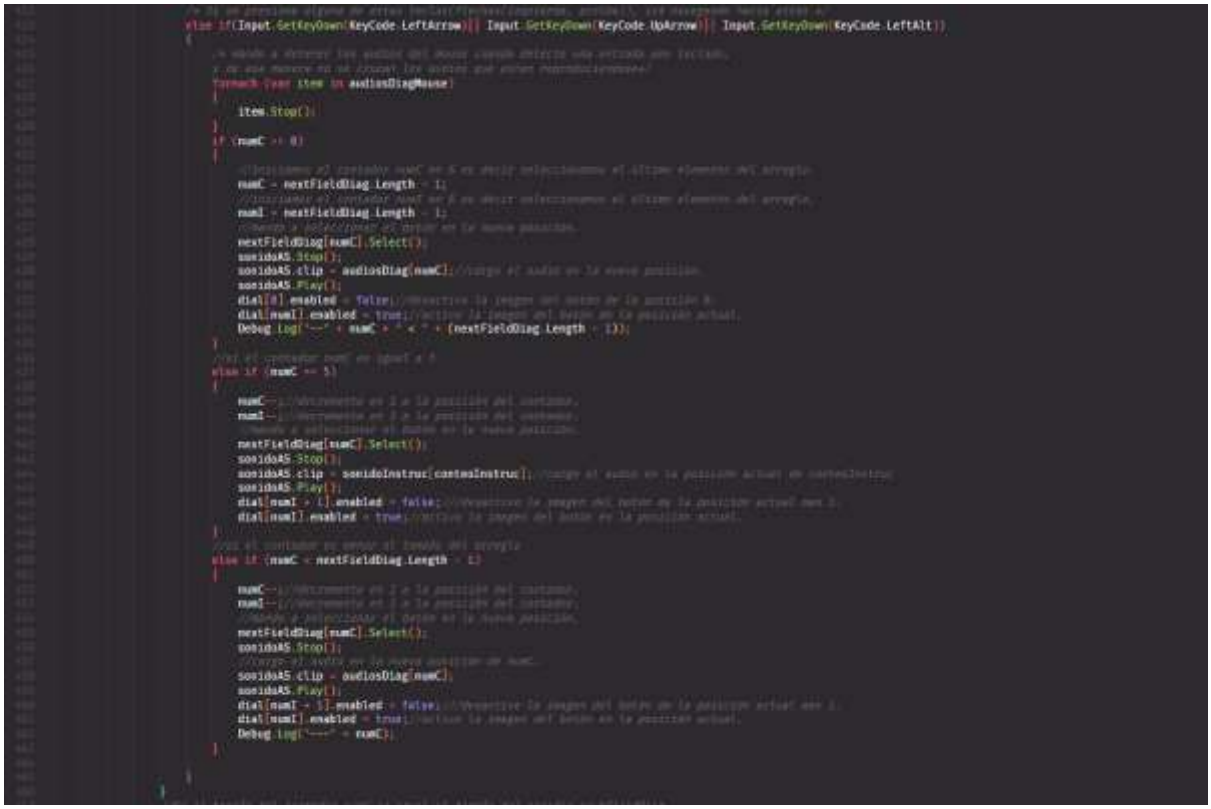


Figura 295. Control por teclado función Tabulando() parte 3.

```
        //Si el tamaño del contador numC es igual al tamaño del arreglo nextFieldDiag
        else if (numC == nextFieldDiag.Length - 1)
        {
            /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
            if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
            {
                /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                y de esa manera no se cruzan los audios que esten reproduciendose*/
                foreach (var item in audiosDiagMouse)
                {
                    item.Stop();
                }
                dial[numI].enabled = false;//entonces se desactiva el objeto que esta en esa posición.
                numC = 0;//se inicia todo en 0
                numI = 0;//se inicia todo en 0
                nextFieldDiag[numC].Select();//selecciono el boton en la posición actual.
                sonidoAS.Stop();//detengo todos los audios.
                sonidoAS.clip = audiosDiag[numC];//agrego el audio en la nueva posición.
                sonidoAS.Play();//mando a reproducir el audio
                dial[numI].enabled = true;//mando a activar la imagen del boton en la posición actual.
            }
            /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
            else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
            {
                numC--;//decremento en 1 a la posición del contador.
                numI--;//decremento en 1 a la posición del contador.
                /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                y de esa manera no se cruzan los audios que esten reproduciendose*/
                foreach (var item in audiosDiagMouse)
                {
                    item.Stop();
                }
                dial[numI+1].enabled = false;//desactivo la imagen del botón de la posición actual mas 1.
                nextFieldDiag[numC].Select();//selecciono el boton en la posición actual.
                sonidoAS.Stop();//detengo todos los audios que esten reproduciendose.
                sonidoAS.clip = audiosDiag[numC];//cargo el audio en la nueva posición.
                sonidoAS.Play();//mando a reproducir el audio.
                dial[numI].enabled = true;//activo la imagen del botón de la posición actual.
            }
        }
    }
```

Figura 296. Control por teclado función Tabulando() parte 4.

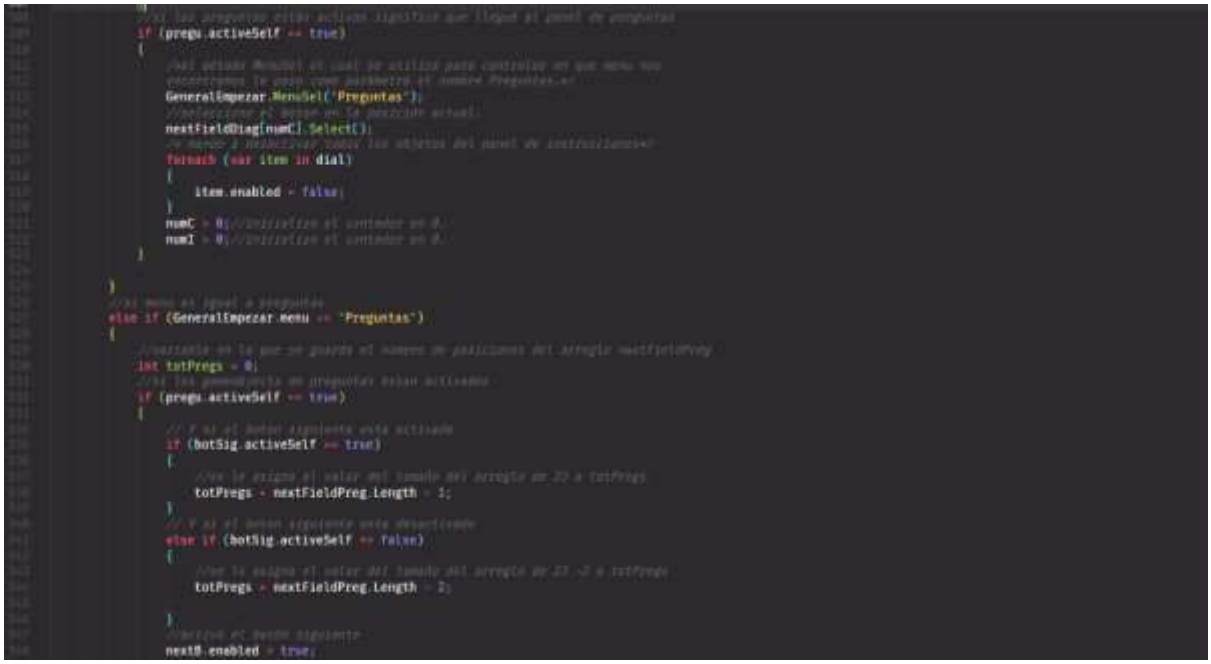


Figura 297. Control por teclado función Tabulando() parte 5.

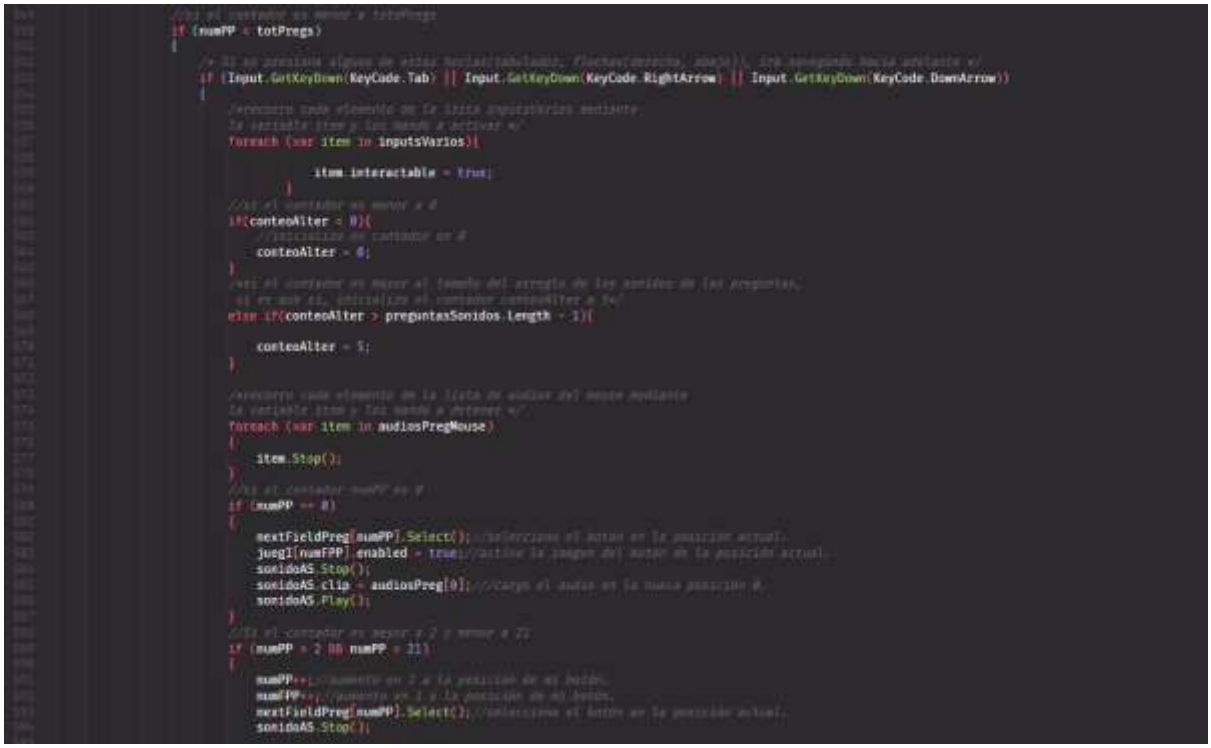


Figura 298. Control por teclado función Tabulando() parte 6.

Figura 299. Control por teclado función Tabulando() parte 7.

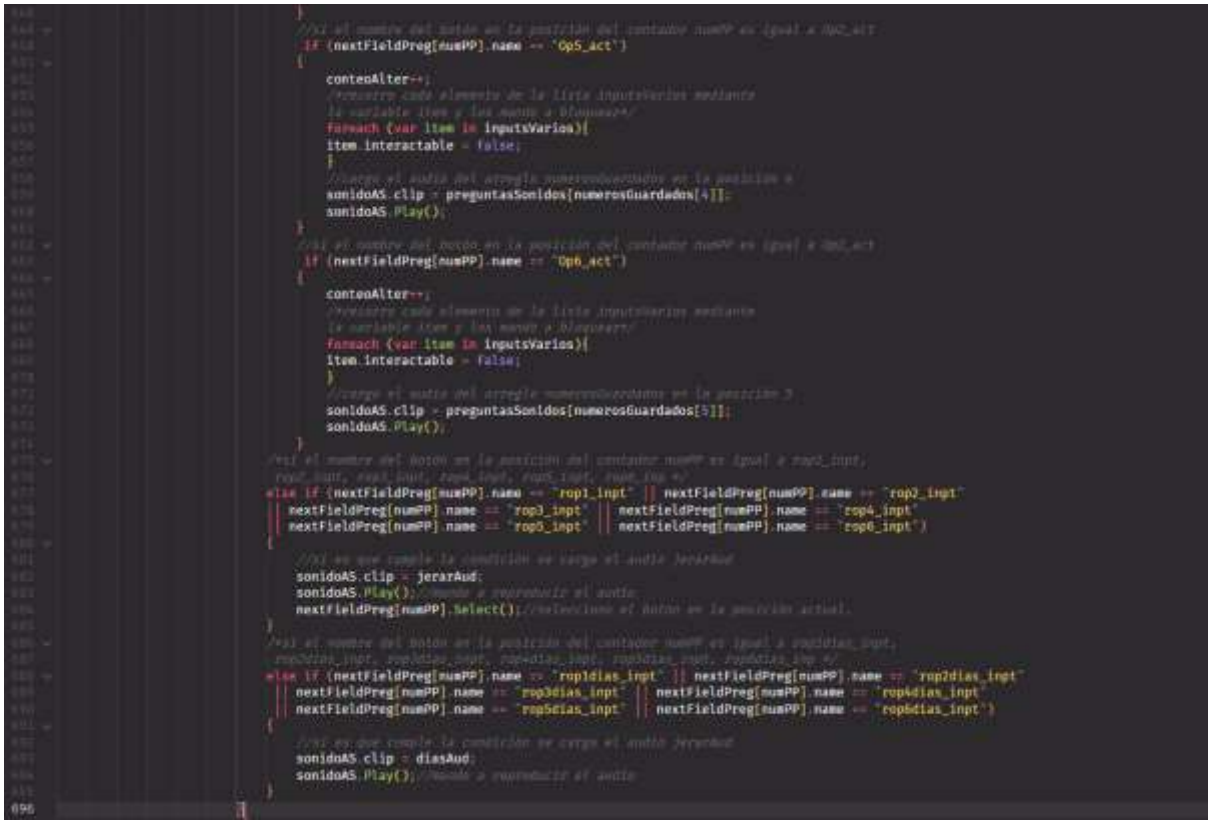


Figura 300. Control por teclado función Tabulando() parte 8.

```
else
{
    numPP++;//incrementa el contador en 1.
    numFPP++;//incrementa el contador en 1.
    nextFieldPreg[numPP].Select();//selecciona el boton en la posicion actual.
    sonidoAS.Stop();
    /*si el nombre del boton en la posicion del contador numPP es igual PreferenciaB_txt*/
    if (nextFieldPreg[numPP].name == "PreferenciaB_txt")
    {
        audiosConteo=1;//inicializa el contador en 1
        nextFieldPreg[numPP].Select();//selecciona el boton en la posicion actual.
        sonidoAS.clip = audiosPreg[1];//carga el audio del arreglo audiosPreg la posicion 1
        sonidoAS.Play();//manda a reproducir el audio

    }
    /*si el nombre del boton en la posicion del contador numPP es igual Salir_txt*/
    if (nextFieldPreg[numPP].name == "Salir")
    {
        audiosConteo=0;//inicializa el contador en 0
        nextFieldPreg[numPP].Select();/*  */
        //selecciona el boton en la posicion actual.
        sonidoAS.clip = audiosPreg[0];//carga el audio del arreglo audiosPreg la posicion 0
        sonidoAS.Play();//manda a reproducir el audio
    }
    /*si el nombre del boton en la posicion del contador numPP es igual Atras_txt*/
    if (nextFieldPreg[numPP].name == "Atras")
    {
        audiosConteo=4;
        nextFieldPreg[numPP].Select();//selecciona el boton en la posicion actual.
        sonidoAS.clip = audiosPreg[4];//carga el audio del arreglo audiosPreg la posicion 4
        sonidoAS.Play();


    }
    /*si el nombre del boton en la posicion del contador numPP es igual Titulo_txt*/
    if (nextFieldPreg[numPP].name == "Titulo_txt")
    {
        audiosConteo=2;
        nextFieldPreg[numPP].Select();//selecciona el boton en la posicion actual.
        sonidoAS.clip = audiosPreg[2];//carga el audio del arreglo audiosPreg la posicion 2
        sonidoAS.Play();
    }
    /*si el nombre del boton en la posicion del contador numPP es igual Desarrollo_txt*/
    if (nextFieldPreg[numPP].name == "Desarrollo_txt")
    {
        audiosConteo=3;
        nextFieldPreg[numPP].Select();//selecciona el boton en la posicion actual.
        sonidoAS.clip = audiosPreg[3];//carga el audio del arreglo audiosPreg la posicion 3
        sonidoAS.Play();
    }
```

Figura 301. Control por teclado función Tabulando() parte 9.

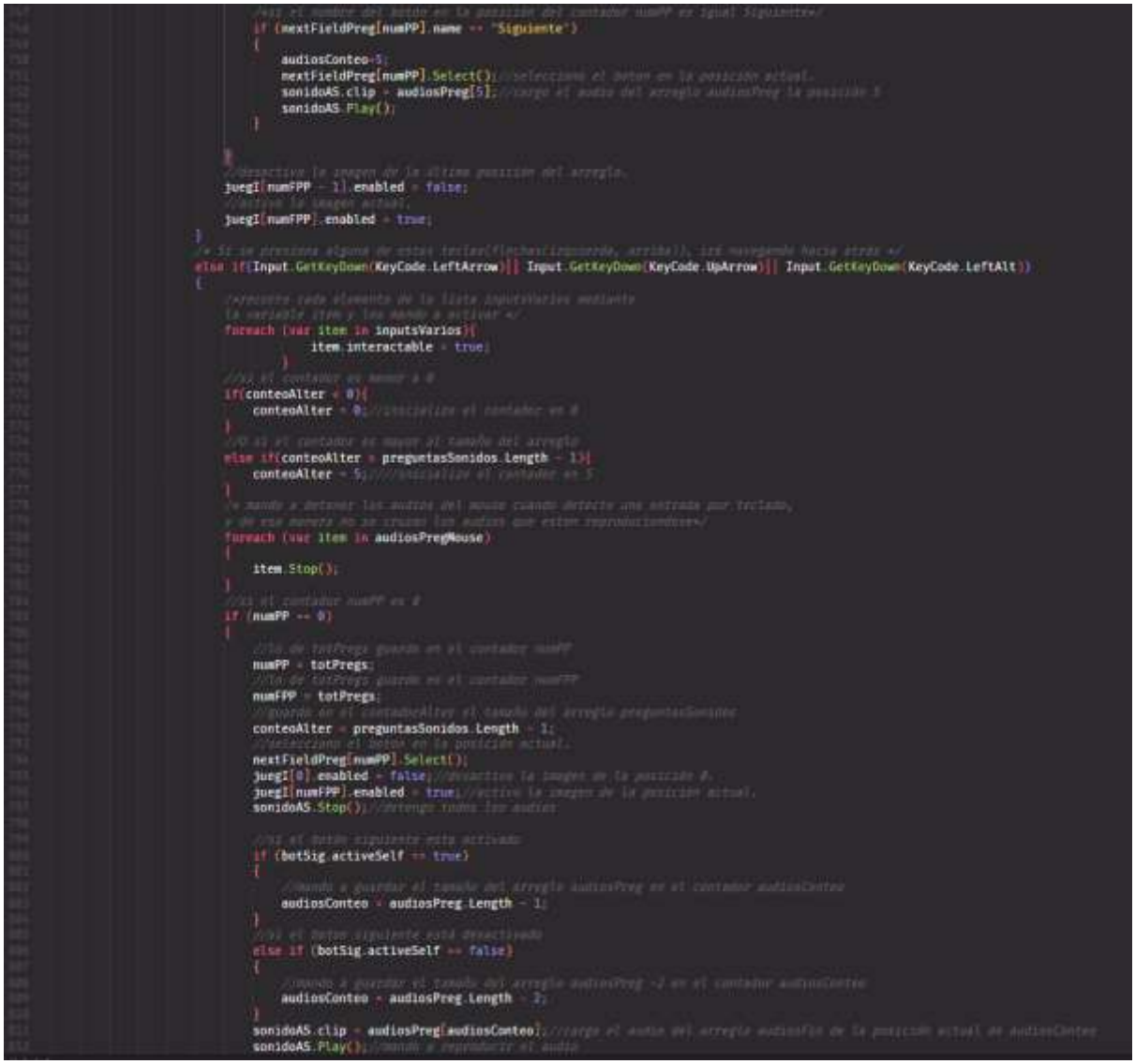
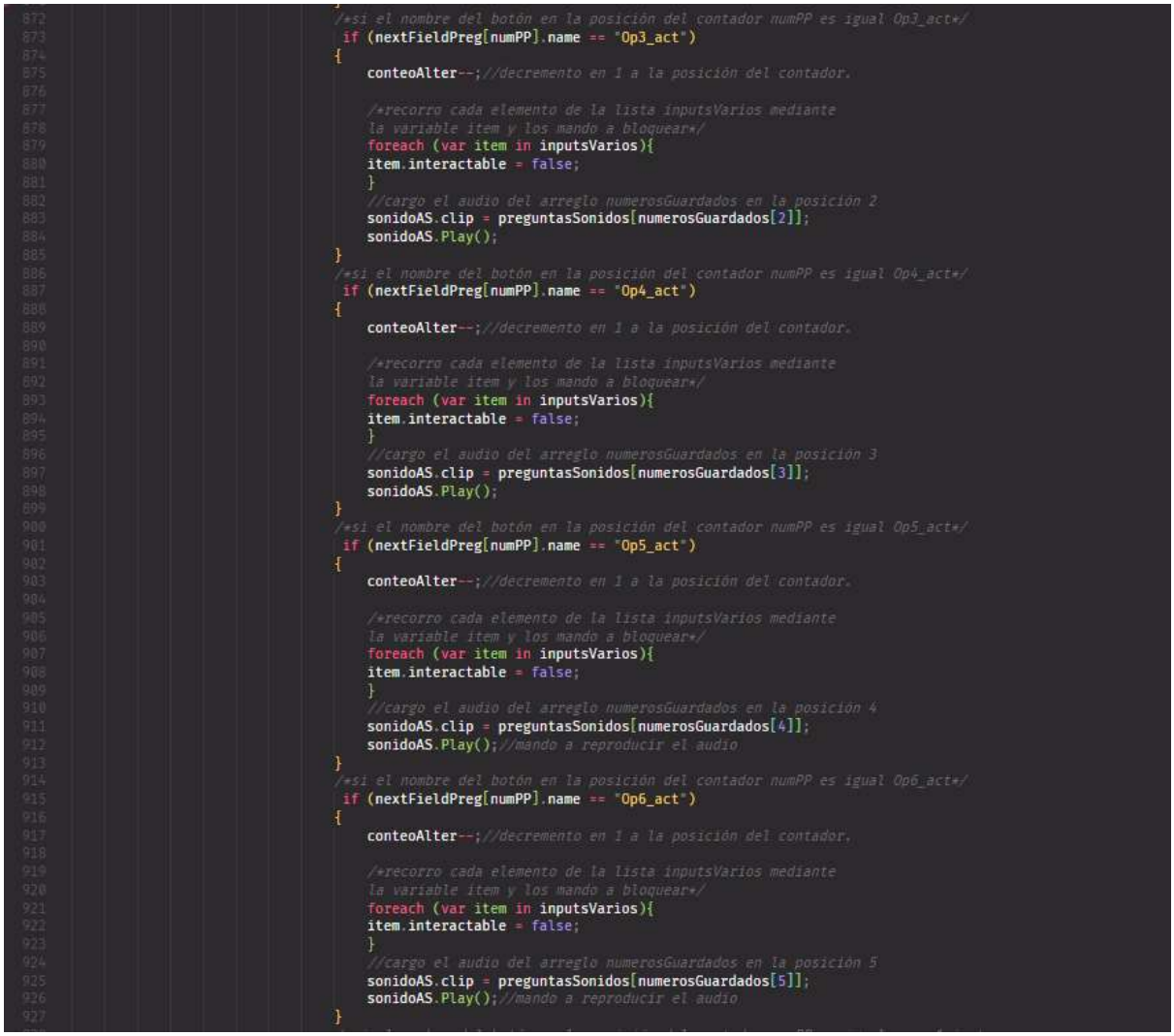


Figura 302. Control por teclado función Tabulando() parte 10.

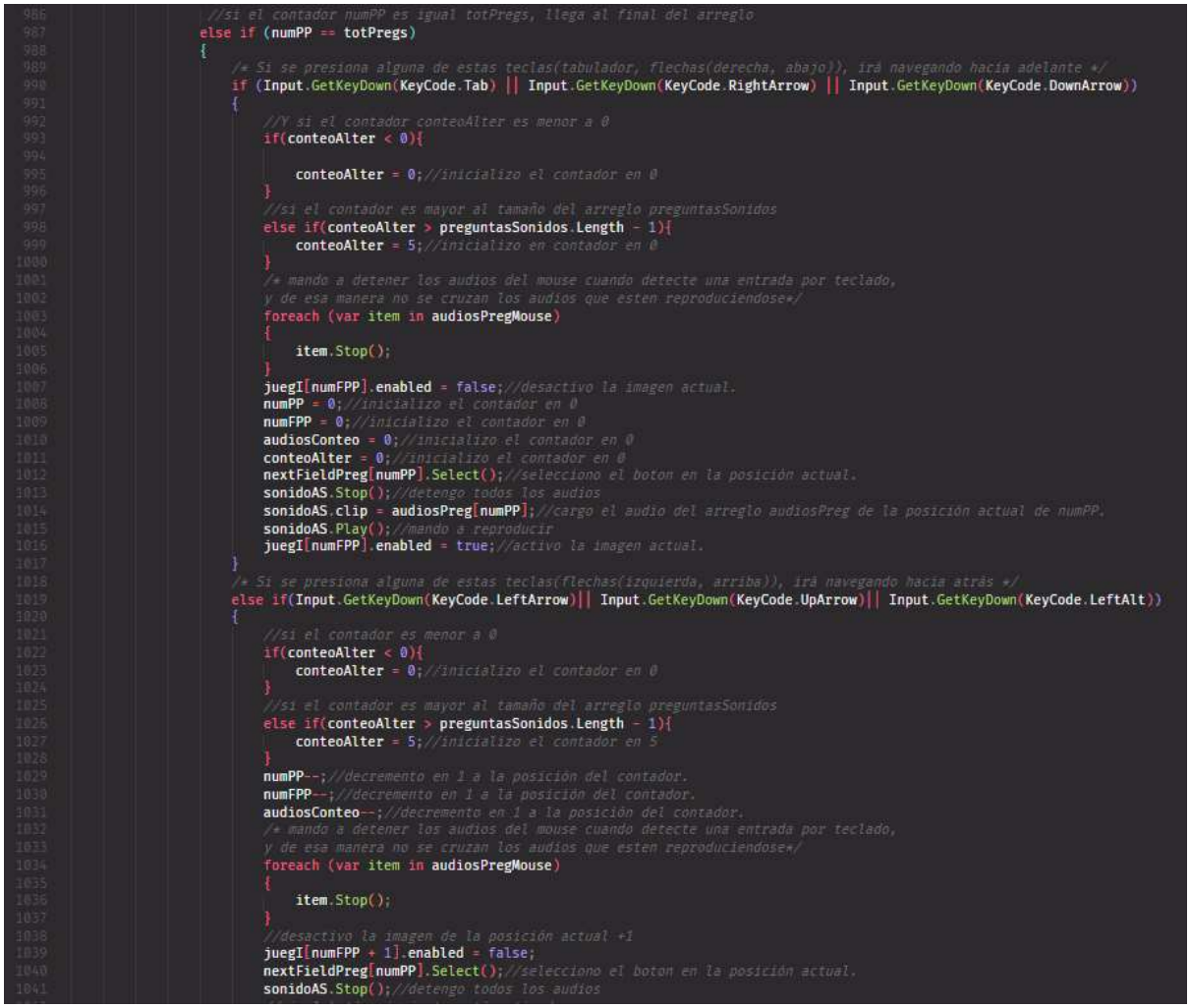
Figura 303. Control por teclado función Tabulando() parte 11.



```
/*si el nombre del botón en la posición del contador numPP es igual Op3_act*/
 if (nextFieldPreg[numPP].name == "Op3_act")
{
    conteoAlter--;//decremento en 1 a la posición del contador.

    /*recorro cada elemento de la lista inputsVarios mediante
    la variable item y los mando a bloquear*/
    foreach (var item in inputsVarios){
    item.interactable = false;
    }
    //cargo el audio del arreglo numerosGuardados en la posición 2
    sonidoAS.clip = preguntasSonidos[numerosGuardados[2]];
    sonidoAS.Play();
}
/*si el nombre del botón en la posición del contador numPP es igual Op4_act*/
 if (nextFieldPreg[numPP].name == "Op4_act")
{
    conteoAlter--;//decremento en 1 a la posición del contador.

    /*recorro cada elemento de la lista inputsVarios mediante
    la variable item y los mando a bloquear*/
    foreach (var item in inputsVarios){
    item.interactable = false;
    }
    //cargo el audio del arreglo numerosGuardados en la posición 3
    sonidoAS.clip = preguntasSonidos[numerosGuardados[3]];
    sonidoAS.Play();
}
/*si el nombre del botón en la posición del contador numPP es igual Op5_act*/
 if (nextFieldPreg[numPP].name == "Op5_act")
{
    conteoAlter--;//decremento en 1 a la posición del contador.

    /*recorro cada elemento de la lista inputsVarios mediante
    la variable item y los mando a bloquear*/
    foreach (var item in inputsVarios){
    item.interactable = false;
    }
    //cargo el audio del arreglo numerosGuardados en la posición 4
    sonidoAS.clip = preguntasSonidos[numerosGuardados[4]];
    sonidoAS.Play();//mando a reproducir el audio
}
/*si el nombre del botón en la posición del contador numPP es igual Op6_act*/
 if (nextFieldPreg[numPP].name == "Op6_act")
{
    conteoAlter--;//decremento en 1 a la posición del contador.

    /*recorro cada elemento de la lista inputsVarios mediante
    la variable item y los mando a bloquear*/
    foreach (var item in inputsVarios){
    item.interactable = false;
    }
    //cargo el audio del arreglo numerosGuardados en la posición 5
    sonidoAS.clip = preguntasSonidos[numerosGuardados[5]];
    sonidoAS.Play();//mando a reproducir el audio
}
```

Figura 304. Control por teclado función Tabulando() parte 12.

Figura 305. Control por teclado función Tabulando() parte 13.



```
//si el contador numPP es igual totPregs, llega al final del arreglo
else if (numPP == totPregs)
{
    /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
    {
        //Y si el contador conteoAlter es menor a 0
        if(conteoAlter < 0){

            conteoAlter = 0;//inicializo el contador en 0
        }
        //si el contador es mayor al tamaño del arreglo preguntasSonidos
        else if(conteoAlter > preguntasSonidos.Length - 1){
            conteoAlter = 5;//inicializo en contador en 0
        }
        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
        y de esa manera no se cruzan los audios que esten reproduciendose*/
        foreach (var item in audiosPregMouse)
        {
            item.Stop();
        }
        juegI[numFPP].enabled = false;//desactivo la imagen actual.
        numPP = 0;//inicializo el contador en 0
        numFPP = 0;//inicializo el contador en 0
        audiosConteo = 0;//inicializo el contador en 0
        conteoAlter = 0;//inicializo el contador en 0
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        sonidoAS.Stop();//detengo todos los audios
        sonidoAS.clip = audiosPreg[numPP];//cargo el audio del arreglo audiosPreg de la posición actual de numPP.
        sonidoAS.Play();//mando a reproducir
        juegI[numFPP].enabled = true;//activo la imagen actual.
    }
    /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
    else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
    {
        //si el contador es menor a 0
        if(conteoAlter < 0){
            conteoAlter = 0;//inicializo el contador en 0
        }
        //si el contador es mayor al tamaño del arreglo preguntasSonidos
        else if(conteoAlter > preguntasSonidos.Length - 1){
            conteoAlter = 5;//inicializo el contador en 5
        }
        numPP--;//decremento en 1 a la posición del contador.
        numFPP--;//decremento en 1 a la posición del contador.
        audiosConteo--;//decremento en 1 a la posición del contador.
        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
        y de esa manera no se cruzan los audios que esten reproduciendose*/
        foreach (var item in audiosPregMouse)
        {
            item.Stop();
        }
        //desactivo la imagen de la posición actual +1
        juegI[numFPP + 1].enabled = false;
        nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
        sonidoAS.Stop();//detengo todos los audios
```

Figura 306. Control por teclado función Tabulando() parte 15.

```
                nextFieldPreg[numPP].Select();//selecciono el boton en la posición actual.
                sonidoAS.Stop();//detengo todos los audios
                //si el botón siguiente está activado
                if (botSig.activeSelf == true)
                    {
                        //cargo el audio del arreglo audiosFin de la posición actual de audiosConteo
                        sonidoAS.clip = audiosPreg[audiosConteo];
                    }
                    //si el botón siguiente está desactivado
                    else if (botSig.activeSelf == false)
                    {
                        //cargo el audio diasAud
                        sonidoAS.clip = diasAud;
                    }

                sonidoAS.Play();//mando a reproducir el audio
                juegI[numFPP].enabled = true;//activo la imagen actual.
            }
        }


    }
```

Figura 307. Control por teclado función Tabulando() parte 16.

T

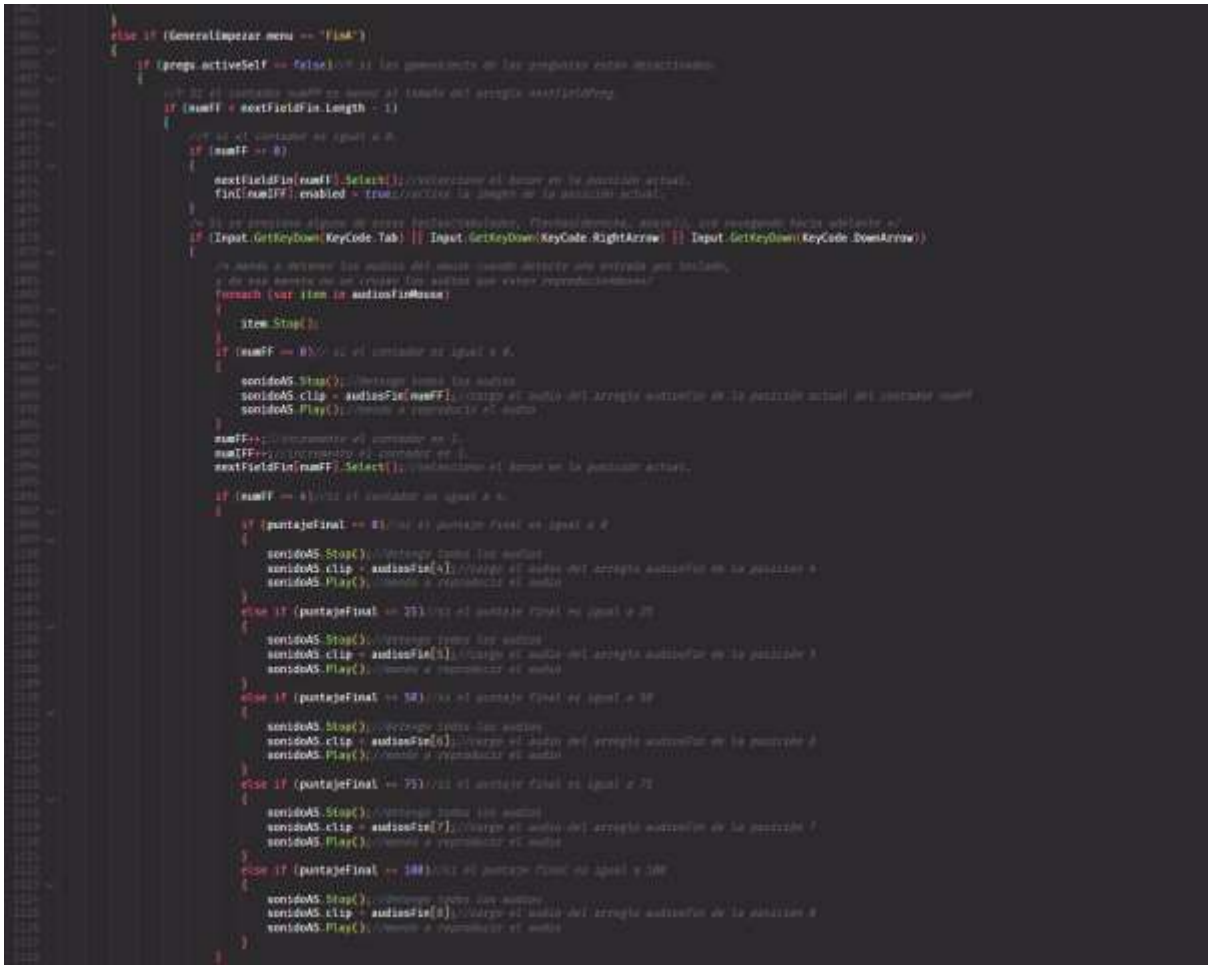


Figura 308. Control por teclado función Tabulando() parte 17.

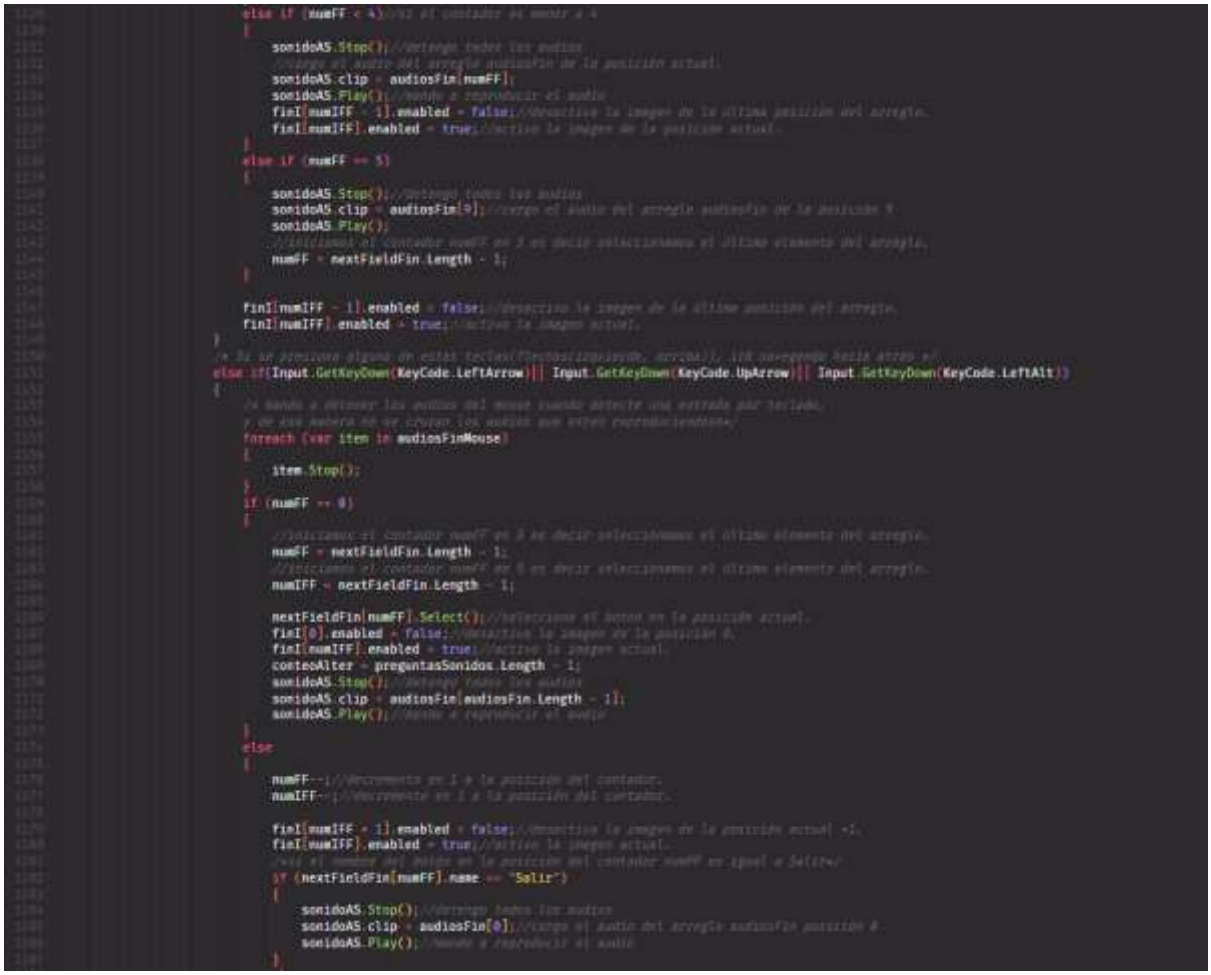


Figura 309. Control por teclado función Tabulando() parte 18.

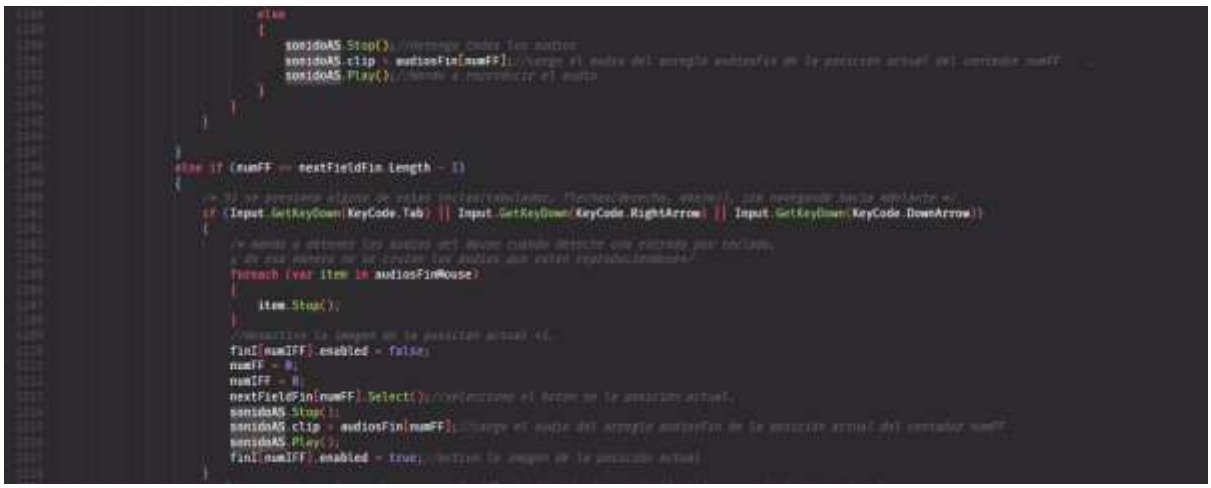
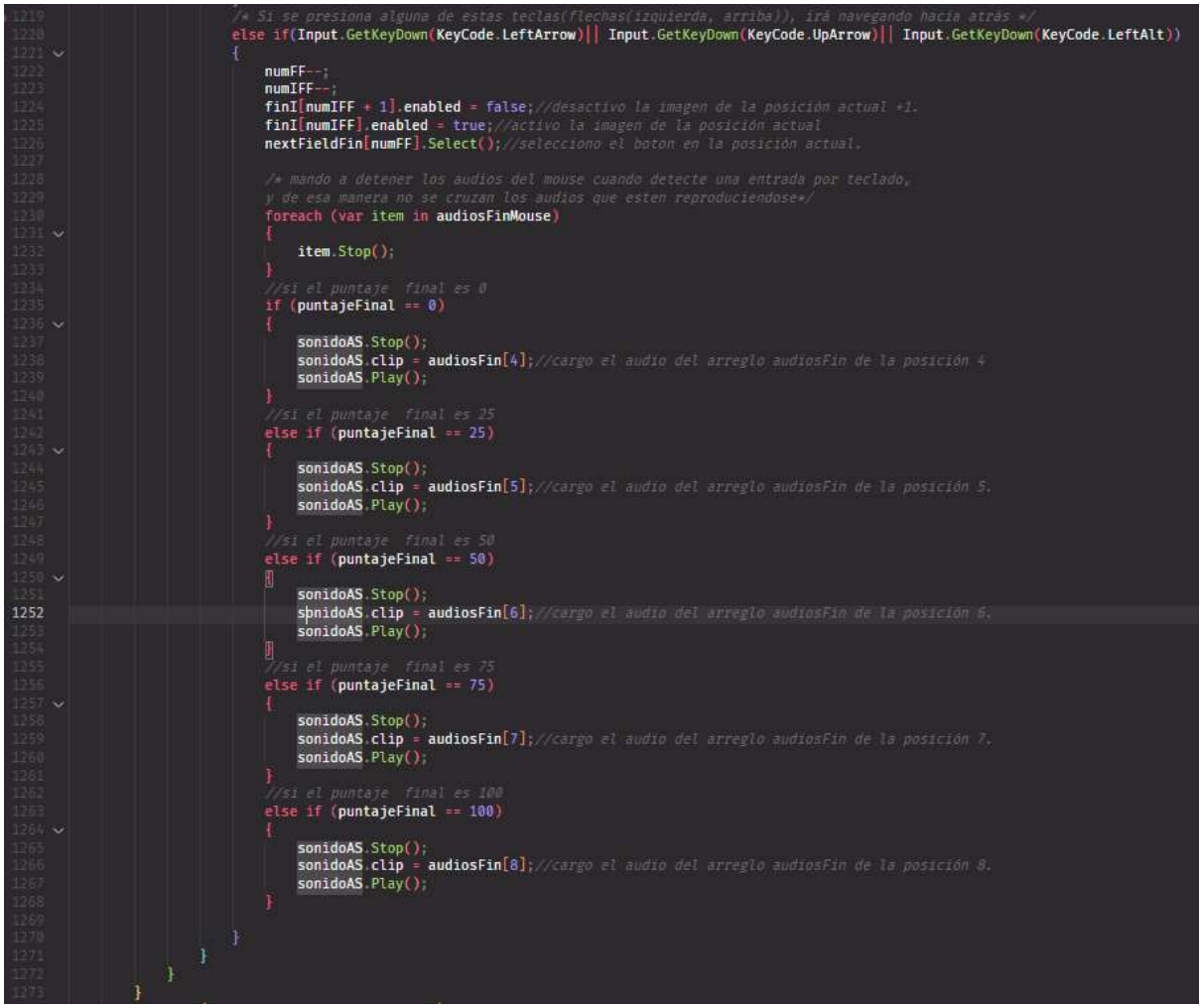


Figura 310. Control por teclado función Tabulando() parte 19.

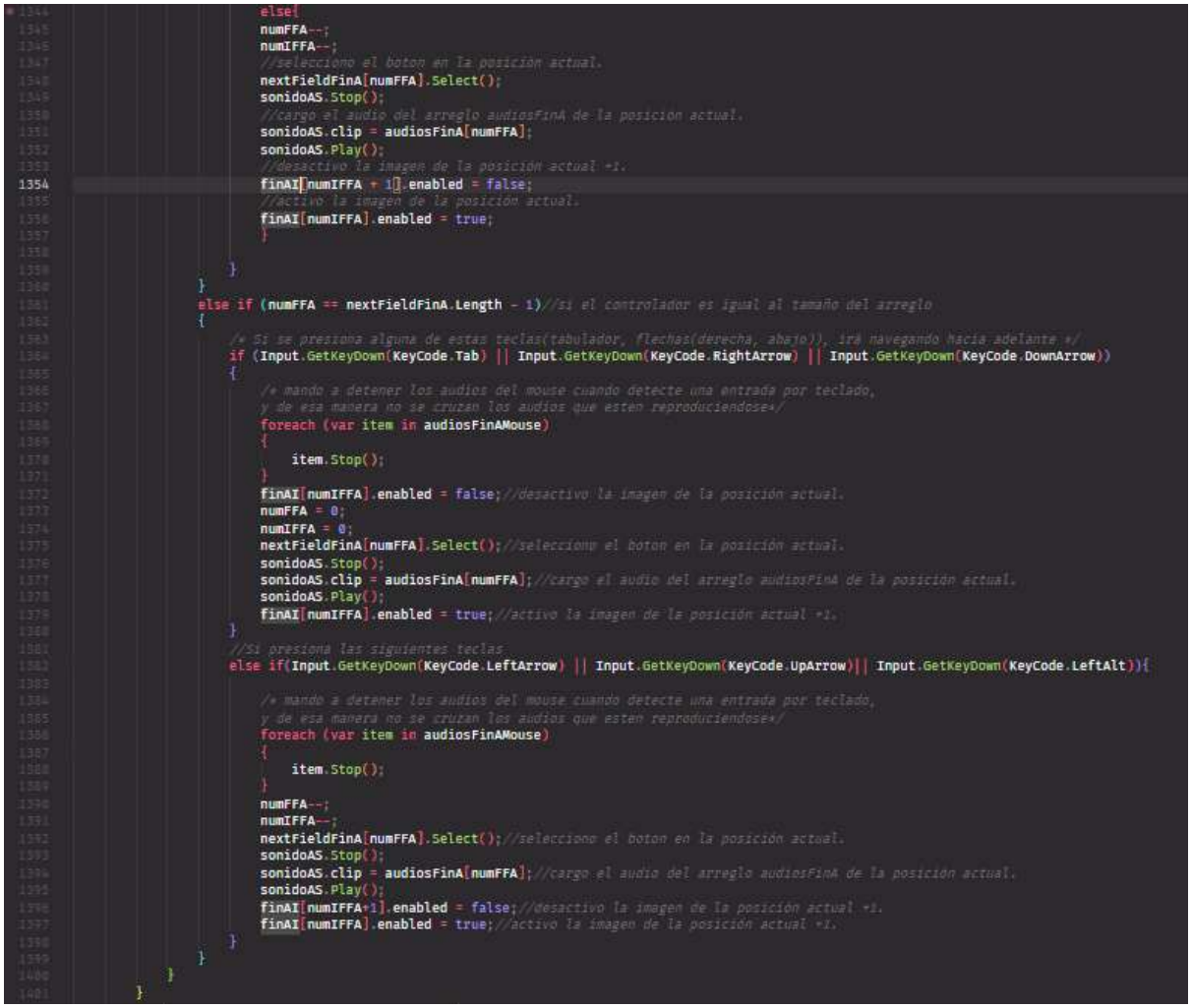


```
1219                /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
1220                else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
1221                {
1222                    numFF--;
1223                    numIFF--;
1224                    finI[numIFF + 1].enabled = false;//desactivo la imagen de la posición actual +1.
1225                    finI[numIFF].enabled = true;//activo la imagen de la posición actual
1226                    nextFieldFin[numFF].Select();//selecciono el boton en la posición actual.
1227
1228                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1229                    y de esa manera no se cruzan los audios que esten reproduciendose*/
1230                    foreach (var item in audiosFinMouse)
1231                    {
1232                        item.Stop();
1233                    }
1234                    //si el puntaje  final es 0
1235                    if (puntajeFinal == 0)
1236                    {
1237                        sonidoAS.Stop();
1238                        sonidoAS.clip = audiosFin[4];//cargo el audio del arreglo audiosFin de la posición 4
1239                        sonidoAS.Play();
1240                    }
1241                    //si el puntaje  final es 25
1242                    else if (puntajeFinal == 25)
1243                    {
1244                        sonidoAS.Stop();
1245                        sonidoAS.clip = audiosFin[5];//cargo el audio del arreglo audiosFin de la posición 5.
1246                        sonidoAS.Play();
1247                    }
1248                    //si el puntaje  final es 50
1249                    else if (puntajeFinal == 50)
1250                    {
1251                        sonidoAS.Stop();
1252                        sonidoAS.clip = audiosFin[6];//cargo el audio del arreglo audiosFin de la posición 6.
1253                        sonidoAS.Play();
1254                    }
1255                    //si el puntaje  final es 75
1256                    else if (puntajeFinal == 75)
1257                    {
1258                        sonidoAS.Stop();
1259                        sonidoAS.clip = audiosFin[7];//cargo el audio del arreglo audiosFin de la posición 7.
1260                        sonidoAS.Play();
1261                    }
1262                    //si el puntaje  final es 100
1263                    else if (puntajeFinal == 100)
1264                    {
1265                        sonidoAS.Stop();
1266                        sonidoAS.clip = audiosFin[8];//cargo el audio del arreglo audiosFin de la posición 8.
1267                        sonidoAS.Play();
1268                    }
1269
1270                }
1271            }
1272        }
1273    }
```

Figura 311. Control por teclado función Tabulando() parte 20.

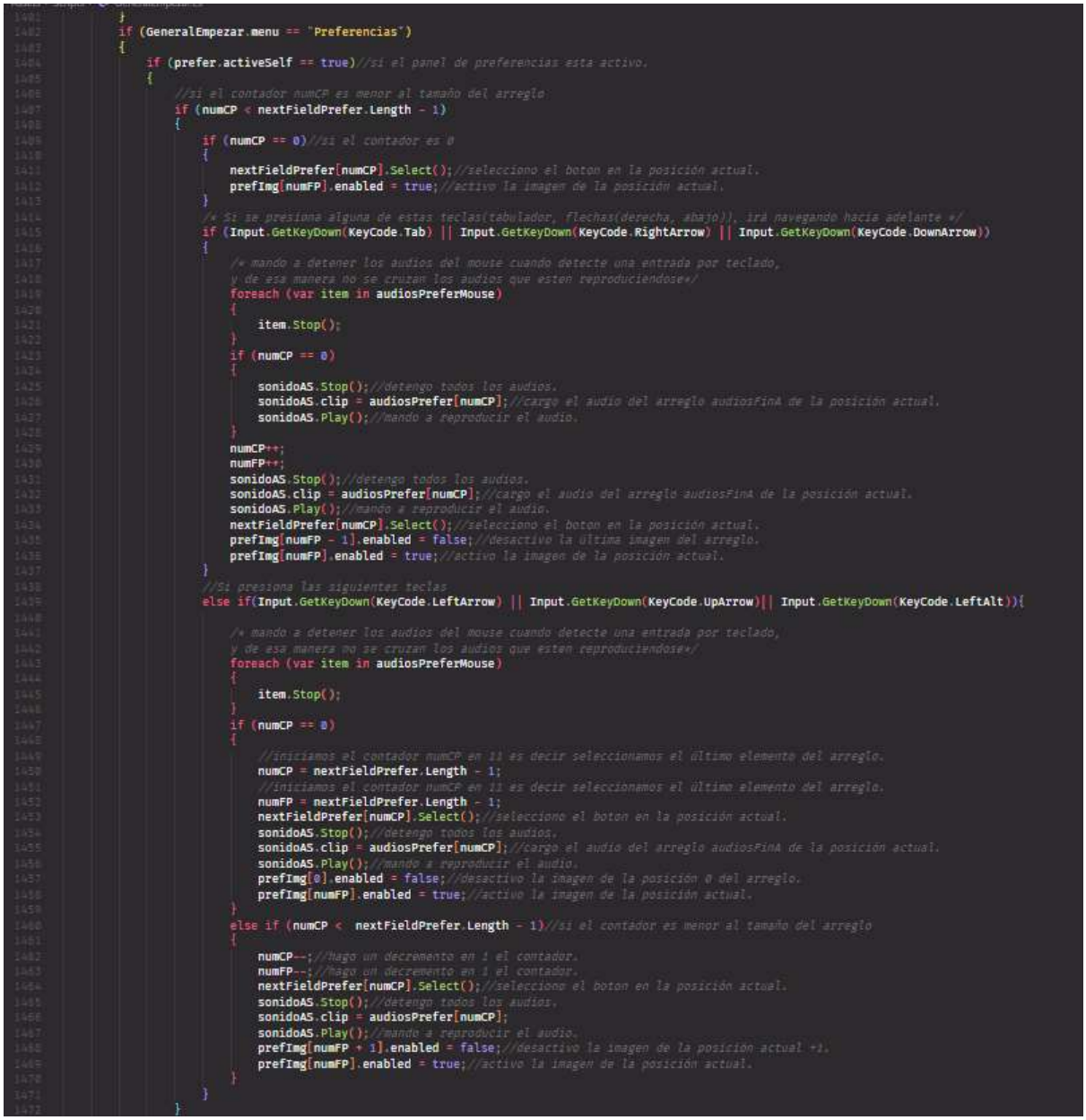
```
else if (GeneralEmpezar.menu == "Fin")
{
    //si los gameobjects de las preguntas estan desactivados y el boton sigueinte esta activado
    if (pregu.activeSelf == false && botSig.activeSelf == true)
    {
        //Y si el contador es menor al tamaño del arrelo
        if (numFFA < nextFieldFinA.Length - 1)
        {
            //si el contador es 0
            if (numFFA == 0)
            {
                //selecciono el boton en la posición actual.
                nextFieldFinA[numFFA].Select();
                //activo la imagen de la posición actual.
                finAI[numIFFA].enabled = true;
            }
            /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
            if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
            {
                /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                y de esa manera no se cruzan los audios que esten reproduciendose*/
                foreach (var item in audiosFinAMouse)
                {
                    item.Stop();
                }
                if (numFFA == 0)
                {
                    sonidoAS.Stop();
                    //cargo el audio del arreglo audiosFinA de la posición actual del contador numFFA.
                    sonidoAS.clip = audiosFinA[numFFA];
                    sonidoAS.Play();//mando a reproducir el audio
                }
                numFFA++;//incremento el contador en 1.
                numIFFA++;//incremento el contador en 1.
                //selecciono el boton en la posición actual.
                nextFieldFinA[numFFA].Select();
                sonidoAS.Stop();
                //cargo el audio del arreglo audiosFinA de la posición actual.
                sonidoAS.clip = audiosFinA[numFFA];
                sonidoAS.Play();
                //desactivo la última imagen del arreglo finAI
                finAI[numIFFA - 1].enabled = false;
                //activo la imagen de la posición actual.
                finAI[numIFFA].enabled = true;
            }
            /* Si se presiona alguna de estas teclas(flechas(izquierda, arriba)), irá navegando hacia atrás */
            else if(Input.GetKeyDown(KeyCode.LeftArrow)|| Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt))
            {
                /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                y de esa manera no se cruzan los audios que esten reproduciendose*/
                foreach (var item in audiosFinAMouse)
                {
                    item.Stop();
                }

                if (numFFA == 0)//si el contadoe es 0
                {
                    //iniciamos el contador numFFA en 5 es decir seleccionamos el último elemento del arreglo.
                    numFFA = nextFieldFinA.Length - 1;
                    //iniciamos el contador numIFFA en 5 es decir seleccionamos el último elemento del arreglo.
                    numIFFA = nextFieldFinA.Length - 1;
                    //selecciono el boton en la posición actual.
                    nextFieldFinA[numFFA].Select();
                    finAI[0].enabled = false;//desactivo la imagen de la posición 0 del arreglo.
                    finAI[numIFFA].enabled = true;//activo la imagen de la posición actual.
                    sonidoAS.Stop();
                    //cargo el audio del arreglo audiosFinA de la posición actual.
                    sonidoAS.clip = audiosFinA[numFFA];
                    sonidoAS.Play();
                }
```

Figura 312. Control por teclado función Tabulando() parte 21.



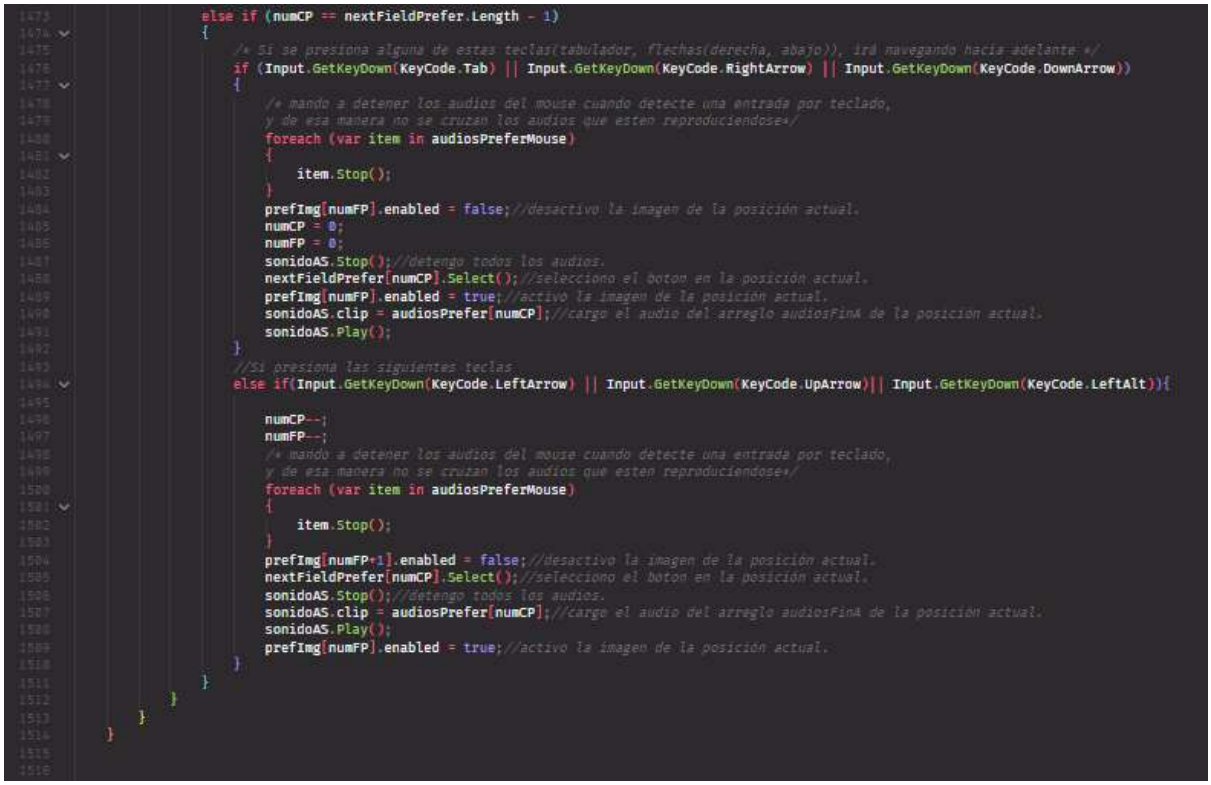
```
                    else{
                    numFFA--;
                    numIFFA--;
                    //selecciono el boton en la posición actual.
                    nextFieldFinA[numFFA].Select();
                    sonidoAS.Stop();
                    //cargo el audio del arreglo audiosFinA de la posición actual.
                    sonidoAS.clip = audiosFinA[numFFA];
                    sonidoAS.Play();
                    //desactivo la imagen de la posición actual +1.
                    finAI[numIFFA + 1].enabled = false;
                    //activo la imagen de la posición actual.
                    finAI[numIFFA].enabled = true;
                    }

                }
            }
            else if (numFFA == nextFieldFinA.Length - 1)//si el controlador es igual al tamaño del arreglo
            {
                /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
                if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
                {
                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                    y de esa manera no se cruzan los audios que esten reproduciendose*/
                    foreach (var item in audiosFinAMouse)
                    {
                        item.Stop();
                    }
                    finAI[numIFFA].enabled = false;//desactivo la imagen de la posición actual.
                    numFFA = 0;
                    numIFFA = 0;
                    nextFieldFinA[numFFA].Select();//selecciono el boton en la posición actual.
                    sonidoAS.Stop();
                    sonidoAS.clip = audiosFinA[numFFA];//cargo el audio del arreglo audiosFinA de la posición actual.
                    sonidoAS.Play();
                    finAI[numIFFA].enabled = true;//activo la imagen de la posición actual +1.
                }
                //Si presiona las siguientes teclas
                else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){

                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                    y de esa manera no se cruzan los audios que esten reproduciendose*/
                    foreach (var item in audiosFinAMouse)
                    {
                        item.Stop();
                    }
                    numFFA--;
                    numIFFA--;
                    nextFieldFinA[numFFA].Select();//selecciono el boton en la posición actual.
                    sonidoAS.Stop();
                    sonidoAS.clip = audiosFinA[numFFA];//cargo el audio del arreglo audiosFinA de la posición actual.
                    sonidoAS.Play();
                    finAI[numIFFA+1].enabled = false;//desactivo la imagen de la posición actual +1.
                    finAI[numIFFA].enabled = true;//activo la imagen de la posición actual +1.
                }
            }
        }
    }
```

Figura 313. Control por teclado función Tabulando() parte 22.



```
        }
        if (GeneralEmpezar.menu == "Preferencias")
        {
            if (prefer.activeSelf == true)//si el panel de preferencias esta activo.
            {
                //si el contador numCP es menor al tamaño del arreglo
                if (numCP < nextFieldPrefer.Length - 1)
                {
                    if (numCP == 0)//si el contador es 0
                    {
                        nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                        prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
                    }
                    /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
                    if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
                    {
                        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                        y de esa manera no se cruzan los audios que esten reproduciendose*/
                        foreach (var item in audiosPreferMouse)
                        {
                            item.Stop();
                        }
                        if (numCP == 0)
                        {
                            sonidoAS.Stop();//detengo todos los audios.
                            sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinA de la posición actual.
                            sonidoAS.Play();//mando a reproducir el audio.
                        }
                        numCP++;
                        numFP++;
                        sonidoAS.Stop();//detengo todos los audios.
                        sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinA de la posición actual.
                        sonidoAS.Play();//mando a reproducir el audio.
                        nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                        prefImg[numFP - 1].enabled = false;//desactivo la última imagen del arreglo.
                        prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
                    }
                    //Si presiona las siguientes teclas
                    else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){

                        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                        y de esa manera no se cruzan los audios que esten reproduciendose*/
                        foreach (var item in audiosPreferMouse)
                        {
                            item.Stop();
                        }
                        if (numCP == 0)
                        {
                            //iniciamos el contador numCP en 11 es decir seleccionamos el último elemento del arreglo.
                            numCP = nextFieldPrefer.Length - 1;
                            //iniciamos el contador numCP en 11 es decir seleccionamos el último elemento del arreglo.
                            numFP = nextFieldPrefer.Length - 1;
                            nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                            sonidoAS.Stop();//detengo todos los audios.
                            sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinA de la posición actual.
                            sonidoAS.Play();//mando a reproducir el audio.
                            prefImg[0].enabled = false;//desactivo la imagen de la posición 0 del arreglo.
                            prefImg[numFP].enabled = true; //activo la imagen de la posición actual.
                        }
                        else if (numCP <  nextFieldPrefer.Length - 1)//si el contador es menor al tamaño del arreglo
                        {
                            numCP--;//hago un decremento en 1 el contador.
                            numFP--;//hago un decremento en 1 el contador.
                            nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                            sonidoAS.Stop();//detengo todos los audios.
                            sonidoAS.clip = audiosPrefer[numCP];
                            sonidoAS.Play();//mando a reproducir el audio.
                            prefImg[numFP + 1].enabled = false;//desactivo la imagen de la posición actual +1.
                            prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
                        }
                    }
                }
            }
```

Figura 314. Control por teclado función Tabulando() parte 23.



```
            else if (numCP == nextFieldPrefer.Length - 1)
            {
                /* Si se presiona alguna de estas teclas(tabulador, flechas(derecha, abajo)), irá navegando hacia adelante */
                if (Input.GetKeyDown(KeyCode.Tab) || Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.DownArrow))
                {
                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                    y de esa manera no se cruzan los audios que esten reproduciendose*/
                    foreach (var item in audiosPreferMouse)
                    {
                        item.Stop();
                    }
                    prefImg[numFP].enabled = false;//desactivo la imagen de la posición actual.
                    numCP = 0;
                    numFP = 0;
                    sonidoAS.Stop();//detengo todos los audios.
                    nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                    prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
                    sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinA de la posicion actual.
                    sonidoAS.Play();
                }
                //Si presiona las siguientes teclas
                else if(Input.GetKeyDown(KeyCode.LeftArrow) || Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown(KeyCode.LeftAlt)){

                    numCP--;
                    numFP--;
                    /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
                    y de esa manera no se cruzan los audios que esten reproduciendose*/
                    foreach (var item in audiosPreferMouse)
                    {
                        item.Stop();
                    }
                    prefImg[numFP+1].enabled = false;//desactivo la imagen de la posición actual.
                    nextFieldPrefer[numCP].Select();//selecciono el boton en la posición actual.
                    sonidoAS.Stop();//detengo todos los audios.
                    sonidoAS.clip = audiosPrefer[numCP];//cargo el audio del arreglo audiosFinA de la posicion actual.
                    sonidoAS.Play();
                    prefImg[numFP].enabled = true;//activo la imagen de la posición actual.
                }
            }
        }
    }
}
```

Figura 315. Control por teclado función Tabulando() parte 23.

```
    /*Esta funcion me permite poner el cronómetro para poder inicializar
    cada vez que respondo las preguntas*/
    public void correTiempo()
    {
        float TimerControl = Time.time - StartTime;
        string mins = ((int)TimerControl / 60).ToString("00");
        string segs = (TimerControl % 60).ToString("00");
        string milisegs = ((TimerControl * 100) % 100).ToString("00");
        string TimerString = string.Format("{00}:{01}:{02}", mins, segs, milisegs);
        tiempoText.text = TimerString;//en esta variable se guarda el tiempo de respuesta empleado por pregunta.
    }
```

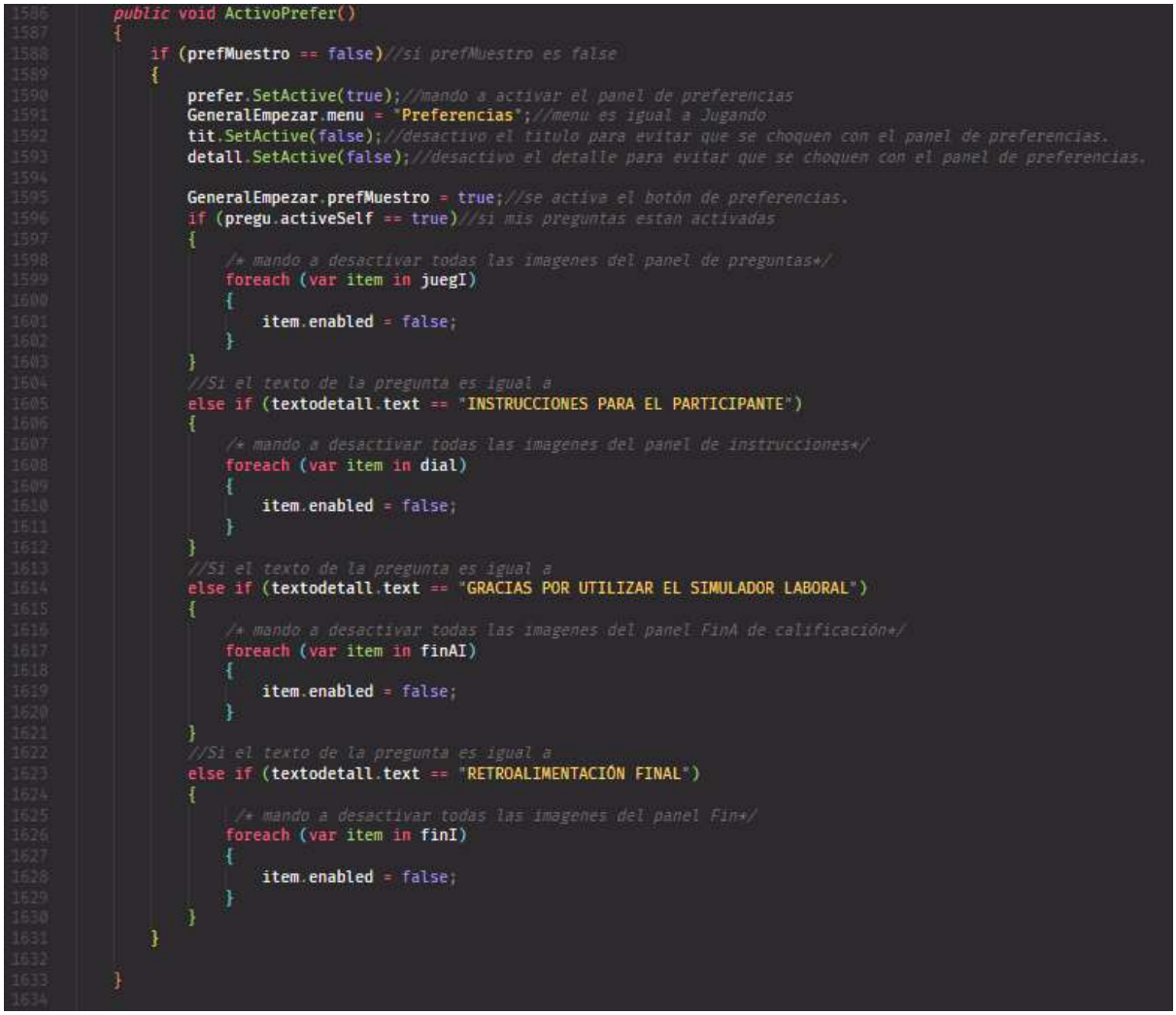
Figura 316. Función correTiempo().

```
1529    //Esta función es llamada cuando se presiona el botón empezar
1530    public void Empiezo()
1531    {
1532        nuevoJuego();//inicializa todo como un juego nuevo.
1533        textoTit.text = "EJERCITARIO EL TIEMPO";//se muestra en el panel main
1534        textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE";//se muestra en el panel main
1535        textoconten.text = instrucciones[conteoInstruc];//
1536        datetime = DateTime.Now.ToString("hh:mm:ss");//registra la hora de inicio
1537        //la imagen de la casa la cual corresponde al botón de reiniciar cambia por la imagen del boton siguiente.
1538        sigoFin.image.sprite = ImagenSiguiente;
1539        GenerarAleatoriosSinRepetir();//llamo al metodo de generar números aleatorios.
1540        GeneralEmpezar.muevoCamara = false;//deshabilito el movimiento de la cámara.
1541        empiezo.enabled = false;//desactivo el botón empezar.
1542        textoEmpiezo.gameObject.SetActive(false);//desactivo el texto del botón empezar
1543        /*Activo todos los objetos del panel de instrucciones*/
1544        juego.SetActive(true);
1545        botSig.SetActive(true);
1546        botAtras.SetActive(true);
1547        botPref.SetActive(true);
1548        botSalir.SetActive(true);
1549        tit.SetActive(true);
1550        detall.SetActive(true);
1551        conten.SetActive(true);
1552        GeneralEmpezar.menu = "Jugando";//menu es igual a Jugando
1553    }
```

Figura 317. Función Empiezo().

```
1555    /*Funcion para generar números aleatorios*/
1556    void GenerarAleatoriosSinRepetir()
1557    {
1558
1559        numerosGuardados = new List<int>();//crea una lista de numeros aleatorios guardados
1560        numPares = new List<int>();//se crea una lista de numeros pares correspondiente a las preguntas
1561        listaFinalPregs = new List<int>();//se crea lista de preguntas randomicas.
1562        int posicionAleatoria;
1563        //si el contador es menor al tamaño de la lista contenPregFinal es decir al número de preguntas
1564        for (int i = 0; i < (contenPregFinal.Count); i++)
1565        {
1566            do
1567            {
1568                /* mando a generar números aleatorios de 0 a 5, ya que contenPregFinal tiene 6 preguntas*/
1569                posicionAleatoria = Random.Range(0, (contenPregFinal.Count));
1570            //mientras numerosGuardados contenga un número de posicionAleatoria
1571            } while (numerosGuardados.Contains(posicionAleatoria));
1572            numerosGuardados.Add(posicionAleatoria);//se agrega la posiciónAleatoria al arreglo numerosGuardados
1573        }
1574        //recorro con un buble y digo si i es menor al número de numerosGuardados y le incremento en 1
1575        for (int i = 0; i < numerosGuardados.Count; i++)
1576        {
1577            //añade los numeros guardados al arreglo contenPreg en el orden que recupere el contador i
1578            contenPreg.Add(contenPregFinal[numerosGuardados[i]]);
1579            //añade los numeros guardados al arreglo alternaJeraen el orden que recupere el contador i
1580            alternaJera.Add(alternaJeraFinal[numerosGuardados[i]]);
1581            //añade los numeros guardados al arreglo alternaDias en el orden que recupere el contador i
1582            alternaDias.Add(alternaDiasFinal[numerosGuardados[i]]);
1583        }
1584    }
1585
```

Figura 318. Función GenerarAleatoriosSinRepetir().



```
public void ActivoPrefer()
{
    if (prefMuestro == false)//si prefMuestro es false
    {
        prefer.SetActive(true);//mando a activar el panel de preferencias
        GeneralEmpezar.menu = "Preferencias";//menu es igual a Jugando
        tit.SetActive(false);//desactivo el titulo para evitar que se choquen con el panel de preferencias.
        detall.SetActive(false);//desactivo el detalle para evitar que se choquen con el panel de preferencias.

        GeneralEmpezar.prefMuestro = true;//se activa el botón de preferencias.
        if (pregu.activeSelf == true)//si mis preguntas estan activadas
        {
            /* mando a desactivar todas las imagenes del panel de preguntas*/
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
        {
            /* mando a desactivar todas las imagenes del panel de instrucciones*/
            foreach (var item in dial)
            {
                item.enabled = false;
            }
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL")
        {
            /* mando a desactivar todas las imagenes del panel FinA de calificación*/
            foreach (var item in finAI)
            {
                item.enabled = false;
            }
        }
        //Si el texto de la pregunta es igual a
        else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
        {
            /* mando a desactivar todas las imagenes del panel Fin*/
            foreach (var item in finI)
            {
                item.enabled = false;
            }
        }
    }

}
```

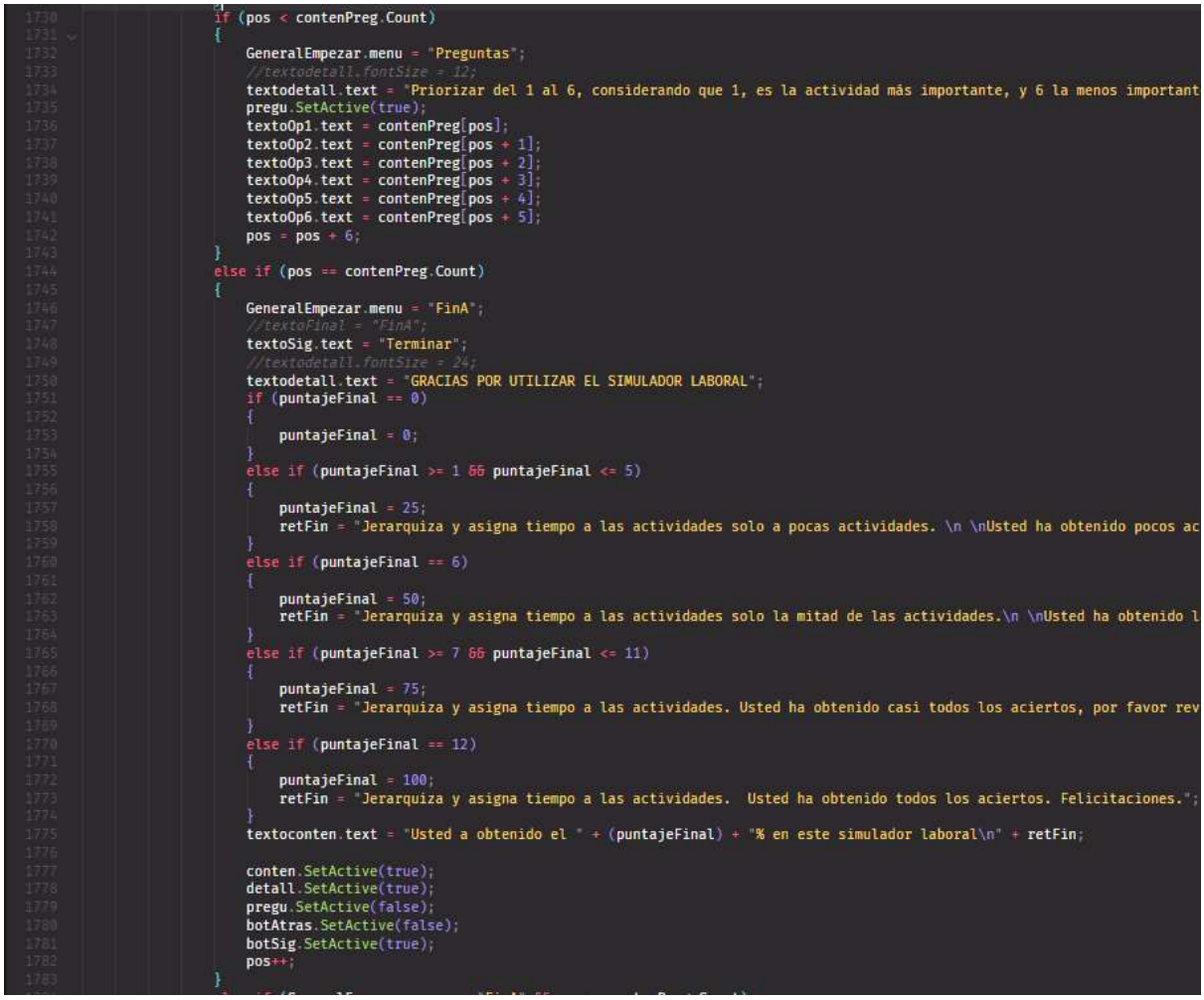
Figura 319. Función ActivoPrefer().

```
1635    public void DesctivoPrefer()
1636    {
1637        if (prefMuestro == true)//si el panel de preferencias esta activo
1638        {
1639            prefer.SetActive(false);//desactivo el panel de preferencias
1640            tit.SetActive(true);//activo el titulo
1641            detall.SetActive(true);//activo el detalle
1642            GeneralEmpezar.prefMuestro = false;//desactivo botón de
1643            if (pregu.activeSelf == true)//si las preguntas están activadas
1644            {   //el menu es igual a Preguntas
1645                GeneralEmpezar.menu = "Preguntas";
1646            }
1647            //Si el texto de la pregunta es igual a
1648            else if (textodetall.text == "INSTRUCCIONES PARA EL PARTICIPANTE")
1649            {
1650                //el menu es igual a Jugando
1651                GeneralEmpezar.menu = "Jugando";
1652            }
1653            //Si el texto de la pregunta es igual a
1654            else if (textodetall.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL")
1655            {
1656                //el menu es igual a FinA
1657                GeneralEmpezar.menu = "FinA";
1658            }
1659            //Si el texto de la pregunta es igual a
1660            else if (textodetall.text == "RETROALIMENTACIÓN FINAL")
1661            {
1662                //el menu es igual a Fin
1663                GeneralEmpezar.menu = "Fin";
1664            }
1665        }
1666    }
```

Figura 320. Función DesctivoPrefer().

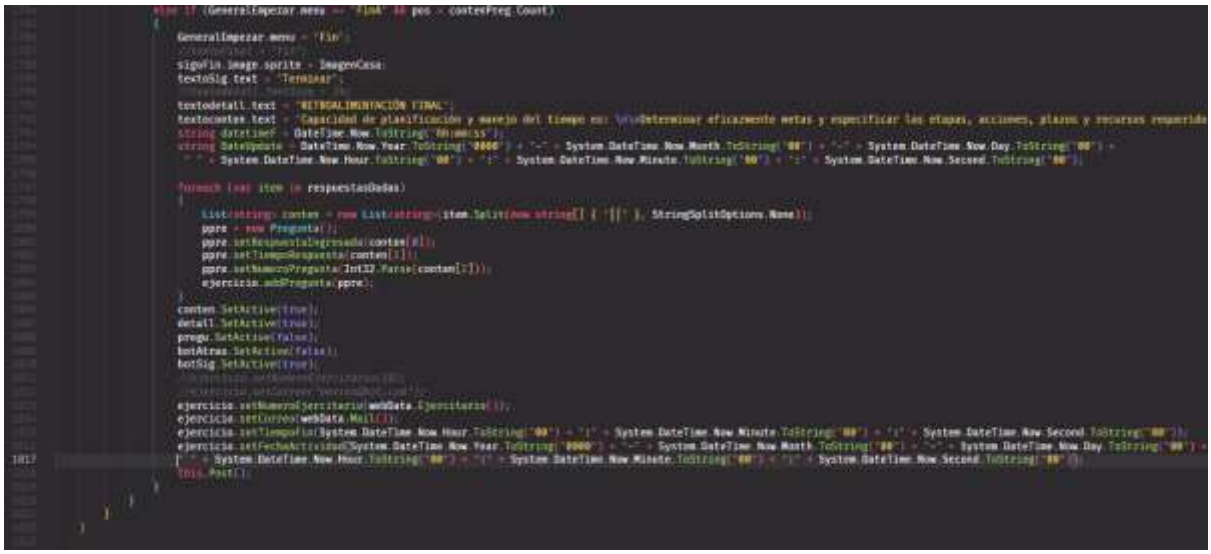
```
public void Siguiente()
{
    ReinicioSeleccion();//llamo a la función para que cada vez que presione el boton siguiente reinicie los valores a 0.

    /* Solo si el panel de preferencias está desactivado podra pasar al siguiente panel al presionar la tecla siguiente .
    caso contrario deberá cerrar el panel de preferencias primero*/
    if (prefer.activeSelf == false)//si el panel de preferencias esta activo
    {
        conteoInstruc++;//incremento el contador en 1

        //si el contador es menor al número de instrucciones
        if (conteoInstruc < instrucciones.Count)
        {
            //se pasa a textoconten la instrucciones de la posición actual del contador conteoInstruc
            textoconten.text = instrucciones[conteoInstruc];
        }
        //si menu es igual a Fin
        else if (GeneralEmpezar.menu == "Fin")
        {

            GeneralEmpezar.muevoCamara = true;//activo el movimiento de la cámara con el mouse.
            empiezo.enabled = true;//activo el botón empezar.
            textoEmpiezo.gameObject.SetActive(true);//activo el texto del boton empiezo
            tiempoText.text = "00:00:00";//inicilizo la variable de cronometro a 0
            juego.SetActive(false);
            botSig.SetActive(false);
            botAtras.SetActive(false);
            botPref.SetActive(false);
            botSalir.SetActive(false);
            tit.SetActive(false);
            detall.SetActive(false);
            conten.SetActive(false);
            prefer.SetActive(false);
            foreach (var item in finI)
            {
                item.enabled = false;
            }
            GeneralEmpezar.menu = "Main";
            pos = 0;
            contRes = 0;
            numC = 0;
            numI = 0;
            numCP = 0;
            numFP = 0;
            numPP = 0;
            numFPP = 0;
            numFF = 0;
            numIFF = 0;
            puntajeFinal = 0;
            conteoAlter = 0;
            audiosConteo = 0;
        }
        else
        {
            conten.SetActive(false);
            nextFieldDiag[0].Select();
            botSig.SetActive(false);
            botAtras.SetActive(true);
            foreach (var item in juegI)
            {
                item.enabled = false;
            }
```

Figura 321. Función Siguiente() parte 1.



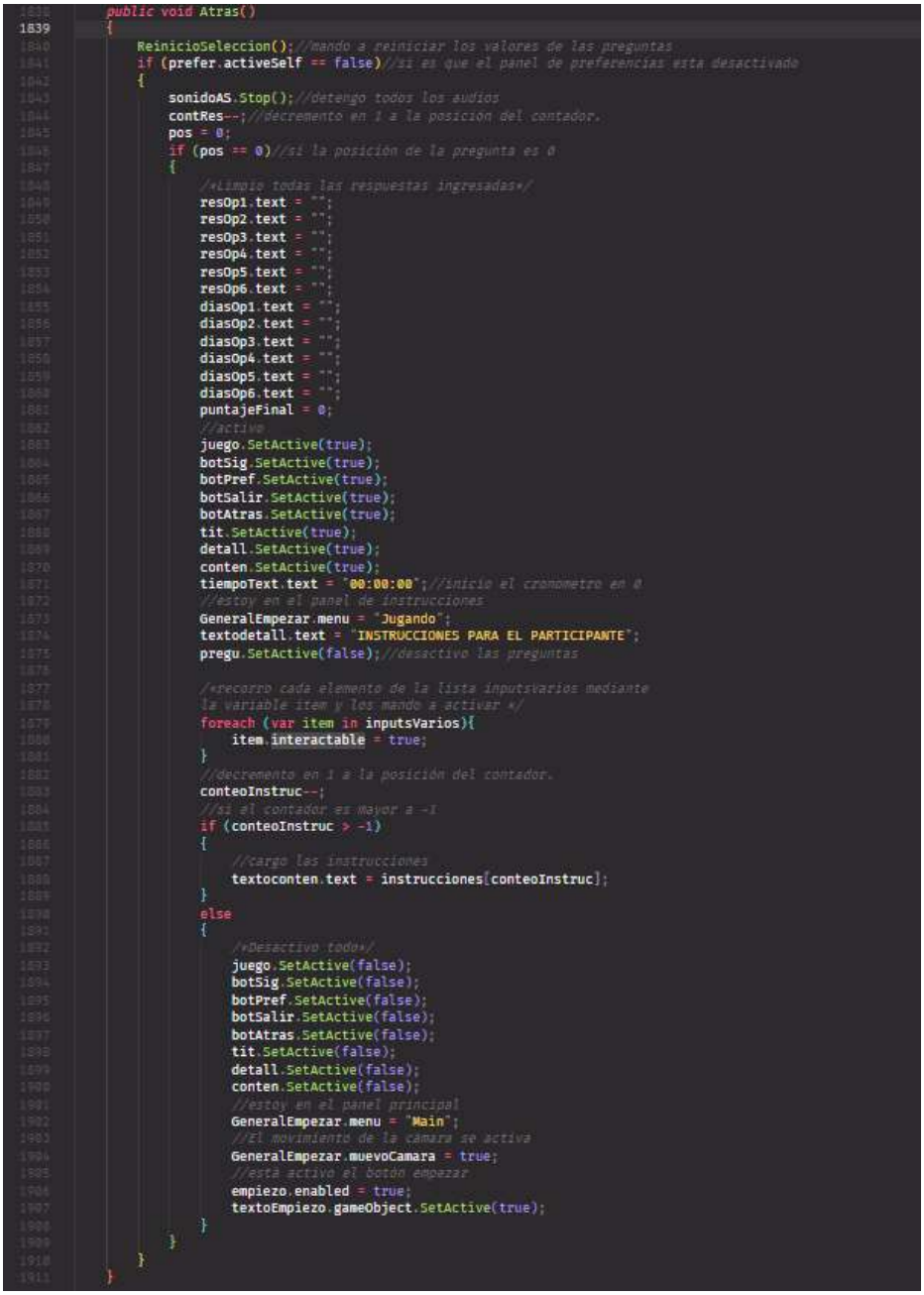
```
if (pos < contenPreg.Count)
{
    GeneralEmpezar.menu = "Preguntas";
    //textodetall.fontSize = 12;
    textodetall.text = "Priorizar del 1 al 6, considerando que 1, es la actividad más importante, y 6 la menos important
    pregu.SetActive(true);
    textoOp1.text = contenPreg[pos];
    textoOp2.text = contenPreg[pos + 1];
    textoOp3.text = contenPreg[pos + 2];
    textoOp4.text = contenPreg[pos + 3];
    textoOp5.text = contenPreg[pos + 4];
    textoOp6.text = contenPreg[pos + 5];
    pos = pos + 6;
}
else if (pos == contenPreg.Count)
{
    GeneralEmpezar.menu = "FinA";
    //textoFinal = "FinA";
    textoSig.text = "Terminar";
    //textodetall.fontSize = 24;
    textodetall.text = "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL";
    if (puntajeFinal == 0)
    {
        puntajeFinal = 0;
    }
    else if (puntajeFinal >= 1 && puntajeFinal <= 5)
    {
        puntajeFinal = 25;
        retFin = "Jerarquiza y asigna tiempo a las actividades solo a pocas actividades. \n \nUsted ha obtenido pocos ac
    }
    else if (puntajeFinal == 6)
    {
        puntajeFinal = 50;
        retFin = "Jerarquiza y asigna tiempo a las actividades solo la mitad de las actividades.\n \nUsted ha obtenido l
    }
    else if (puntajeFinal >= 7 && puntajeFinal <= 11)
    {
        puntajeFinal = 75;
        retFin = "Jerarquiza y asigna tiempo a las actividades. Usted ha obtenido casi todos los aciertos, por favor rev
    }
    else if (puntajeFinal == 12)
    {
        puntajeFinal = 100;
        retFin = "Jerarquiza y asigna tiempo a las actividades.  Usted ha obtenido todos los aciertos. Felicitaciones.";
    }
    textoconten.text = "Usted a obtenido el " + (puntajeFinal) + "% en este simulador laboral\n" + retFin;

    conten.SetActive(true);
    detall.SetActive(true);
    pregu.SetActive(false);
    botAtras.SetActive(false);
    botSig.SetActive(true);
    pos++;
}
```

Figura 322. Función Siguiente() parte 2.



```
else if (GeneralEmpezar.menu == "FinA" && pos < contenPreg.Count)
{
    GeneralEmpezar.menu = "Fin";
    //textodetall.fontSize = 12;
    sigoFin.image.sprite = ImagenCasa;
    textoSig.text = "Terminar";
    //textodetall.fontSize = 24;
    textodetall.text = "RETROALIMENTACIÓN FINAL";
    textoconten.text = "Capacidad de planificación y manejo del tiempo en: \r\nDeterminar eficazmente metas y especificar las etapas, acciones, plazos y recursos requeridos
    string datetime= DateTime.Now.ToString("hh:mm:ss");
    string DateUpdate = DateTime.Now.Year.ToString("0000") + "-" + System.DateTime.Now.Month.ToString("00") + "-" + System.DateTime.Now.Day.ToString("00") +
    " " + System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00");

    foreach (var item in respuestasDadas)
    {
        List<string> conten = new List<string>(item.Split(new string[] { "||" }, StringSplitOptions.None));
        ppre = new Pregunta();
        ppre.setRespuestaAgregada(conten[0]);
        ppre.setTiempoRespuesta(conten[1]);
        ppre.setNumeroPregunta(Int32.Parse(conten[2]));
        ejercicio.addPregunta(ppre);
    }
    conten.SetActive(true);
    detall.SetActive(true);
    pregu.SetActive(false);
    botAtras.SetActive(false);
    botSig.SetActive(true);
    //ejercicio.setNumeroEjercitario(0);
    //ejercicio.setCorreo("correo@hot.com");
    ejercicio.setNumeroEjercitario(webData.Ejercitarios());
    ejercicio.setCorreo(webData.Mail());
    ejercicio.setTiempoFin(System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00"));
    ejercicio.setFechaActividad(System.DateTime.Now.Year.ToString("0000") + "-" + System.DateTime.Now.Month.ToString("00") + "-" + System.DateTime.Now.Day.ToString("00") +
    " " + System.DateTime.Now.Hour.ToString("00") + ":" + System.DateTime.Now.Minute.ToString("00") + ":" + System.DateTime.Now.Second.ToString("00"));
    this.Post();
}
```

Figura 323. Función Siguiente() parte 3.



```
public void Atras()
{
    ReinicioSeleccion(); //mando a reiniciar los valores de las preguntas
    if (prefer.activeSelf == false)//si es que el panel de preferencias esta desactivado
    {
        sonidoAS.Stop();//detengo todos los audios
        contRes--;//decremento en 1 a la posición del contador.
        pos = 0;
        if (pos == 0)//si la posición de la pregunta es 0
        {
            /*Limpio todas las respuestas ingresadas*/
            resOp1.text = "";
            resOp2.text = "";
            resOp3.text = "";
            resOp4.text = "";
            resOp5.text = "";
            resOp6.text = "";
            diasOp1.text = "";
            diasOp2.text = "";
            diasOp3.text = "";
            diasOp4.text = "";
            diasOp5.text = "";
            diasOp6.text = "";
            puntajeFinal = 0;
            //activo
            juego.SetActive(true);
            botSig.SetActive(true);
            botPref.SetActive(true);
            botSalir.SetActive(true);
            botAtras.SetActive(true);
            tit.SetActive(true);
            detall.SetActive(true);
            conten.SetActive(true);
            tiempoText.text = "00:00:00";//inicio el cronometro en 0
            //estoy en el panel de instrucciones
            GeneralEmpezar.menu = "Jugando";
            textodetall.text = "INSTRUCCIONES PARA EL PARTICIPANTE";
            pregu.SetActive(false);//desactivo las preguntas

            /*recorro cada elemento de la lista inputsvarios mediante
            la variable item y los mando a activar */
            foreach (var item in inputsVarios){
                item.interactable = true;
            }
            //decremento en 1 a la posición del contador.
            conteoInstruc--;
            //si el contador es mayor a -1
            if (conteoInstruc > -1)
            {
                //cargo las instrucciones
                textoconten.text = instrucciones[conteoInstruc];
            }
            else
            {
                /*Desactivo todo*/
                juego.SetActive(false);
                botSig.SetActive(false);
                botPref.SetActive(false);
                botSalir.SetActive(false);
                botAtras.SetActive(false);
                tit.SetActive(false);
                detall.SetActive(false);
                conten.SetActive(false);
                //estoy en el panel principal
                GeneralEmpezar.menu = "Main";
                //El movimiento de la cámara se activa
                GeneralEmpezar.muevoCamara = true;
                //está activo el botón empezar
                empiezo.enabled = true;
                textoEmpiezo.gameObject.SetActive(true);
            }
        }
    }
}
```

Figura 324. Función Atras().

Esta función permite validar que los valores ingresados por teclado sean números y si ingresa otros valores se manda a reproducir un audio en el que le indica al participante que solo debe ingresar números.

```
1912        /*función para validar que se ingresen solo números como respuestas*/
1913        public void soloNumeros(TMP_InputField textBox1)
1914        {
1915            try
1916            {   //convierto a texto el valor ingresado
1917                int.Parse(textBox1.text);
1918            }
1919            catch (Exception e)
1920            {
1921                //si ingresa un valor no numerico es limpiará el objeto
1922                textBox1.text = "";
1923            }
1924        }
```

Figura 325. Función soloNumeros().

Esta función permite controlar que el no se ingresen números repetidos por teclado, esto se hace ya que son valores para jerarquizar prioridad de actividades y cuando se trate de ingresar un valor ya repetido se reproducirá un audio indicando que el número ingresado es repetido.

```
1926    /*Función para validar que no se ingresen números repetidos*/
1927    public void noRepetidosIngreso(TMP_InputField textBox1)
1928    {
1929        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1930         y de esa manera no se cruzan los audios que esten reproduciendose*/
1931        foreach (var item in audiosPregMouse)
1932        {
1933            item.Stop();
1934        }
1935        /*mando a recorrer cada valor del arreglo y digo
1936        que si mi item es diferente del valor que se esta ingresando y si el valor de item
1937        es igual al valor ingresado y es no esta vacio mando a reprouducir
1938        el audio de que el valor ingresado es repetido */
1939        foreach (var item in jerarquiaInput)
1940        {
1941            if (item.name != textBox1.name)
1942            {
1943                if (item.text == textBox1.text && textBox1.text != "")
1944                {
1945                    textBox1.text = "";//limpio el valor ingresado
1946                    sonidoAS.clip = avisoRepe;//cargo el audio de aviso de número repetido
1947                    sonidoAS.Play();//mando a reproducir el audio
1948                }
1949            }
1950        }
1951    }
1952
```

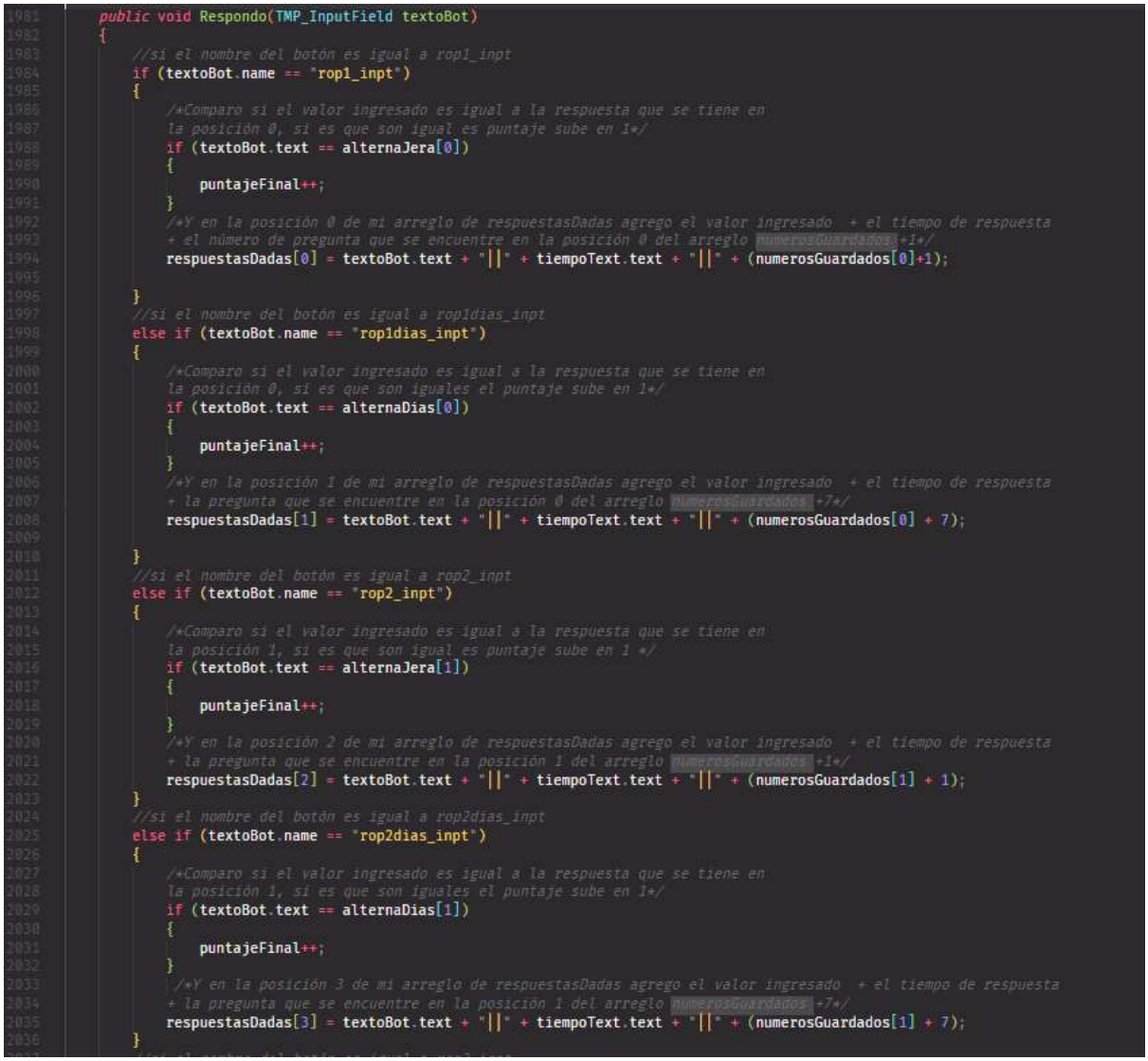
Figura326"Función noRepetidosIngreso()"

Esta función controla que para las respuestas de jerarquizar se ingresen solo valores de 1 a 6, caso contrario, se limpia el valor ingresado y a la vez se reproduce un audio en el que le indica que solo debe ingresar valores de 1 a 6.

```
1953    /*Función para validar que solo se ingresen valores no mayores a 6*/
1954    public void solohastaSeis(TMP_InputField textBox1)
1955    {
1956        /* mando a detener los audios del mouse cuando detecte una entrada por teclado,
1957         y de esa manera no se cruzan los audios que esten reproduciendose*/
1958        foreach (var item in audiosPregMouse)
1959        {
1960            item.Stop();
1961        }
1962        try
1963        {   //convierto el string a entero
1964            int.Parse(textBox1.text);
1965            //si el valor ingresado es mayor a 6
1966            if (int.Parse(textBox1.text) > 6)
1967            {
1968                //limpio el campo de texto
1969                textBox1.text = "";
1970                //cargo el audio de aviso de que el número es mayor a 6
1971                sonidoAS.clip = avisoMayor;
1972                sonidoAS.Play();//mando a reproducir el audio
1973            }
1974        }
1975        catch (Exception e)
1976        {
1977            //limpio el campo de texto
1978            textBox1.text = "";
1979        }
1980    }
```

Figura 327. Función solohastaSeis().

Figura 328. Las tres últimas funciones son llamadas dentro de cada objeto de respuesta.



```
public void Respondo(TMP_InputField textoBot)
{
    //si el nombre del botón es igual a rop1_inpt
    if (textoBot.name == "rop1_inpt")
    {
        /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
        la posición 0, si es que son igual es puntaje sube en 1*/
        if (textoBot.text == alternaJera[0])
        {
            puntajeFinal++;
        }
        /*Y en la posición 0 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
        + el número de pregunta que se encuentre en la posición 0 del arreglo numerosGuardados +1*/
        respuestasDadas[0] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0]+1);

    }
    //si el nombre del botón es igual a rop1dias_inpt
    else if (textoBot.name == "rop1dias_inpt")
    {
        /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
        la posición 0, si es que son iguales el puntaje sube en 1*/
        if (textoBot.text == alternaDias[0])
        {
            puntajeFinal++;
        }
        /*Y en la posición 1 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
        + la pregunta que se encuentre en la posición 0 del arreglo numerosGuardados +7*/
        respuestasDadas[1] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[0] + 7);

    }
    //si el nombre del botón es igual a rop2_inpt
    else if (textoBot.name == "rop2_inpt")
    {
        /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
        la posición 1, si es que son igual es puntaje sube en 1 */
        if (textoBot.text == alternaJera[1])
        {
            puntajeFinal++;
        }
        /*Y en la posición 2 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
        + la pregunta que se encuentre en la posición 1 del arreglo numerosGuardados +1*/
        respuestasDadas[2] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1] + 1);
    }
    //si el nombre del botón es igual a rop2dias_inpt
    else if (textoBot.name == "rop2dias_inpt")
    {
        /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
        la posición 1, si es que son iguales el puntaje sube en 1*/
        if (textoBot.text == alternaDias[1])
        {
            puntajeFinal++;
        }
        /*Y en la posición 3 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
        + la pregunta que se encuentre en la posición 1 del arreglo numerosGuardados +7*/
        respuestasDadas[3] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[1] + 7);
    }
```

Figura 329. Fragmento de código de la función Respondo() parte 1.

```
        //si el nombre del botón es igual a rop3_inpt
        else if (textoBot.name == "rop3_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 2, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaJera[2])
            {
                puntajeFinal++;
            }
            /*Y en la posición 4 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 2 del arreglo numerosGuardados +1*/
            respuestasDadas[4] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[2] + 1);
        }
        //si el nombre del botón es igual a rop3dias_inpt
        else if (textoBot.name == "rop3dias_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 2, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaDias[2])
            {
                puntajeFinal++;
            }
            /*Y en la posición 5 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 2 del arreglo numerosGuardados +7*/
            respuestasDadas[5] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[2] + 7);
        }
        //si el nombre del botón es igual a rop4_inpt
        if (textoBot.name == "rop4_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 3, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaJera[3])
            {
                puntajeFinal++;
            }
            /*Y en la posición 6 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 3 del arreglo numerosGuardados +1*/
            respuestasDadas[6] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[3] + 1);
        }
        //si el nombre del botón es igual a rop4dias_inpt
        else if (textoBot.name == "rop4dias_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 3, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaDias[3])
            {
                puntajeFinal++;
            }
            /*Y en la posición 7 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 3 del arreglo numerosGuardados +7*/
            respuestasDadas[7] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[3] + 7);
        }
```

Figura 330. Fragmento de código de la función Respondo() parte 2.

```
        //si el nombre del botón es igual a rop5_inpt
        if (textoBot.name == "rop5_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 4, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaJera[4])
            {
                puntajeFinal++;
            }
            /*Y en la posición 8 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 4 del arreglo numerosGuardados +1*/
            respuestasDadas[8] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[4] + 1);
        }
        //si el nombre del botón es igual a rop5dias_inpt
        else if (textoBot.name == "rop5dias_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 4, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaDias[4])
            {
                puntajeFinal++;
            }
            /*Y en la posición 9 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 4 del arreglo numerosGuardados +7*/
            respuestasDadas[9] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[4] + 7);
        }
        //si el nombre del botón es igual a rop6_inpt
        if (textoBot.name == "rop6_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 5, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaJera[5])
            {
                puntajeFinal++;
            }
            /*Y en la posición 10 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 5 del arreglo numerosGuardados +1*/
            respuestasDadas[10] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[5] + 1);
        }
        //si el nombre del botón es igual a rop6dias_inpt
        else if (textoBot.name == "rop6dias_inpt")
        {
            /*Comparo si el valor ingresado es igual a la respuesta que se tiene en
            la posición 5, si es que son iguales el puntaje sube en 1*/
            if (textoBot.text == alternaDias[5])
            {
                puntajeFinal++;
            }
            /*Y en la posición 11 de mi arreglo de respuestasDadas agrego el valor ingresado  + el tiempo de respuesta
            + la pregunta que se encuentre en la posición 5 del arreglo numerosGuardados +7*/
            respuestasDadas[11] = textoBot.text + "||" + tiempoText.text + "||" + (numerosGuardados[5] + 7);
        }
    }
```

Figura 331. Fragmento de código de la función Respondo() parte 3.

```
    public void Post()
    {
        var request = new UnityWebRequest("https://simulab.edutech-project.org/api/registrarActividad", "POST");
        string auxJSON = JsonUtility.ToJson(ejercicio);
        byte[] bodyRaw = Encoding.UTF8.GetBytes(auxJSON);
        request.uploadHandler = (UploadHandler)new UploadHandlerRaw(bodyRaw);
        request.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
        request.SetRequestHeader("Content-Type", "application/json");
        request.SendWebRequest();

    }
```

Figura 332. Función Post().

```
    public static void MenuSel(string tipoMen)
    {
        GeneralEmpezar.menu = tipoMen;
    }
```

Figura 333. Función MenuSel().

```
[Serializable]
public class Pregunta
{
    public string respuestaIngresada;
    public string tiempoRespuesta;
    public int numeroPregunta;
    public DateTime inicio;
    public DateTime fin;
    public string getRespuetaIngresada()
    {
        return respuestaIngresada;
    }
    public string getTiempoRespuesta()
    {
        return tiempoRespuesta;
    }
    public void setNumeroPregunta(int numeroPregunta)
    {
        this.numeroPregunta = numeroPregunta;
    }
    public int getPregunta()
    {
        return numeroPregunta;
    }
    public void setInicio()
    {
        this.inicio = System.DateTime.Now;
    }
    public void setFin()
    {
        this.fin = System.DateTime.Now;
    }
    public void setTiempoRespuesta(string tiempoRespuesta)
    {
        this.tiempoRespuesta = tiempoRespuesta;
    }
    public void setRespuestaIngresada(string respt)
    {
        this.respuestaIngresada = respt;
    }
}
```

Figura 334. Clase Pregunta.

```
[Serializable]
public class SimuladorData
{
    public string correo;
    public int numeroEjercitario;
    public string tiempoInicio;
    public string tiempoFin;
    public string fechaDeActividad;
    public List<Pregunta> preguntas = new List<Pregunta>();
    public void ReinicioPreguntas()
    {
        preguntas = new List<Pregunta>();
    }
    public void setNumeroEjercitario(int numero)
    {
        this.numeroEjercitario = numero;
    }
    public int getNumeroEjercitario()
    {
        return this.numeroEjercitario;
    }
    public void setTiempoInicio(string tiempoInicio)
    {
        this.tiempoInicio = tiempoInicio;
    }
    public string getTiempoInicio()
    {
        return this.tiempoInicio;
    }
    public void setTiempoFin(string tiempoFin)
    {
        this.tiempoFin = tiempoFin;
    }
    public string getTiempoFin()
    {
        return this.tiempoFin;
    }
    public void setFechaActividad(string fechaDeActividad)
    {
        this.fechaDeActividad = fechaDeActividad;
    }
    public string getFechaActividad()
    {
        return this.fechaDeActividad;
    }
    public void setCorreo(string correo)
    {
        this.correo = correo;
    }
    public string getCorreo()
    {
        return this.correo;
    }
    public void addPregunta(Pregunta nuevaPreg)
    {
        this.preguntas.Add(nuevaPreg);
    }
    public void updatePregunta(Pregunta upPreg, Pregunta antPreg)
    {
        this.preguntas.Remove(antPreg);
        this.preguntas.Add(upPreg);
    }
    public Pregunta readPregunta(int indice)
    {
        return preguntas[indice];
    }
    public void setInicioPregunta(int numero)
    {
        this.preguntas[numero].setInicio();
    }
}
```

Figura 335. Clase SimuladorData.

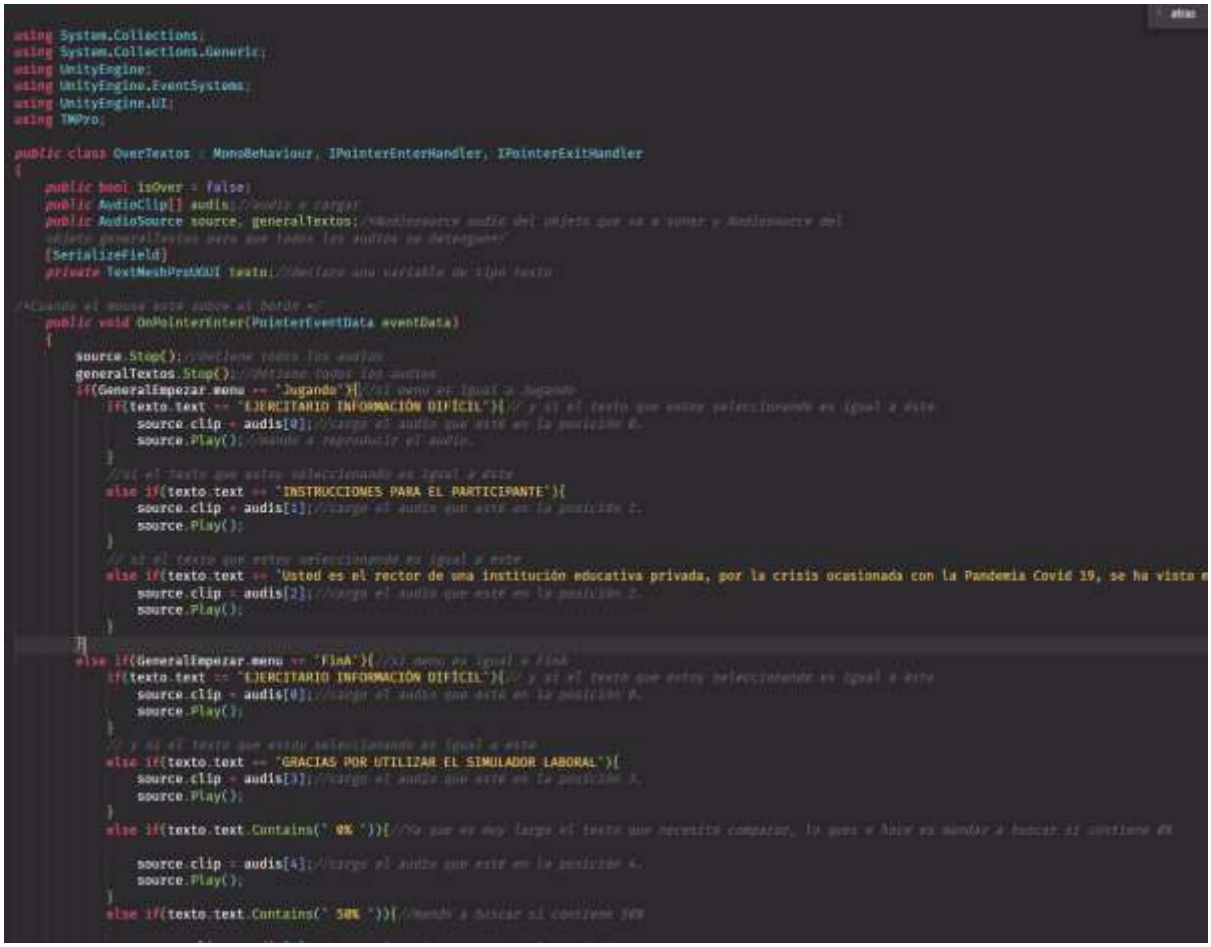

```
{"correo":"jbarrerab1@est.ups.edu.ec","numeroEjercitario":5,
"tiempoInicio":"09:24:58","tiempoFin":"09:25:47",
"fechaDeActividad":"2022-04-06 09:25:47","preguntas":
[{"respuestaIngresada":"1","tiempoRespuesta":"00:23:08","numeroPregunta":3},
{"respuestaIngresada":"5","tiempoRespuesta":"00:01:63","numeroPregunta":9},
{"respuestaIngresada":"3","tiempoRespuesta":"00:01:05","numeroPregunta":2},
{"respuestaIngresada":"7","tiempoRespuesta":"00:00:45","numeroPregunta":8},
{"respuestaIngresada":"6","tiempoRespuesta":"00:01:75","numeroPregunta":6},
{"respuestaIngresada":"10","tiempoRespuesta":"00:01:84","numeroPregunta":12},
{"respuestaIngresada":"1","tiempoRespuesta":"00:01:94","numeroPregunta":1},
{"respuestaIngresada":"1","tiempoRespuesta":"00:01:51","numeroPregunta":7},
{"respuestaIngresada":"1","tiempoRespuesta":"00:00:42","numeroPregunta":4},
{"respuestaIngresada":"1","tiempoRespuesta":"00:01:13","numeroPregunta":10},
{"respuestaIngresada":"1","tiempoRespuesta":"00:01:68","numeroPregunta":5},
{"respuestaIngresada":"1","tiempoRespuesta":"00:01:95","numeroPregunta":11}]}
```


Figura 336. Estructura de preguntas que se genera en el archivo json.

1.5.2. Script MouseOver

Figura 337. Fragmento de código de funciones para reproducir audio cuando el mouse está sobre un botón.

1.5.3. Script OverTextos.cs



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;

public class OverTextos : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler
{
    public bool isOver = false;
    public AudioClip[] audis;//audio a cargar
    public AudioSource source, generalTextos;//Audiosource audio del objeto que se a sumar y Audiosource del
    objeto generalTextos para que todos los audios se detengan
    [SerializeField]
    private TextMeshProUGUI texto;//declaro una variable de tipo texto

//Cuando el mouse este sobre el boton
    public void OnPointerEnter(PointerEventData eventData)
    {
        source.Stop();//detiene todos los audios
        generalTextos.Stop();//detiene todos los audios
        if(GeneralEmpezar.menu == "Jugando"){//si menu es igual a Jugando
            if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
                source.clip = audis[0];//cargo el audio que este en la posición 0.
                source.Play();//mando a reproducir el audio.
            }
            //si el texto que estoy seleccionando es igual a este
            else if(texto.text == "INSTRUCCIONES PARA EL PARTICIPANTE"){
                source.clip = audis[1];//cargo el audio que este en la posición 1.
                source.Play();
            }
            // si el texto que estoy seleccionando es igual a este
            else if(texto.text == "Usted es el rector de una institución educativa privada, por la crisis ocasionada con la Pandemia Covid 19, se ha visto e
                source.clip = audis[2];//cargo el audio que este en la posición 2.
                source.Play();
            }
        }
        else if(GeneralEmpezar.menu == "Fina"){//si menu es igual a Fina
            if(texto.text == "EJERCITARIO INFORMACIÓN DIFÍCIL"){// y si el texto que estoy seleccionando es igual a este
                source.clip = audis[0];//cargo el audio que este en la posición 0.
                source.Play();
            }
            // y si el texto que estoy seleccionando es igual a este
            else if(texto.text == "GRACIAS POR UTILIZAR EL SIMULADOR LABORAL"){
                source.clip = audis[3];//cargo el audio que este en la posición 3.
                source.Play();
            }
            else if(texto.text.Contains(" 0% ")){//Ya que es muy largo el texto que necesito comparar, lo que e hace es mandar a buscar si contiene 0%

                source.clip = audis[4];//cargo el audio que este en la posición 4.
                source.Play();
            }
            else if(texto.text.Contains(" 50% ")){//mando a buscar si contiene 50%
```

Figura 338. Fragmento de código del script OverTextos.cs.

1.5.4. Script MouseColor.cs

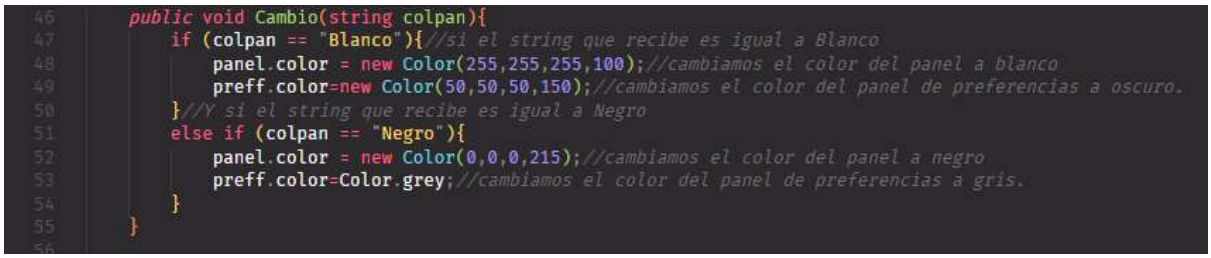
Figura 339. Función que al estar el mouse sobre un botón cambia de color.

Figura 340. Función que al salir el mouse del botón cambia al color por defecto.

1.5.5. Script PreferSctipt.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.UI;
using System.IO;

public class PreferSctipt : MonoBehaviour
{
    [SerializeField]
    private Image[] botones;//arreglo de imagenes utilizado para cambiar el color de los botones
    [SerializeField]
    /*arreglo de textos utilizado para cambiar el color del texto,
     cambio fuente de texto y cambio de tamaño*/
    private TextMeshProUGUI[] objeto,pregs,objPrefer, objDet;
    [SerializeField]
    //arreglo de textos el cual contiene las fuentes a utilizar
    private TextMeshProUGUI[] tiposLetra;
    [SerializeField]
    //arreglo de textos el cual contiene los titulos de contrastes a usar
    private TextMeshProUGUI tem1, tem2;
    [SerializeField]
    /*arreglo de imagenes el cual contiene el panel,
    el boton de desactivar audio y el panel de preferencias*/
    private Image panel, muteBtn,preff;
    //variable de tipo boton
    public Button botMute;
    public AudioSource[] sonidoAS;
    //variable de tipo texto del boton silenciar.
    public TextMeshProUGUI silbtn;
    //variable estatica para controlar en que menu estamos.
    public static string menu="Main";
    //variable estatica para controlar el tipo de contraste
    public static string colorBt="Negro";
    // variable estatica para controlar el audio.
    public static bool isMute = false;
    [SerializeField]
    //se crea un arreglo de imagenes de los botones principales
    private Image[] BotonesImagenes;
    //se crea dos sprites para controlar el activar y desactivar el audio.
    public Sprite ImagenMute, ImagenMuteS;
    //variable booleana para controlar cuando la fuente Dyslexic este activa.
    private bool Disle=false;
```

Figura 341. Declaración de variables.



```
    public void Cambio(string colpan){
        if (colpan == "Blanco"){//si el string que recibe es igual a Blanco
            panel.color = new Color(255,255,255,100);//cambiamos el color del panel a blanco
            preff.color=new Color(50,50,50,150);//cambiamos el color del panel de preferencias a oscuro.
        }//Y si el string que recibe es igual a Negro
        else if (colpan == "Negro"){
            panel.color = new Color(0,0,0,215);//cambiamos el color del panel a negro
            preff.color=Color.grey;//cambiamos el color del panel de preferencias a gris.
        }
    }
```

Figura 342. Función para cambiar el color del panel.

```
/*Funcion que permite sileciar la salida de audio del simulador*/
    public void MuteAll(){
        //si la variable estatica es igual a false y colorBt es blanco
        if(isMute == false && PreferSctipt.colorBt == "Blanco"){
            //mandamos a sileciar todos los audios.
            foreach (var s in sonidoAS){
                s.volume = 0f;
            }
            isMute = true;
            //si esta desactivado el audio asignamos la imagen ImagenMute al botón
            botMute.image.sprite = ImagenMute;
            muteBtn.color = Color.yellow;//cambiamos a amarillo
        }
        //si la variable estatica es igual a false y colorBt es Negro
        else if(isMute == false && PreferSctipt.colorBt == "Negro"){
            //mandamos a sileciar todos los audios.
            foreach (var s in sonidoAS){
                s.volume = 0f;
            }
            isMute = true;
            botMute.image.sprite = ImagenMute;
            muteBtn.color = Color.white;
        }
        else {//caso contrario
            //mandamos a activar la salida de audios.
            foreach (var s in sonidoAS){
                s.volume = 1f;
            }
            isMute = false;
            //Si colorBt es blanco
            if(PreferSctipt.colorBt == "Blanco"){
                //cambiamos la imagen
                botMute.image.sprite = ImagenMuteS;
                //cambiamos a blanco
                muteBtn.color = Color.white;
            }
            else{
                botMute.image.sprite = ImagenMuteS;
                //caso contrario lo cambia a amarillo
                muteBtn.color = Color.yellow;
            }

        }
    }
```

. Función para desactivar todos los audios.

```
/*Función que recibe como parametro un string dicha función permite cambiar el color de los botones*/
    public void CambioColBot(string te){
        //si el string que recibe es igual a Blanco
        if (te == "Blanco"){
            //cambia todos los botones a blanco
            foreach(var b in botones){
                b.color = Color.white;
                PreferSctipt.ColB("Blanco");
            }
            foreach(var b in botonesImagenes){
                b.color = Color.white;
            }
            foreach(var b in pregs){
                b.color = Color.white;
            }
        }
        //si el string que recibe es igual a Negro
        else if (te == "Negro"){
            //cambia todos los botones a negro
            foreach(var b in botones){
                b.color = Color.black;
                PreferSctipt.ColB("Negro");
            }
            //los botones principales los cambiamos a amarillo
            foreach(var b in botonesImagenes){
                b.color = Color.yellow;
            }
        }
    }
```

Figura 343. Función para cambiar el color de los botones.

```
/*Esta funcion recibe como parametro un string dicha función permite cambiar el color del texto*/
    public void CambioColText(string te){
        //si el string que recibe es igual a Negro
        if (te == "Negro"){
            /*cambia la letra a azul*/
            foreach (var obj in objeto) {
                obj.color= Color.blue;
            }
            foreach (var obj in tiposLetra) {
                obj.color= Color.blue;
            }
            foreach (var obj in pregs) {
                obj.color= Color.blue;
            }
            foreach (var obj in objPrefer) {
                obj.color= Color.blue;
            }
        }
        //si el string que recibe es igual a Amarillo
        if (te == "Amarillo"){
            /*cambia la letra a amarillo*/
            foreach (var obj in objeto) {
                obj.color = Color.yellow;
            }
            foreach (var obj in tiposLetra) {
                obj.color = Color.yellow;
            }
            foreach (var obj in pregs) {
                obj.color= Color.yellow;
            }
            foreach (var obj in objPrefer) {
                obj.color=Color.yellow;
            }
        }
    }
```

Figura 344. Función para cambiar el color del texto de todos los componentes de la interfaz.

```
/*Esta función permite recibe como parámetro un string al cual se le pasa la fuente, esta función
 permite personalizar el texto de la interfaz haciendo uso de tres tipos de fuente de texto*/
    public void cambioFuente(TMP_FontAsset fu){
        //si la fuente que recibe es igual a
        if(fu.ToString() == "OpenDyslexic-Regular SDF (TMPro.TMP_FontAsset)"){
            /*se define un tamaño para cuando esté activa esta fuente.*/
            foreach(var t in objeto){
                t.fontSize = 19f;
            }
            foreach(var t in pregs){
                t.fontSize = (18f);
            }
            foreach(var t in objPrefer){
                t.fontSize = (17f);
            }

            Disle=true;//si está activa
        }
        else{
            Disle=false;// si esta desactivada
        }
            /*cambia de fuente a todos los textos de la interfaz*/
            foreach(var t in objeto){
                t.font = fu;
            }
            foreach(var t in pregs){
                t.font = fu;
            }
            foreach(var t in objPrefer){
                t.font = fu;
            }
            //cambia el tipo de fuente
            tem1.font = fu;
            tem2.font = fu;
    }

```

Figura 345. Función para cambiar la fuente texto de toda la interfaz.

```
/*Funció que recibe como parámetro un entero dicha funcion permite reducir y aumentar el tamaño de texto*/
    public void cambioTamano(int tam){

        if(Disle=true){//si esta activa dicha fuente
            /*del tamaño definido restamos -2 */
            foreach(var t in objeto){
                t.fontSize = tam-2;
            }
            /*del tamaño definido restamos -7 a objetos de preferencias*/
            foreach(var t in objPrefer){
                t.fontSize = tam-7;
            }
            /*del tamaño definido restamos -7 a preguntas */
            foreach(var t in pregs){
                t.fontSize = (tam-7);
            }
            //cambia el tamaño
            tem1.fontSize = tam;
            tem2.fontSize = tam;
        }

        else{//si Disle es igual a false
            /*cambiamos el tamaño de texto a todos los objetos*/
            foreach(var t in objeto){
                t.fontSize = tam;
            }
            foreach(var t in pregs){
                t.fontSize = (tam);
            }
            foreach(var t in objPrefer){
                t.fontSize = tam;
            }//cambia el tamaño
            tem1.fontSize = tam;
            tem2.fontSize = tam;
        }


    }
/*Función que permite detener todos los sonidos cuando se descative el audio */
```

Figura 346. Función para cambiar el tamaño de texto.

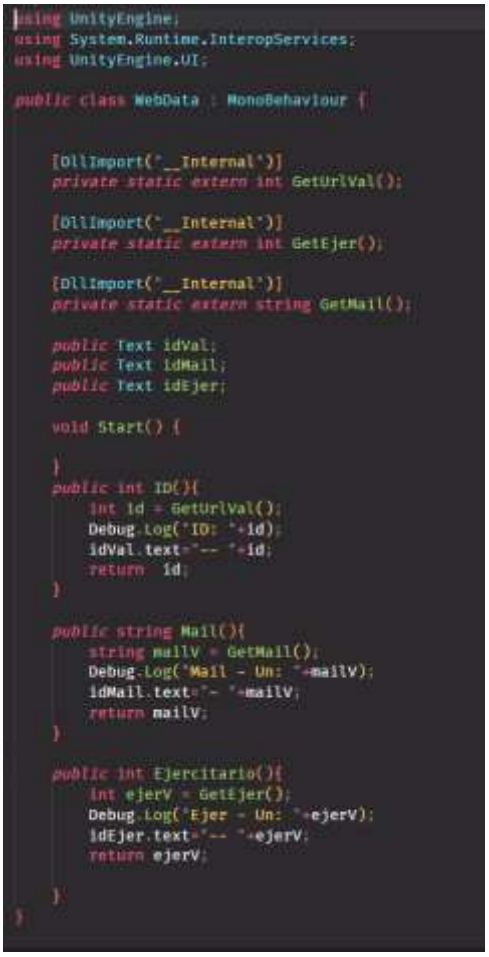
```
    public static void MenuSel(string tipoMen){
        Debug.Log("Eras "+PreferSctipt.menu+" ahora eres "+tipoMen);
        PreferSctipt.menu = tipoMen;
    }

```

Figura 347. Función para controlar en que panel se posiciona.

```
    public static void ColB(string col){
        PreferSctipt.colorBt = col;
    }
```

Figura 348. Función que controla el cambio de color de los botones según el color del panel que se vaya a definir.

1.5.6. Script WebData.cs



```
using UnityEngine;
using System.Runtime.InteropServices;
using UnityEngine.UI;

public class WebData : MonoBehaviour {


    [DllImport("__Internal")]
    private static extern int GetUrlVal();

    [DllImport("__Internal")]
    private static extern int GetEjer();

    [DllImport("__Internal")]
    private static extern string GetMail();

    public Text idVal;
    public Text idMail;
    public Text idEjer;

    void Start() {

    }
    public int ID(){
        int id = GetUrlVal();
        Debug.Log("ID: "+id);
        idVal.text="-- "+id;
        return  id;
    }

    public string Mail(){
        string mailV = GetMail();
        Debug.Log("Mail - Un: "+mailV);
        idMail.text="- "+mailV;
        return mailV;
    }

    public int Ejercitario(){
        int ejerV = GetEjer();
        Debug.Log("Ejer - Un: "+ejerV);
        idEjer.text="-- "+ejerV;
        return ejerV;

    }
}
```

Figura 349. Script WebData para los servicios web.

1.6. Directorio del proyecto de cada Simulador Laboral

1) Carpeta que contiene las animaciones creadas para el proyecto.
2) Carpeta que contiene los audios utilizados para la descripción de textos.
3) Carpeta que contiene las imágenes utilizadas para los botones.
4) Carpeta que contiene imágenes utilizadas para animar objetos del escenario(laptop).
5) Carpeta que contiene las fuentes de texto utilizadas.
6) Carpeta que contiene materiales utilizados en el escenario.
7) Carpeta que contiene al personaje del escenario.
8) Carpeta que contiene el archivo para la comunicación de la API de Unity con el servidor web.
9) Carpeta que contiene la escena del proyecto.
10) Carpeta que contiene los scripts generados para el proyecto.

Figura 350. Directorio del proyecto.

1.7 Construyendo y ejecutando los simuladores laborales en WebGL

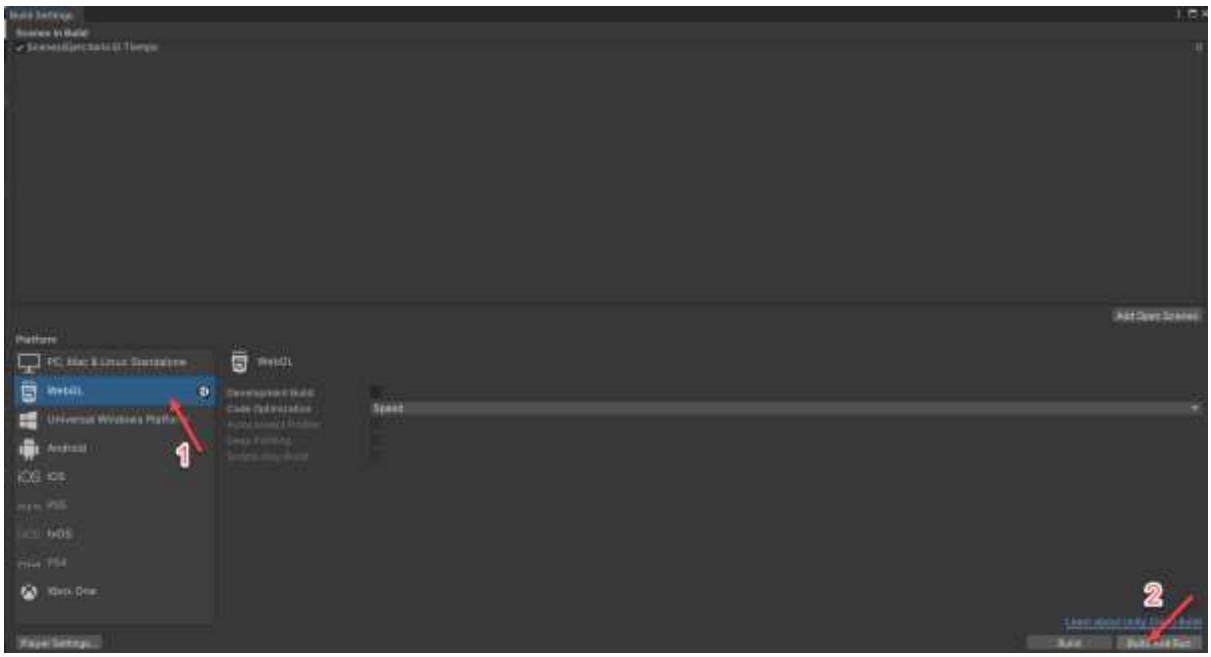


Figura 351. Construcción a WebGL.

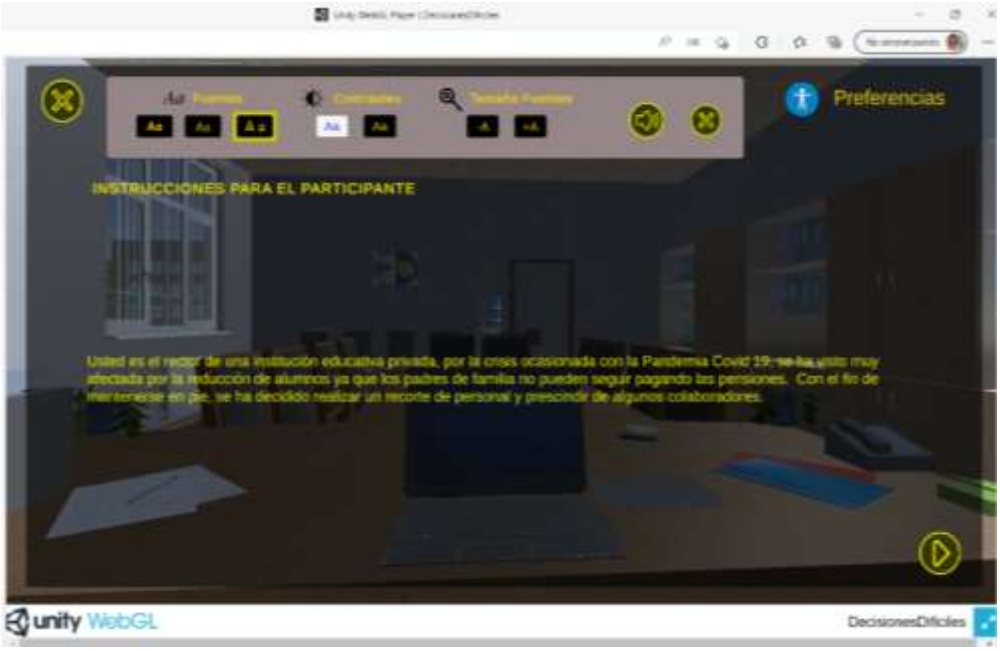


Figura 352. Ejecutando en WebGL Simulador Información Difícil.

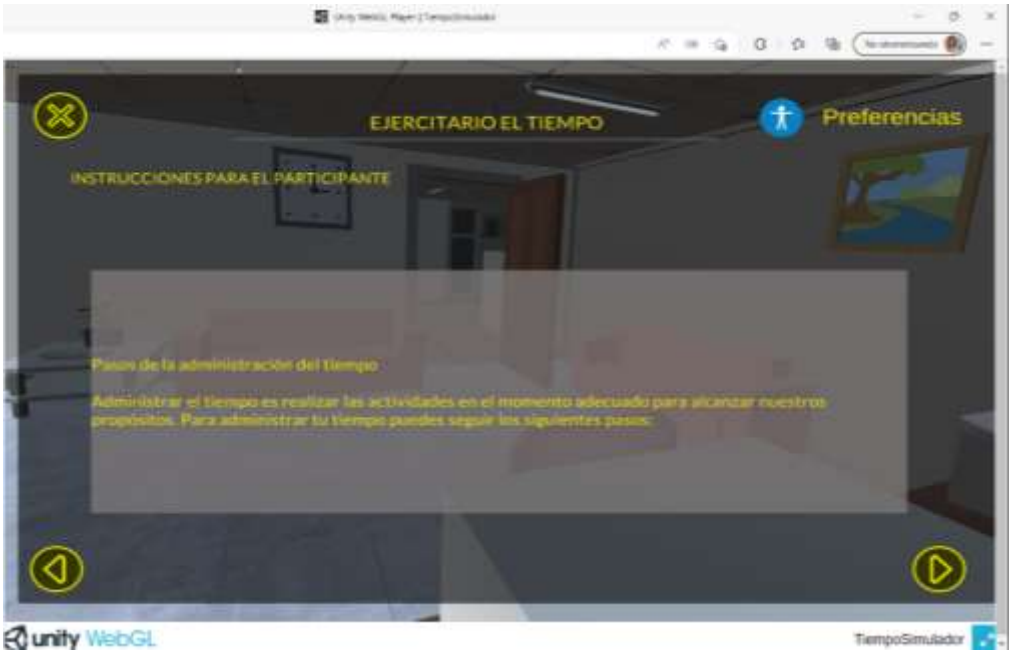


Figura 353. Simulador Laboral El Tiempo.

Anexos F

Manual de Usuario de Simuladores Laborales 3D

" DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE SIMULADORES LABORALES 3D, BASADO EN EJERCITARIOS EL TIEMPO E INFORMACIÓN DIFÍCIL, DENTRO DEL MARCO DEL PROYECTO EDUTECH"

Manual de usuario orientado al uso y funcionamiento de los simuladores laborales 3D dirigido a usuarios con discapacidad y para aquellas personas que las asistan, la intención es ayudarlos a entender los simuladores laborales 3D desarrollados.

Est. Narcisa de Jesús Araujo Pérez
naraujop@est.ups.edu.ec

Los simuladores laborales 3D pueden ser interactuados mediante el mouse o teclado. Si interactúa con el teclado se necesita las teclas que se resaltan a continuación.

Figura 354. Para navegar por el simulador con los botones del lado derecho, utiliza los botones resaltados de amarillo.

Figura 355. Para navegar por el simulador con los botones del lado izquierdo, utiliza los botones resaltados de amarillo.

Orden de pantallas en el simulador

A continuación, se muestra el flujo de las ventanas para conocer el orden en el que se van a ir abriendo.

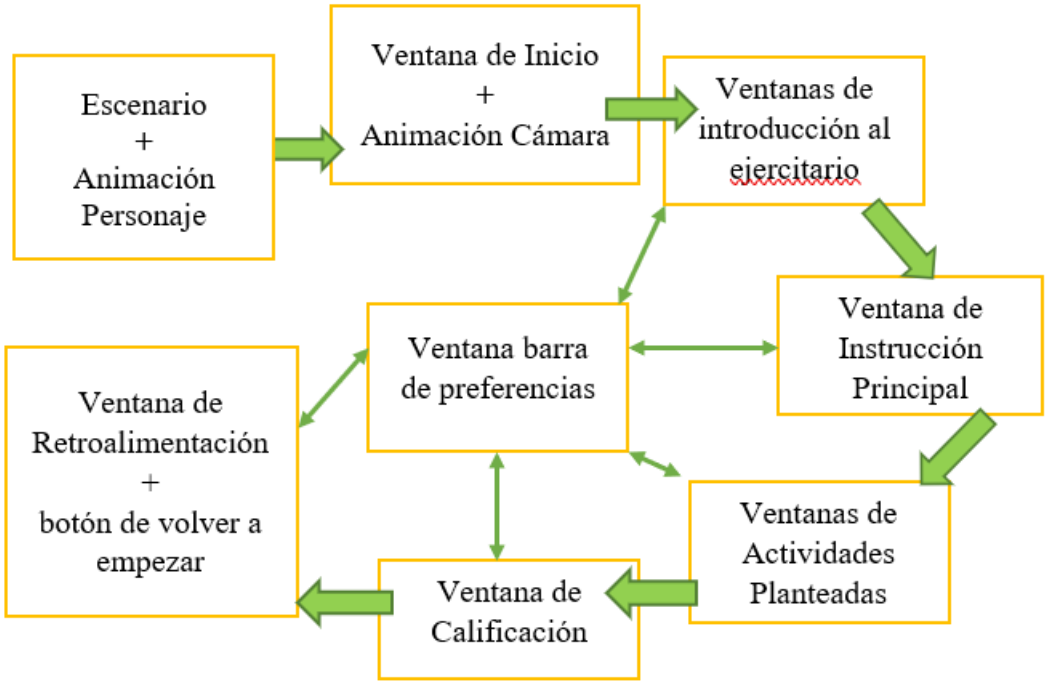


Figura 356. Flujo de ventanas de simuladores laborales.

Comenzar a interactuar

Una vez elegido el simulador laboral se mostrará en pantalla una barra de cargando, enseguida termine de cargar el simulador aparecerá en pantalla un personaje el cual simulará la interacción con la situación real del ejercitario(ejercicio) planteado.

Prueba de controles

En ambos casos, tanto si desea interactuar con el mouse o el teclado se resaltará el botón en el que esté posicionado, acompañado del audio correspondiente como se muestra a continuación.

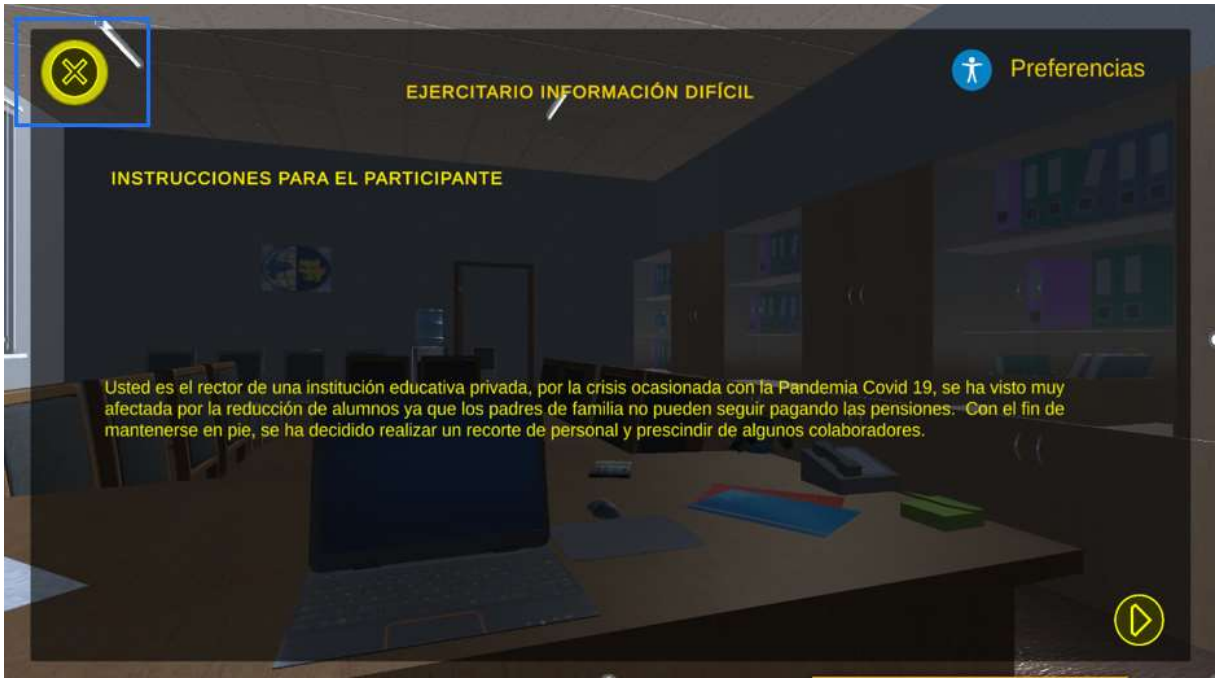


Figura 357. Ventana en donde se observa que botón se resalta de amarillo al posicionarse en él.

Interfaz de Usuario Simulador Laboral El Tiempo

Una vez que termine la animación del personaje se mostrará la siguiente ventana en donde el participante podrá mover mediante el mouse la vista actual de la cámara, una vez que de clic en el botón “empezar” se desactivará el movimiento de la cámara y se pasará a la ventana de instrucciones.

Figura 358. Ventana Principal en donde se muestra el botón de empezar.

En la siguiente ventana se presenta información referente al ejercitario, información que se le servirá al participante para poder interactuar con las actividades que se le planteen, adicional se habilita el botón *"atrás"* que al dar clic le permitirá volver a la ventana principal, en caso de dar clic en el botón "*siguiente*" se pasará a la siguiente ventana de instrucciones, una vez que haya revisado todas las instrucciones pasará a la ventana de actividades.

Figura 359. Ventana de instrucciones para el participante.

En la siguiente ventana se muestra las actividades planteadas para el participante, el objeto del lado izquierdo que se señala en un cuadro azul corresponde a la actividad planteada y campos de entrada de texto del lado derecho que se señala con unos cuadros amarillos corresponde a los espacios en donde el participante debe ingresar las respuestas de la actividad planteada.

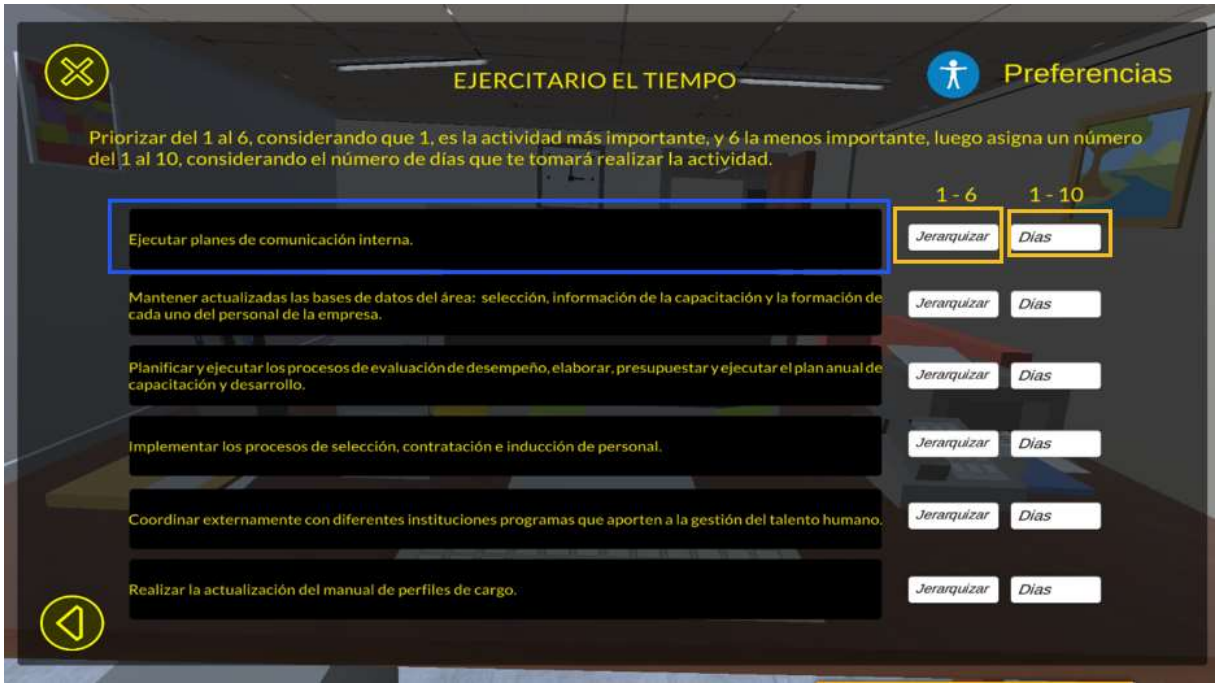


Figura 360. Vista panel en donde se indica las actividades planteadas con las que deberá interactuar el participante.

En la siguiente ventana se puede observar que cuando se interactúa con el teclado y al estar posicionado en una actividad se deshabilita los campos de respuesta, lo cual evita que se ingresen valores inconscientemente siendo así de mucha ayuda para aquella persona con discapacidad visual.

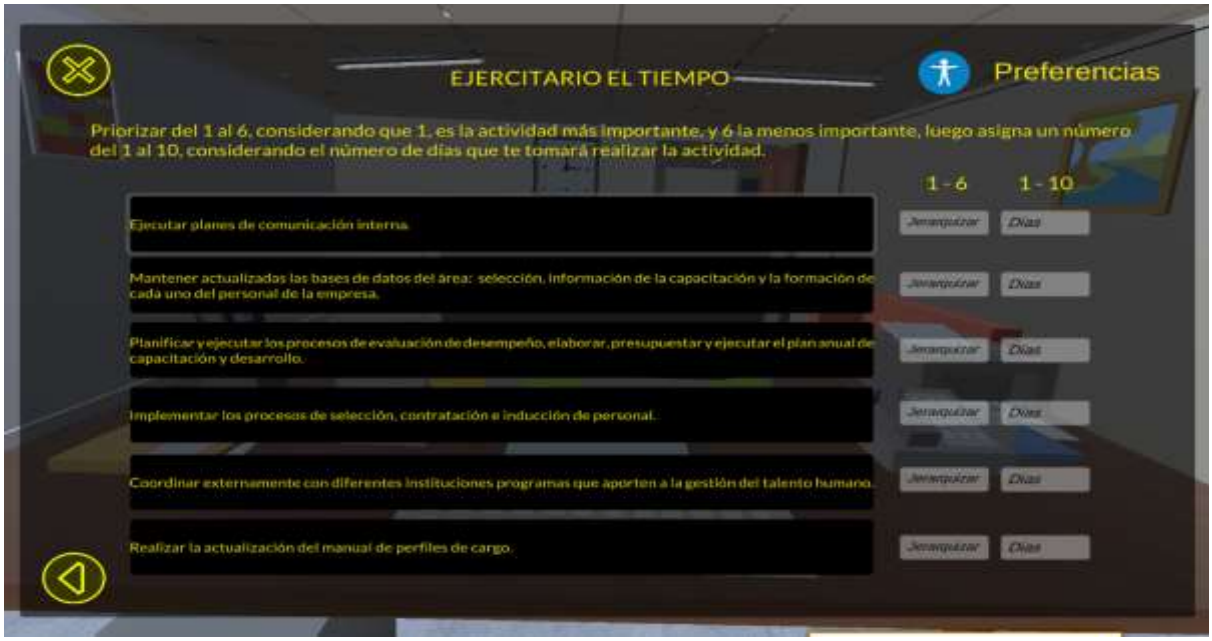


Figura 361. Vista en donde se inactivan los campos de texto de las respuestas al interactuar con el teclado.

A continuación, se muestra el ingreso de respuestas por el participante, es importante mencionar que cada vez que se ingresen valores erróneos se reproducirá un audio alertándole al participante, también es importante mencionar que el botón *"siguiente"* solo se activará si el participante ha completado todas las respuestas.

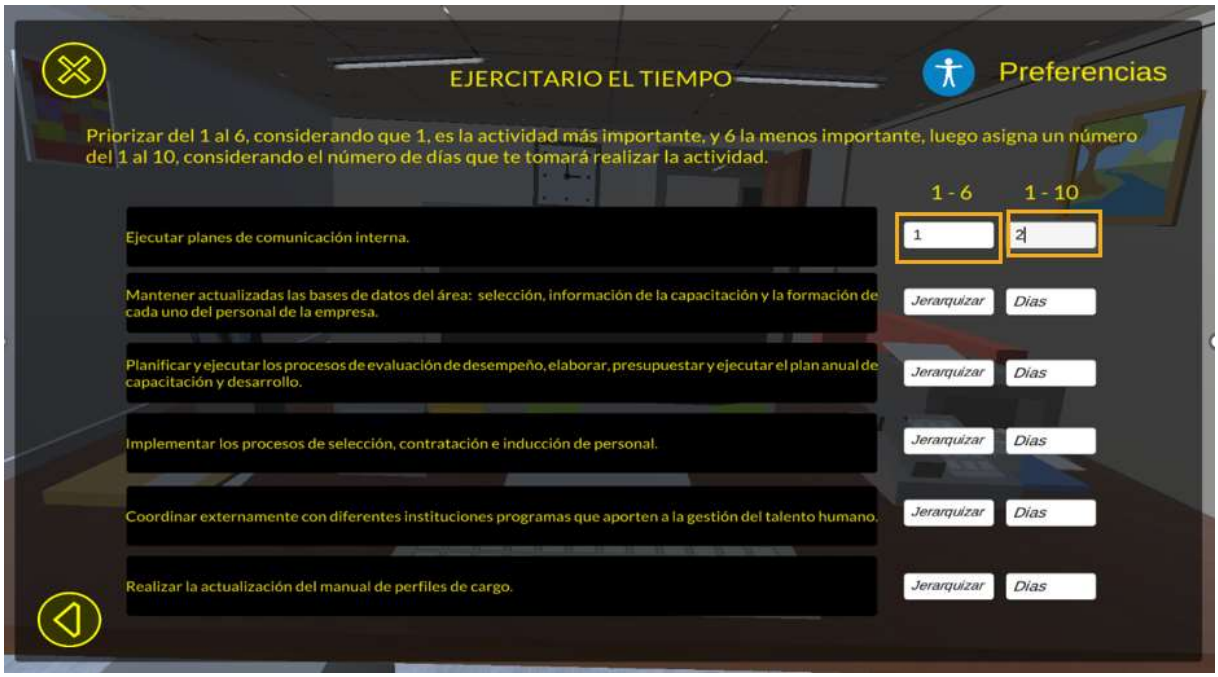


Figura 362. Vista de ingreso de respuestas del participante.

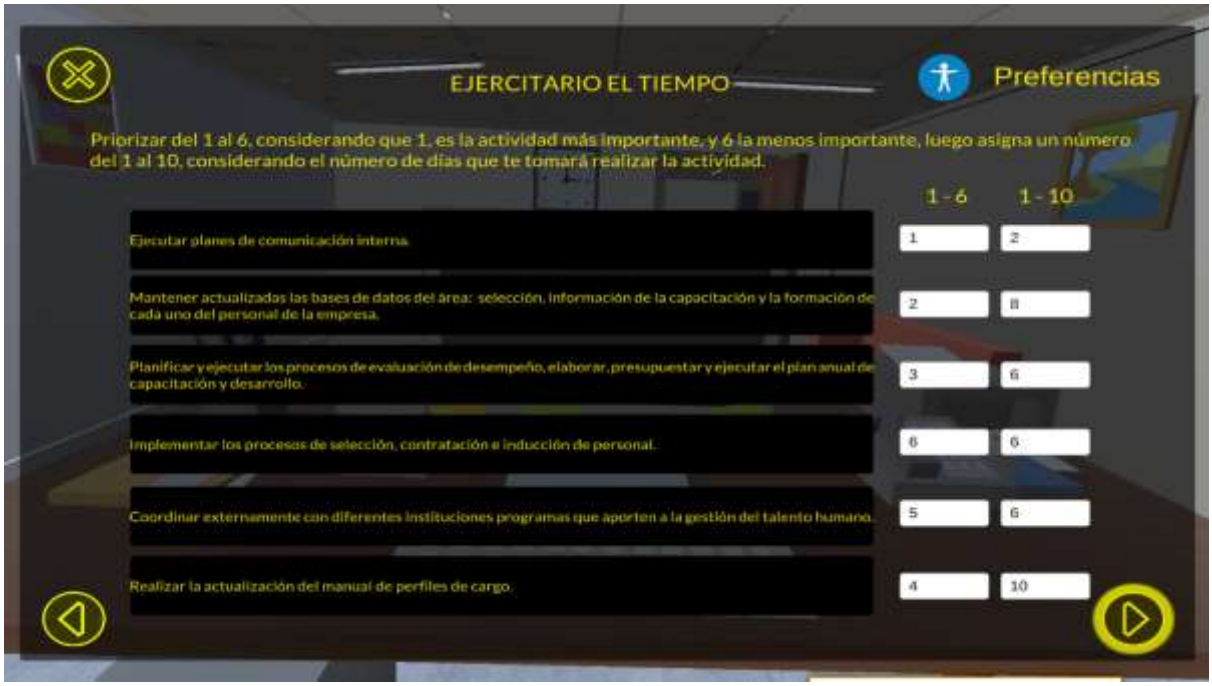


Figura 363. Vista en donde en participante ha completado todas las respuestas y por ello se activa el botón siguiente.

En la siguiente ventana se muestra la calificación obtenida luego de haber interactuado con todas las actividades planteadas, luego el participante podrá pasar a la siguiente ventana dando clic o seleccionando el botón *"siguiente*".

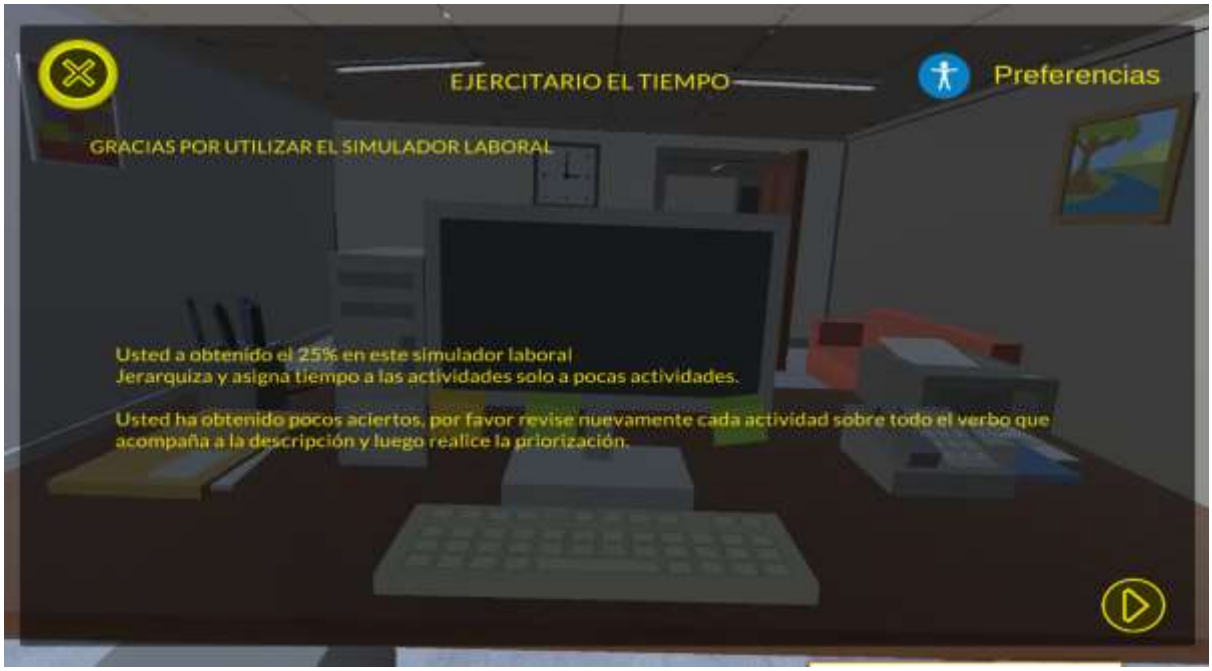


Figura 364. Vista panel de calificación luego de que el participante haya finalizado las actividades planteadas.

En la siguiente ventana se presenta al participante la retroalimentación del ejercitario con el que ha interactuado, también muestra un botón con una imagen de casa el cual al dar clic pasará a la ventana principal en donde podrá volver a empezar la interacción.

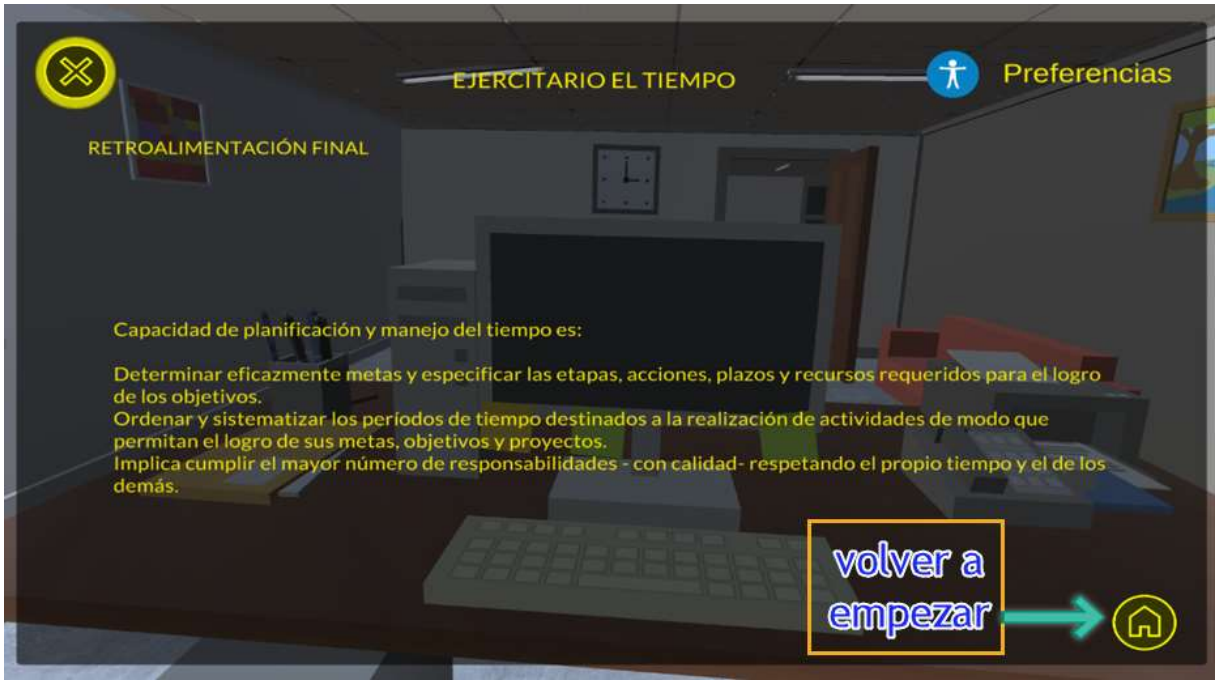


Figura 365. Ventana de retroalimentación.

Interfaz de Usuario Simulador Laboral Información Difícil

Una vez que termine la animación del personaje se mostrará la siguiente ventana en donde el participante podrá mover mediante el mouse la vista actual de la cámara, una vez que de clic en el botón *"empezar"* se desactivará el movimiento de la cámara y se pasará a la ventana de instrucciones.

Figura 366. Ventana Principal en donde se muestra el botón de empezar.

En la siguiente ventana se presenta información referente al ejercitario, información que se le servirá al participante para poder interactuar con las actividades que se le planteen, luego de dar clic en el botón "*siguiente*" se pasará a la ventana de actividades.

Figura 367. Ventana de instrucciones para el participante.

En la siguiente ventana se muestra las actividades planteadas al participante, en donde deberá dar clic en la opción que considere correcta según sea el caso, una vez que se seleccione una de las opciones automáticamente pasará a la siguiente pregunta, en esta ventana también se habilita el botón "*atrás*" en caso de que el participante necesite volver a revisar las instrucciones del ejercitario.

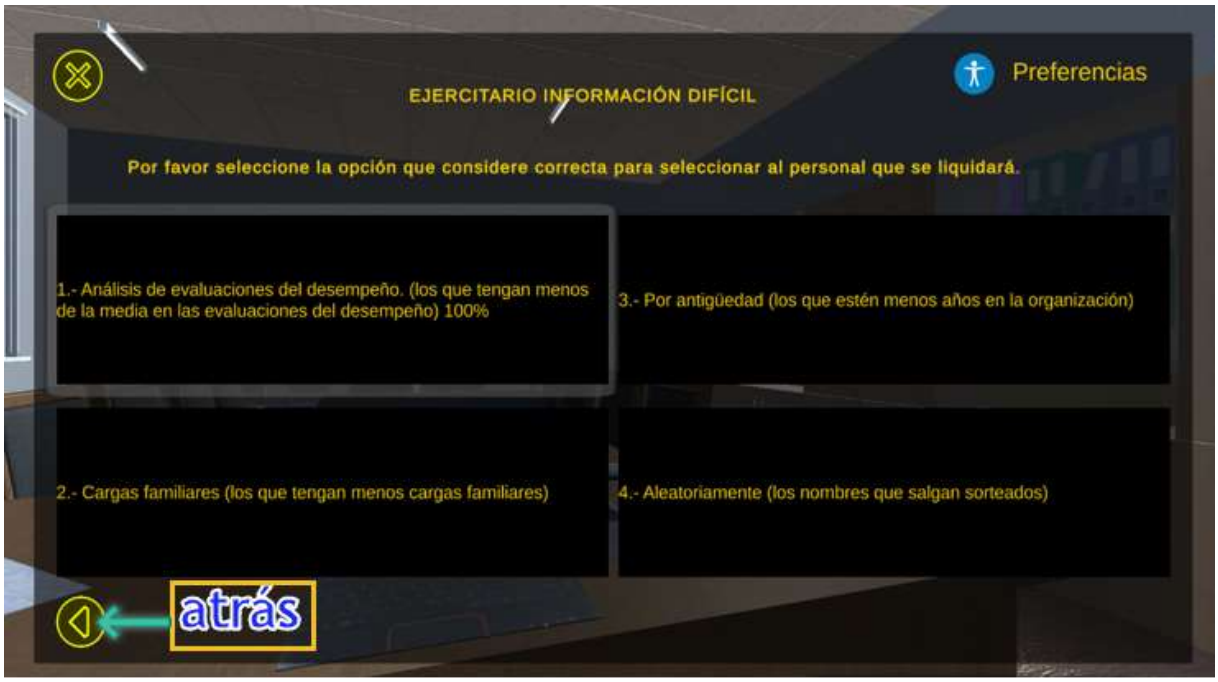


Figura 368. Ventana en donde el participante interactúa con las actividades.

En la siguiente ventana se muestra la calificación obtenida luego de haber interactuado con todas las actividades planteadas, luego el participante podrá pasar a la siguiente ventana dando clic o seleccionando el botón *"siguiente*".

Figura 369. Ventana en donde el participante revisa la calificación obtenida.

En la siguiente ventana se presenta al participante la retroalimentación del ejercitario con el que ha interactuado, también muestra un botón con una imagen de casa el cual al dar clic pasará a la ventana principal en donde podrá volver a empezar la interacción.

Figura 370. Ventana de retroalimentación.

Interactuando con las opciones de accesibilidad de la barra de preferencias de los simuladores laborales

En la siguiente ventana se muestra la barra de preferencias la cual se activa al presionar el botón “preferencias” en donde el participante podrá interactuar con todas las opciones de accesibilidad en cada ventana que se muestre el botón *“preferencias”*. A continuación, se activa la fuente de texto “Arial Bold” para seleccionar cada una de las opciones debe dar clic sobre la opción que se desee como se muestra.

Figura 371. Vista en donde el participante selecciona la fuente de texto Arial Bold.

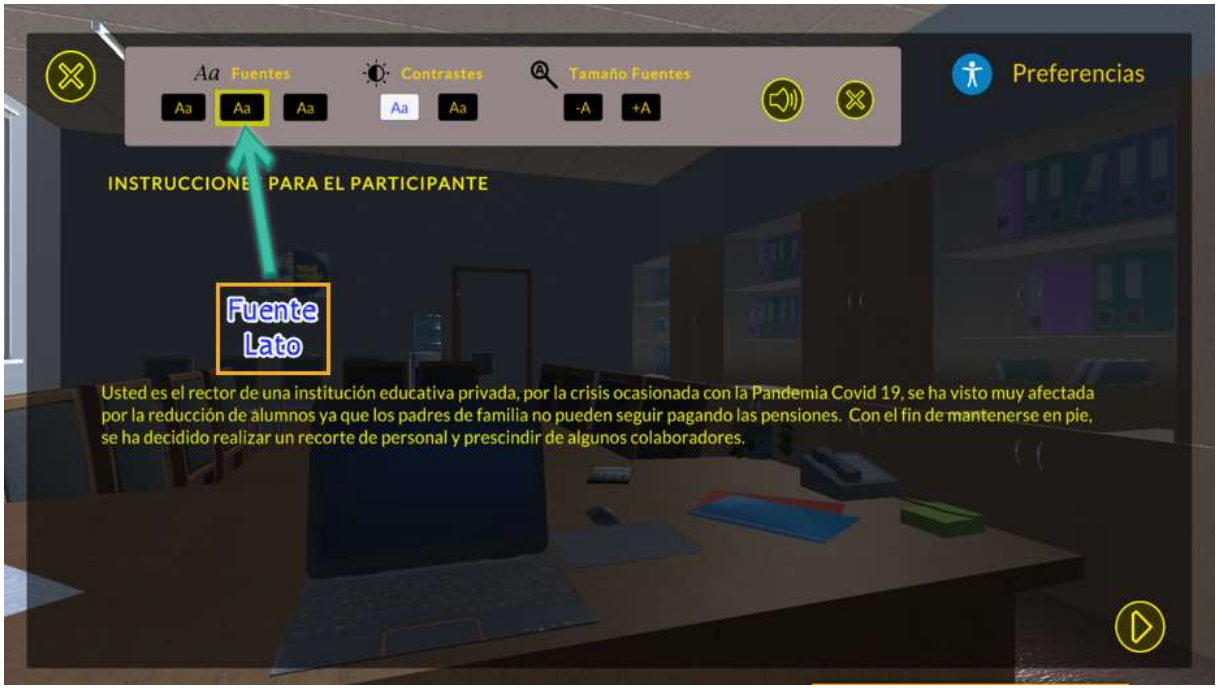


Figura 372. Vista en donde el participante selecciona la fuente de texto Lato.

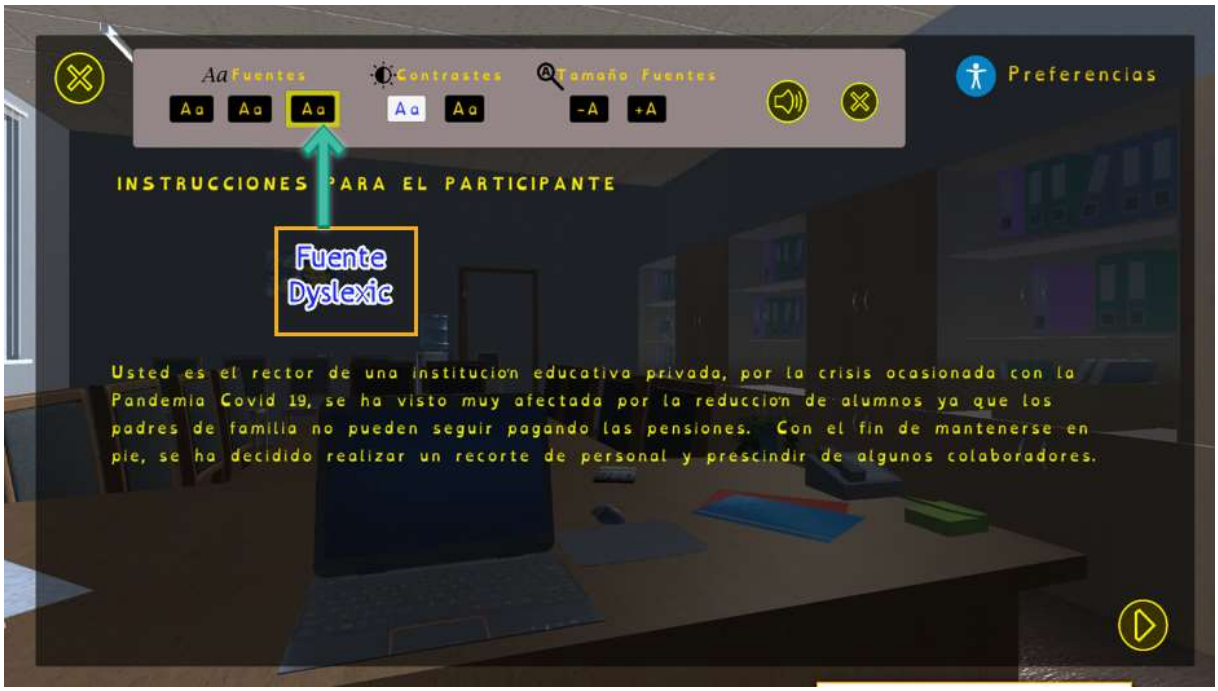


Figura 373. Vista en donde el participante selecciona la fuente Dyslexic.

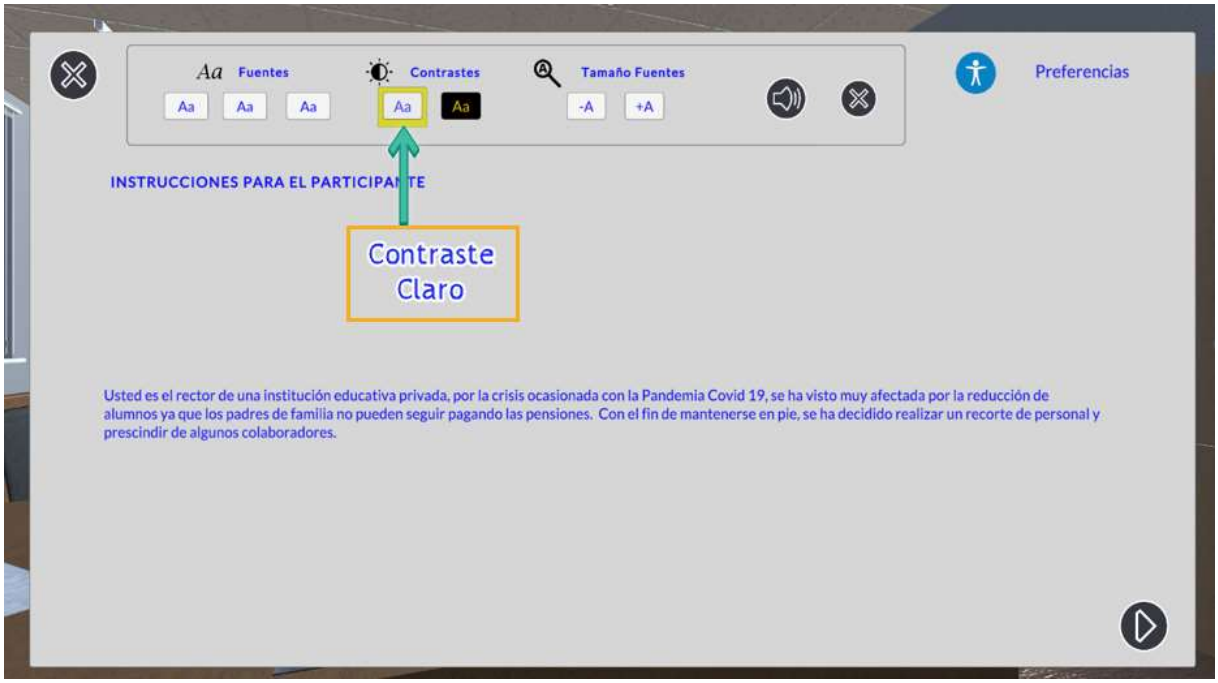


Figura 374. Vista en donde el participante selecciona un contraste claro.

Figura 375. Vista en donde el participante selecciona un contraste obscuro.

Figura 376. Vista en donde participante presiona el botón de reducir el tamaño de letra.

Figura 377. Vista en donde el participante presiona el botón para aumentar el tamaño de letra.

Figura 378. Vista en donde el participante desactiva la descripción de audios.

Figura 379. Vista en donde el participante cierra la barra de preferencias al presionar el botón salir de preferencias.