Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería de Computadores

Trabajo Fin de Grado

Aplicación de ayuda al aprendizaje de lectoescritura a personas con necesidades complejas de comunicación

Autor: Ignacio Rubio Valverde

Tutora: Sira Elena Palazuelos Cagigas

2024

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Computadores

Trabajo Fin de Grado

Aplicación de ayuda al aprendizaje de lectoescritura a personas con necesidades complejas de comunicación

Autor: Ignacio Rubio Valverde

Tutora: Sira Elena Palazuelos Cagigas

Tribunal:

Presidente: Pedro Martín Sánchez

Vocal 1º: Pablo Ramos Sainz

Vocal 2º: Sira Elena Palazuelos Cagigas

Fecha de depósito: 04 de Febrero de 2024

A todo el que un día creyó que no era suficiente...

"Nuestro mayor miedo, no es que no encajemos... Nuestro mayor miedo es que tenemos una fuerza desmesurada, es nuestra luz y no nuestra oscuridad lo que más nos asusta, empequeñecerse no ayuda al mundo, no hay nada inteligente en encogerse para que otros no se sientan inseguros a tu alrededor, todos deberíamos brillar como hacen los niños, no es cosa de unos pocos, sino de todos, y al dejar brillar nuestra propia luz, inconscientemente damos permiso a otros para hacer lo mismo, al liberarnos de nuestro propio miedo, nuestra presencia libera automáticamente a otros..."

Coach Carter (Largometraje, 2005)

Agradecimientos

A mis cuatro abuelos, sé que estáis muy orgullosos de mí por haber llegado hasta aquí, pero esto no hubiera sido posible si vosotros no hubierais puesto la primera piedra de este camino. No habréis podido estudiar nunca, pero me habéis enseñado las lecciones mas importantes de mi vida. Orgullo es lo que siento yo de verdad por ser vuestro nieto.

A mis padres y a mi hermano, gracias por ayudarme a seguir cuando lo más fácil hubiera sido tirar la toalla y rendirse. Como me decís siempre, al final, sobre todo si te esfuerzas, las cosas llegan.

A mi tío Miguel, este trabajo no estará basado en tu discapacidad, pero has sido, eres y siempre serás una gran inspiración para mí. Millones de gracias por estar a mi lado siempre que lo necesito.

A Sira, Marisa y Candela, gracias por todo lo que habéis hecho por mí durante la creación de este trabajo. Sin vuestra ayuda, de valor incalculable, nada de esto hubiera sido posible. En el nombre del autor de este trabajo aparece el mío, pero Athena también es vuestro. Ha sido un verdadero placer trabajar con vosotras a lo largo de todos estos meses.

A Óliver, gracias por ayudarme a dar los primeros pasos en este trabajo y a asentar lo que luego serían las bases de él. Sin tu gran apoyo al principio, el final nunca hubiera llegado.

A Miguel, gracias por todos tus consejos y por toda la ayuda con el diseño de este trabajo. Eres de lo mejor que me ha dado la universidad y nunca te podré estar lo suficientemente agradecido por todo lo que has hecho por mí.

A Lucía, David, Santi, Víctor, Andrés, Pablo, Omar, Andrea, Dani, Inés y los demás. Caminante, no hay camino, se hace camino al andar. Pero mejor hacerlo acompañado de personas que merecen la pena.

Y por último, gracias a mí. Hemos aguantado contra viento y marea, nos hemos sentido superados e insuficientes, hemos tropezado con muchísimas piedras en el camino... ¡Pero lo hemos conseguido!

Resumen

La lectura y escritura son habilidades cruciales para la integración de las personas, y las dificultades en su aprendizaje pueden tener un gran impacto a todos los niveles. En ese contexto, se ha desarrollado Athena, un entorno diseñado para apoyar a niños con TEA en el aprendizaje de la lectoescritura. Incluye Athena Escenas, una aplicación Android que presenta visualmente escenas con audio sincronizado que resalta letras y sonidos para facilitar la comprensión y Athena Creaciones permite a logopedas y familiares personalizar y adaptar las escenas para cada niño, fomentando la curiosidad y la motivación en el aprendizaje.

Palabras clave: Lectoescritura, aplicación Android, escenas visuales.

Abstract

Reading and writing are crucial skills for people's integration, and difficulties in learning them can have a significant impact at all levels. In this context, Athena has been developed, an environment designed to support children with Autism Spectrum Disorder (ASD) in learning literacy. It includes Athena Scenes, an Android application that visually presents scenes with synchronized audio, highlighting letters and sounds to enhance comprehension. Athena Creations allows speech therapists and family members to customize and adapt scenes for each child, fostering curiosity and motivation in learning.

Keywords: Literacy, Android application, visual scenes.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

La literatura es mi Utopía. En ella no tengo limitaciones. No hay barrera de sentidos que me pueda apartar de los dulces discursos de mis amigos los libros. Ellos me hablan sin vergüenza y sin incomodidad.

(Helen Keller, 1880-1968)

La lectura y la escritura son dos habilidades esenciales para poder conocer bien el mundo que nos rodea. Sin embargo, muchas personas tienen dificultades para poder aprender y escribir. Helen Keller fue una de esas personas. Keller se quedó sorda y ciega a una edad muy temprana, pero fue capaz de aprender a leer y a escribir a pesar de sus dificultades. Su historia de superación nos enseña que por muchas dificultades que haya, si se ponen los medios necesarios, cualquier persona puede aprender a leer y a escribir.

Afortunadamente, en la era digital en la que nos encontramos, las personas con necesidades complejas de comunicación (NCC) tienen acceso a la información, al ordenador con adaptaciones de acceso y a las tecnologías móviles como ninguna generación anterior, con un menor coste y con una mayor aceptación social [1]. Gracias a estas facilidades y al avance en la aceptación, los niños con NCC están utilizando este tipo de tecnologías para participar en las aulas de educación ordinaria, para aprender a leer y escribir, para prepararse para una participación activa y significativa en la sociedad adulta, así como para interactuar con otras personas [2].

Dado que dentro de las dificultades específicas del aprendizaje (DEA) hay un amplísimo rango de distintas dificultades, nos hemos centrado en los niños y las niñas con Trastorno del Espectro Autista (TEA). Los TEA son discapacidades del desarrollo causadas por diferencias en el cerebro. Algunas personas con TEA tienen una diferencia conocida, como una afección genética. Los científicos creen que los TEA tienen múltiples causas que, al actuar juntas, cambian las maneras más comunes en las que las personas se desarrollan, se comunican y aprenden [3]. Gracias a estos descubrimientos, nos hacemos a la idea de que las personas con este tipo de trastornos pueden llegar a aprender y hacer todo lo que se propongan, pero, al tener características y rasgos fuera de lo común en su cerebro, hay que enfocar de manera diferente dicho aprendizaje.

Con el objetivo de apoyar desde la ingeniería el aprendizaje de lectoescritura, hemos creado un entorno de sistemas y aplicaciones llamado Athena, el cual facilitará el aprendizaje de los niños y niñas en gran medida. Dicho entorno está compuesto por los siguientes elementos:

- Una aplicación que proporcionará a estas niñas y niños estímulos audiovisuales mediante actividades basadas en elementos de su alrededor, ya sean personas, animales, objetos cotidianos o cualquier cosa que les pueda llamar la atención, para que consigan aprender a leer de una manera adaptada a sus necesidades psicológicas, aumentando la motivación al guiarnos por su curiosidad.

- Una serie de aplicaciones y sistemas que proporcionarán a profesionales y familiares (no podemos olvidarnos de ellos, ya que son el pilar fundamental en el que se apoya el desarrollo y aprendizaje de los niños) una serie de herramientas para poder añadir, modificar y adaptar a cada necesidad los elementos audiovisuales que mostrará el sistema. Este entorno está compuesto principalmente por una base de datos, para almacenar la información importante sobre los usuarios, y las actividades propuestas; almacenamientos tanto locales como en la nube, para guardar los elementos audiovisuales de una manera eficiente; una aplicación móvil, para la configuración de las actividades por parte de los profesionales y otra aplicación móvil, para la realización de las actividades por parte de los niños y niñas.

Capítulo 2

Estudio Teórico

2.1 Estado del Arte

Una vez que se adquiere la capacidad de leer, la lectura tiene un impacto positivo en el éxito escolar, la empleabilidad, la independencia y la autonomía, proporciona un medio para el aprendizaje permanente, el entretenimiento y la introspección, y en el caso de las personas que usan comunicación aumentativa y alternativa (CAA) conlleva una mejor comunicación cara a cara junto la capacidad añadida de participar en la comunicación asíncrona [2] [4].

La integración de la tecnología digital convencional en los dispositivos de CAA de alta tecnología con salida de voz, que incorporan símbolos o fotografías para la representación de los conceptos, y la aparición de "apps" en productos Apple y Android, ha proporcionado mayores oportunidades para que las personas que utilizan CAA se involucren con la tecnología digital [5]. Los niños con NCC están utilizando la CAA para participar en las aulas de educación ordinaria, para aprender a leer y escribir, para prepararse para una participación activa y significativa en la sociedad adulta, así como para interactuar con otras personas [2].

De forma específica, los nuevos diseños de las apps de CAA como son las escenas visuales y las vídeo escenas visuales mejoran la comprensión, la expresión y el aprendizaje del lenguaje de niños con discapacidad severa [6] y con la incorporación de la característica de "programación justo en el momento" permiten incorporar nuevo vocabulario en el momento en que surge la necesidad, aprovechando las interacciones a lo largo del día, como momentos que se pueden enseñar posteriormente en las aplicaciones [7] [8].

Ahora bien, aunque existe un sustancial cuerpo de investigación que demuestra convincentemente los beneficios de la CAA para los niños con NCC [9] el diseño de la tecnología se ha identificado como una de las principales barreras para el aprendizaje de la lectoescritura [10]. Tanto las tecnologías tradicionales de CAA con presentaciones en cuadrículas como los nuevos diseños de escenas visuales o vídeo escenas visuales parecen no apoyar el aprendizaje de la lectura para algunas personas que utilizan CAA [11].

En los diseños tradicionales, los sistemas presentan los símbolos ordenados en cuadriculas, acompañados de la palabra escrita en tamaño pequeño colocada encima o debajo del símbolo de forma estática; la etiqueta de texto también puede aparecer dentro de la barra de mensajes, desplazada del referente cuando se selecciona un símbolo gráfico de la pantalla. Además, se suele utilizar texto apoyado con símbolos para que los niños y las niñas con NCC puedan participar en el aula en actividades de alfabetización como, por ejemplo, el texto apoyado con pictogramas de los programas informáticos News-2-You, PixWriter, o del software Escribir con símbolos. En este tipo de programas que utilizan la salida de voz se emparejan símbolos con texto o el texto sustituye al símbolo para cada palabra de la frase. Aunque estos apoyos a

la CAA (tanto los dispositivos específicos, como el software especializado de texto apoyado con símbolos) están pensados para mejorar las habilidades lingüísticas y de alfabetización, el emparejamiento estático del texto con los símbolos gráficos puede tener un impacto mínimo en el desarrollo de las habilidades de alfabetización [12] [13].

Una posible explicación de la falta de impacto en el reconocimiento de palabras es que los símbolos interfieren en el procesamiento de las palabras escritas y, por tanto, el emparejamiento estático del texto y el símbolo puede no apoyar el aprendizaje del reconocimiento visual de palabras [11] [13] [12] [14] y señalan que cuando el símbolo se empareja de forma estática con el texto, el símbolo se reconoce con poco esfuerzo y, por tanto, la atención se dirige al símbolo gráfico y no se presta atención al texto [15] [16] [17].- El emparejamiento estático de texto y símbolos gráficos es una característica común del diseño de la tecnología de CAA y puede, inintencionadamente, constituir una dificultad para la adquisición del reconocimiento de palabras y, en general, para unos buenos resultados de alfabetización de las personas que necesitan la CAA [11].

2.2 ¿Por qué se ha creado el entorno de aplicaciones?

Debido a que el emparejamiento estático del texto con los símbolos gráficos puede tener un impacto mínimo en el desarrollo de las habilidades de alfabetización, profesionales que trabajan con niños y niñas con TEA se pusieron en contacto con nosotros para solicitar el diseño e implementación de una aplicación que pudiera ayudarles en la enseñanza de la lectura y, de esta manera, facilitar a los profesionales la configuración de los elementos necesarios. Los requisitos de la aplicación serían los siguientes:

- Debe incluir una aplicación de ayuda al aprendizaje de la lectoescritura, que permita:
  - Visualizar una imagen.
  - Mostrar la palabra asociada (cuyo aspecto se modificará cambiando de tamaño y de color el tipo de letra para llamar la atención sobre cada sílaba).
  - Reproducir la palabra (grabada o generada con un sintetizador de voz) de forma síncrona con las sílabas resaltadas.
- Una interfaz lo más intuitiva posible y sencilla para que se pueda utilizar fácilmente desde el primer momento.
- Un fuerte contraste entre los elementos de texto dinámicos y los estáticos, ya que los elementos dinámicos (palabras que se “iluminan”, con un color muy llamativo) son los que están sincronizados con el audio que se reproducirá al mismo tiempo y tienen que llamar la atención.
- El acceso por parte de los niños y niñas debe estar restringido a las actividades o escenas visuales que tienen que ver.
- Debe incluir una aplicación que permita a los profesionales seleccionar el material audiovisual. Podrán utilizar tanto fotos como audios hechos o adaptados por ellos para que los niños y niñas estén lo más familiarizados posible con estos elementos, lo que les motivará en el proceso de aprendizaje,
- Debe permitir que la sincronización entre audio y los elementos dinámicos de texto sea fácilmente configurable por los profesionales.

A partir de estos requisitos, llegamos a la conclusión de que la mejor opción para cumplir estos puntos sería separar lo más posible la aplicación que los niños van a utilizar de las herramientas de los

profesionales, por lo cual, una única aplicación sería inviable. Para satisfacer las condiciones anteriores, la mejor opción es la creación de un entorno con los siguientes elementos:

- Una aplicación para los niños, que una vez iniciada sesión por parte de un profesional en la aplicación pueda elegir el niño o niña que la utilizará en ese momento y que solo aparezcan las escenas asociadas a él o ella, restringiendo así su acceso a más allá de lo que estos necesitan o pueden controlar.
- Una aplicación para los profesionales, en la cual se les proporcionarán distintas herramientas para que estos puedan previsualizar, editar, gestionar y adaptar dichas escenas en el mismo medio (las *tablets*) en el que las verán los niños y las niñas, una vez les sean asignadas.
- Un mecanismo de gestión y almacenamiento de todos los datos y la información necesaria para el correcto funcionamiento de la aplicación, tanto sobre las escenas como sobre los usuarios.

Una vez estudiadas todas las opciones de entornos de programación que nos permitan crear dichas aplicaciones y mecanismos, hemos decidido utilizar los siguientes:

- Kotlin: Es el lenguaje por excelencia en la actualidad para crear aplicaciones que serán ejecutadas en entornos Android. Kotlin es un lenguaje de programación expresivo y conciso que permite reducir los errores de código comunes y se integra fácilmente en las aplicaciones existentes, lo que nos da versatilidad a la hora de adaptar las aplicaciones a las necesidades de los niños y las niñas qué las utilizarán.
- Node.js: Al ser un entorno de ejecución de JavaScript en el lado del servidor, Node.js nos permite crear una API multiplataforma (no importa el sistema operativo en el que se ejecute) y dinámica (podemos modificarla en tiempo real), además de que tiene una gran variedad de paquetes y bibliotecas que facilitan el trabajo.
- MySQL: Las principales razones para usar este sistema de gestión de bases de datos son que es uno de los gestores más rápidos que existen en la actualidad y es muy escalable, lo que nos permite ampliar la base de datos en función del volumen de datos que se manejen.
- Firebase: Ese servicio de hosting de archivos multimedia de Google, gratuito en niveles básicos. Es muy adaptable y permite su integración en los distintos entornos de programación en los que hemos desarrollado las distintas aplicaciones.
- Figma: Herramienta de diseño de interfaces de usuario (UI) que permite crear prototipos interactivos de aplicaciones móviles. Es una herramienta versátil y fácil de usar que ofrece una serie de ventajas para el diseño de aplicaciones, como colaborar en tiempo real con varias personas y tener feedback del diseño, aparte de los prototipos interactivos anteriormente mencionados que permiten revisar con más precisión si los diseños están acordes a las necesidades de los niños.
- Docker: Al no poseer un servidor donde poder almacenar la base de datos relacional que necesita el entorno, esta herramienta nos permite ejecutar el gestor de la base de datos en cualquier máquina que tenga Docker instalado y poder tener más opciones disponibles en caso de no poder usar la máquina principal.

Capítulo 3

Athena

3.1 Introducción al entorno

Athena es un entorno formado por múltiples aplicaciones diseñado para ayudar a niños y niñas con TEA a adquirir habilidades de lectoescritura a través de la reproducción de escenas visuales.

Como podemos ver en la figura 3.1, una escena visual se compone de una imagen, un texto y un audio.

Figura 3.1: Ejemplo de escena visual mostrada en la aplicación

Al pulsar la imagen, comienza la reproducción de la escena. Esta reproducción coordina el audio con la dinámica del texto, que se compone de: reproducción inicial de palabra completa, pausa 1, reproducción secuencial de los fonemas que forman la palabra resaltando cada letra simultáneamente, pausa 2, reproducción de la palabra sílaba por sílaba, resaltando cada sílaba en paralelo con su sonido, pausa 3 y reproducción final de la palabra completa. En la figura 3.2, se puede observar una representación visual de ese ciclo.

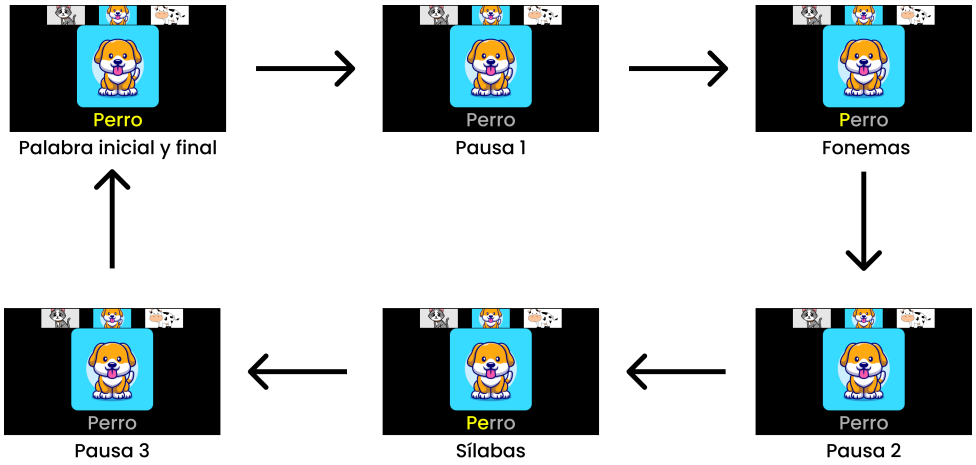

Figura 3.2: Ejemplo del ciclo de reproducción de una escena

Una vez que sabemos lo que es una escena, pasamos a explicar el porqué, partiendo de estas escenas, se ha construido todo un entorno.

Lo primero de todo, hay que puntualizar que el objetivo fundamental de estas escenas son niños con TEA, es decir, niños que necesitan que el acceso a todos los elementos sea lo más intuitivo y sencillo posible. Además, están muy familiarizados con el uso de *tablets*. De estas premisas, nace la primera aplicación del entorno: Athena Escenas.

3.1.1 Athena Escenas

Figura 3.3: Logotipo de la aplicación Athena Escenas

Athena Escenas es una aplicación orientada a que los niños puedan reproducir las escenas visuales de una manera sencilla en *tablets* con sistema operativo Android. En primer lugar, el logopeda o el familiar que esté trabajando con el niño debe iniciar la sesión con las credenciales que se le han asignado. Una vez dentro, selecciona al niño (llamado "alumno" en el entorno), como se ve en la figura 3.4, y ya le puede dar el dispositivo para que lo utilice de manera autónoma, sin necesidad de ayuda.

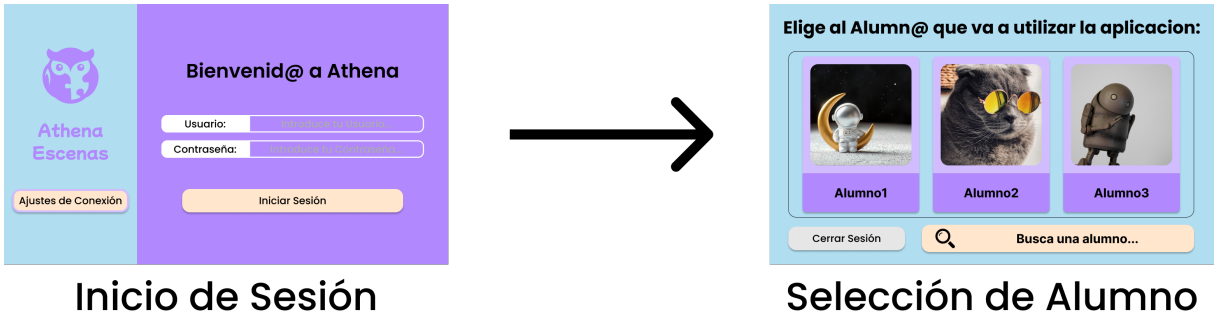

Figura 3.4: Flujo de inicio de sesión de Athena Escenas

Para que el alumno no tenga dificultades a la hora de reproducir las escenas, se ha creado una galería que muestra de una manera muy intuitiva y con colores que no destaquen, aquellas escenas que tiene asignadas. Al pulsar en una de ellas, se visualiza la pantalla de reproducción de la escena (figura 3.1). A continuación, en la figura 3.5, podemos observar una captura de la pantalla de galería.

Figura 3.5: Galería de escenas de la aplicación Athena Escenas

Para que las escenas visuales que se ofrecen a cada niño estén personalizadas a sus necesidades de aprendizaje y preferencias, debemos proporcionar una serie de herramientas a los logopedas para que estos puedan adaptar estas escenas al entorno del niño utilizando elementos que ya han visto o conocen. Para ello, se ha creado la otra aplicación del entorno: Athena Creaciones.

3.1.2 Athena Creaciones

Figura 3.6: Logotipo de la aplicación Athena Creaciones

Athena Creaciones es una aplicación que proporciona herramientas de creación, edición y gestión de las escenas visuales que verán los alumnos en su aplicación, así como de los datos de usuarios y alumnos que la utilizan.

Para ello, como podemos ver en la figura 3.7, se ha diseñado una serie de menús que permiten acceder a estas herramientas, según el objetivo final de estas: usuarios, escenas o alumnos.

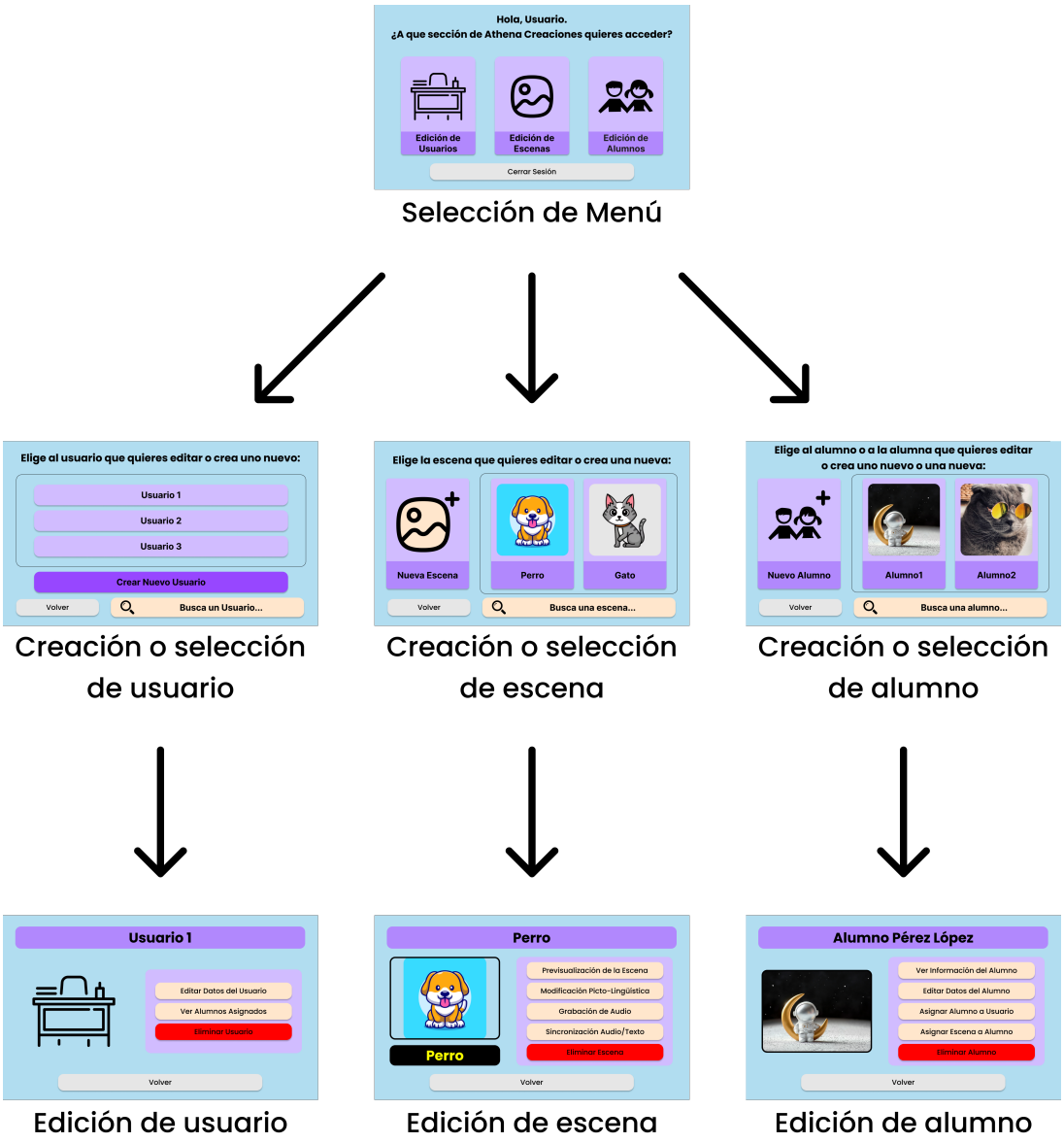

Figura 3.7: Flujo de menús de la aplicación Athena Creaciones

- Por parte de los usuarios, las herramientas disponibles (figura 3.8) son:
  - Creación de Usuario
  - Edición de Usuario
  - Ver los Alumnos que el Usuario tiene asignados
  - Eliminación de Usuario

Figura 3.8: Herramientas de gestión de los usuarios

- Por parte de las escenas, las herramientas disponibles (figura 3.9) son:

- Creación de Escena
- Visualización de Escena
- Ajuste de Tipografía
- Ajustes Picto-lingüísticos
- Grabación de Audio
- Sincronización
- Eliminación de Escena

Figura 3.9: Herramientas de gestión de las escenas

- Por ultimo, las herramientas disponibles para la gestión de los alumnos (figura 3.10) son:
  - Creación de Alumno
  - Visualización de Datos de Alumno
  - Edición de Alumno
  - Asignación de Alumno a Usuarios
  - Asignación de Escenas a Alumno
  - Eliminación de Alumno

Figura 3.10: Herramientas de gestión de los alumnos

3.2 Arquitectura del entorno

Una vez que hemos presentado el entorno desde un punto de vista de funcionalidad, pasamos a la descripción de este desde un punto de vista estructural.

Aunque lo principal que se muestra al usuario final del entorno son las aplicaciones de las *tablets*, cuyas pantallas constituyen la mayor parte de la estructura visual, Athena esta sustentado en otras dos estructuras más, una de datos y una lógica, siendo la coordinación entre estas estructuras un aspecto crucial, ya que tiene que ser perfecta para que la funcionalidad del entorno sea correcta.

Dada esa división de estructuras y según lo descrito por la empresa IBM [18], describiremos la estructura de Athena en tres niveles: nivel de presentación, nivel de datos y nivel lógico.

3.2.1 Nivel de presentación

El nivel de presentación de un sistema engloba todos los elementos con los que el usuario final interactúa. Su principal propósito es mostrar al usuario información, además de recabar datos de este que sean relevantes para el funcionamiento del sistema. [18]

En Athena, como se ha indicado anteriormente, prácticamente la totalidad del nivel de presentación recae sobre las pantallas de las aplicaciones para *tablets*. Estas pantallas están diseñadas a través del uso de archivos XML. En estos archivos se definen y almacenan los recursos visuales de las aplicaciones, como, por ejemplo, iconos, imágenes y cadenas de texto, para, de esta manera, facilitar su modificación y reutilización.[19]

3.2.2 Nivel de datos

El nivel de datos de un sistema, también llamado nivel de base de datos o de *back-end*, engloba los sistemas que almacenan y gestionan todos la información que el sistema maneja. [18]

En este entorno, el nivel de datos se divide en dos partes: la base de datos y los elementos audiovisuales.

- La base de datos de Athena tiene como software gestor (SGBD) el software MySQL [20]. Este software gestiona bases de datos relacionales, haciendo que todos los datos del entorno estén organizados en torno a estructuras de tablas, vistas, filas y columnas relacionadas entre sí. Esto permite que operar con los datos se haga de una manera flexible y sencilla.

  Una vez que tenemos el SGBD, necesitamos un lugar donde almacenar tanto la base de datos, como su gestor. En este caso, se ha optado por configurar un contenedor, gestionado por la aplicación Docker [21] y en un equipo que estará conectado en red local con las *tablets*, de manera que se conforma un servidor local.

  Docker es un programa que permite crear, ejecutar y administrar aplicaciones dentro de entornos aislados llamados contenedores. Los contenedores son unidades de software que empaquetan código, dependencias y configuraciones en un solo lugar, lo que hace que sean portátiles, fáciles de implementar [22] y no necesiten grandes recursos. Por tanto, la principal ventaja que aportan estos contenedores al entorno, es que no se necesita siempre el mismo equipo para alcanzar la información, solo hay que estar en la misma red que las *tablets* y no se necesita invertir demasiado dinero en un equipo puntero y costoso.

- Los elementos audiovisuales son los pilares básicos sobre los que se sustentan las escenas visuales. En comparación con la base de datos, estos elementos suelen tener un tamaño de archivo mucho más grande, por lo que el almacenamiento local no es viable.

  En este caso, se ha optado por configurar un *cloud hosting* gratuito en la plataforma Firebase [23]. Firebase es la plataforma de desarrollo de aplicaciones móviles de Google, y al ser de la misma empresa que es propietaria de Android, el sistema operativo sobre el que se ejecutan las aplicaciones principales del entorno, la integración de este *hosting* es muy sencilla. Además de esto, como las *tablets* están conectadas a internet durante casi la totalidad del tiempo, no se necesitaría desarrollar más sistemas para obtener los elementos de las escenas.

3.2.3 Nivel de aplicación

El nivel de aplicación de un sistema, también conocido como nivel lógico o nivel de lógica de negocio, es el núcleo de la aplicación. En este nivel, la información recogida en el nivel de presentación y la información almacenada en el nivel de datos es procesada usando un conjunto específico de reglas propias de cada sistema [18].

En Athena, este nivel esta dividido en dos subniveles: el subnivel lógico y el subnivel de comunicación.

- El subnivel lógico del entorno se corresponde con todo el código escrito en lenguaje Kotlin, sobre el que se han desarrollado las aplicaciones que se usan tanto niños como logopedas en las *tablets*. Este lenguaje, del que Google es copropietario, esta totalmente integrado en el ecosistema Android y facilita el desarrollo de aplicaciones en este sistema operativo, además de ser un lenguaje seguro por diseño y conciso[24].

- Por la otra parte, el subnivel de comunicación del entorno esta compuesto en su totalidad por la API, desarrollada en lenguaje Node.js[25]. Esta API, que sería la metáfora del sistema nervioso del entorno, nos permite comunicar las distintas áreas de este de una manera clara y ordenada, permitiendo que la información y los recursos lleguen correctamente a todos los lugares donde se requieran.

3.3 Implementación del entorno

Finalmente, después de haber presentado el entorno y haber descrito su estructura, pasamos a especificar su implementación, es decir, como se ha construido el entorno.

Para poder explicar la implementación del entorno de una manera ordenada, como podemos ver en la figura 3.11, dividiremos a Athena en tres bloques según el lugar donde se ejecutan las aplicaciones, resultando así tres grupos de estas: las aplicaciones que se ejecutan en las tablets, las aplicaciones que se ejecutan en el servidor local y las aplicaciones que se ejecutan en la nube.

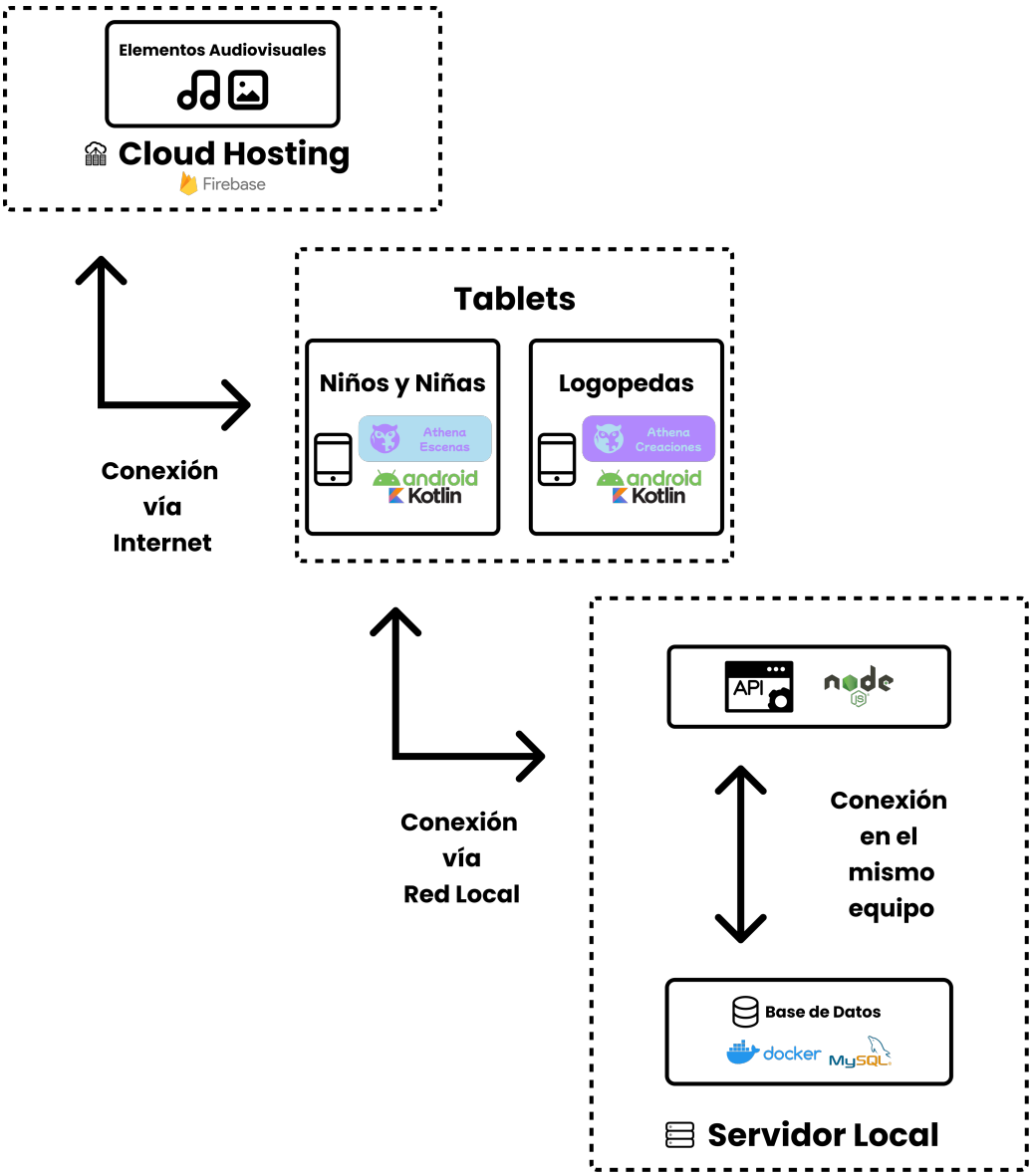

Figura 3.11: Esquema del entorno Athena, dividido según el lugar en el que se ejecuta cada aplicación

3.3.1 Aplicaciones ejecutadas en las *tablets*

Dado que los niños y niñas con TEA están muy familiarizados con el uso de *tablets*, la importancia que tienen estas aplicaciones en el entorno es uno de los principales aspectos que se tuvo en cuenta a la hora de su diseño e implementación.

Ambas aplicaciones están creadas en lenguaje Kotlin, ya que al ser el lenguaje prioritario para el desarrollo de aplicaciones que se ejecutan en el sistema operativo Android, aporta mucha simplicidad gracias a que es un lenguaje muy conciso, y tiene simultaneidad estructurada gracias a las corrutinas que veremos más adelante. Estas dos características son muy importantes a la hora de reproducir las escenas visuales[24].

Pero este lenguaje no sería útil para estas aplicaciones sin un diseño gráfico acorde a las necesidades a los niños con TEA. Para fomentar su concentración y que su aprendizaje se optimice en la medida de lo posible, los diseños de las pantallas de ambas aplicaciones contienen elementos lo más redondeados posible y todos los colores que las conforman son de una gama fría, los cuales no llaman la atención. El objetivo es que los niños se centren en las escenas una vez lleguen a ellas.

Gracias a la aplicación Figma [26], se han creado prototipos funcionales sin código de ambas aplicaciones, que permitían recibir *feedback* visual del resultado final de las pantallas de las aplicaciones, con las dimensiones de la pantalla de referencia del dispositivo real, antes de programarlas en código. Esto ha hecho que la implementación de Athena Escenas y Athena Creaciones fuera mucho más rápida, y en el caso de tener que realizar alguna modificación sobre la marcha, se simplificaran.

Como punto de partida de la implementación de ambas aplicaciones, Athena Escenas y Athena Creaciones, hay que indicar la estructura de esta se compone de dos partes estructurales los archivos de recursos y los archivos de lógica de negocio.

- **Archivos de recursos**

  Estos archivos, escritos en lenguaje XML, encontramos todo el contenido propio que se utiliza en la aplicación, como imágenes, sonidos, animaciones, cadenas de texto, estilos y otros elementos. Al separar el código de la aplicación del contenido estático, se facilita el mantenimiento de la aplicación, ya que el contenido estático puede ser actualizado sin tener que modificar el código [27].

  Los archivos de recursos se organizan en diferentes carpetas, cada una de las cuales contiene un tipo de recurso específico. Las carpetas principales son las siguientes:

  - drawable: Contiene las imágenes, los vectores y otros recursos gráficos propios de la interfaz de cada aplicación.
  - layout: Contiene archivos XML que definen como se verá la estructura de la interfaz de usuario de cada pantalla, o de secciones de estas, en las aplicaciones.
  - values: Contiene cadenas de texto, valores numéricos, especificaciones de estilos y otros recursos que no son gráficos, que modelan estos.

- **Archivos de lógica de negocio**

  En los archivos de lógica de negocio de las aplicaciones Android, escritos en lenguaje Kotlin, encontramos el conjunto de reglas y operaciones que definen cómo esta debe funcionar. Esta lógica es independiente de la interfaz gráfica de la aplicación, es decir, ella misma constituye una unidad que no necesita de la interfaz para ser funcional. Pero tanto en Athena Creaciones, como en Athena Escenas, es imprescindible que toda la lógica este perfectamente coordinada con las interfaces

gráficas. Debido a esto, dividimos a estos archivos en dos grupos: las clases de la lógica de negocio [28] y las actividades [29].

Las clases son los bloques de construcción básicos de cualquier aplicación Android. Son unidades de código que definen objetos. Los objetos son instancias de clases que pueden almacenar datos y realizar acciones. En las aplicaciones Android de Athena, estas clases se dividen en paquetes según su funcionalidad:

- Adaptadores (paquete "adapters"): Son las clases que tienen como utilidad definir como se van a organizar las galerías de elementos visuales propios, que se han creado específicamente para las aplicaciones.
- Solicitudes API (paquete "apirequests"): Son las clases que tienen como utilidad almacenar la información que se necesita en cada llamada a la API y generar la representación de esta información en objetos JSON [30] que serán enviados a dicha API.
- Respuestas API (paquete "apiresponses"): Son las clases que tienen como utilidad almacenar la información que se aporta en cada respuesta que nos llega de la API. Son el objeto inverso a las solicitudes, ya que almacenan la información que nos llega en los objetos JSON enviados por la API.
- Clases (paquete "clases"): Es el paquete que agrupa distintas clases que agrupan información necesaria para el funcionamiento de las aplicaciones. Un ejemplo de una de estas clases, es la clase "InformacionSesion", cuya instanciación en un objeto hace que se recoja, en todo momento, que una aplicación Android esta ejecutándose, qué usuario esta usando la *app*, qué alumno, y qué escena se esta visualizando o editando [31].
- Interfaces (paquete "interfaces"): Es el paquete que agrupa las interfaces no gráficas de las aplicaciones. Una interfaz no gráfica en Kotlin contiene declaraciones de métodos específicos, para que luego los usen las clases que las implementan [32].
- View Holders (paquete "viewholders"): Son las clases que tienen como utilidad definir la estructura visual y el comportamiento de los objetos que se muestran en las galerías de elementos propios de cada aplicación.

En Kotlin, las actividades son componentes de la interfaz de usuario que representan una pantalla o vista en una aplicación Android y definen su funcionamiento. Es decir, estas proporcionan el nexo de unión entre el nivel de presentación de las aplicaciones del entorno y el nivel de lógica de negocio. Debido a que las interfaces gráficas del entorno deben ser lo más homogéneas posibles, independientemente del modelo de *tablet* utilizado, todas las actividades del entorno implementan los métodos de escalado definidos en la interfaz no gráfica "ClaseGrafica", ya que el escalado automático nativo de Android mueve los elementos según su tamaño. El funcionamiento de todas las actividades de las aplicaciones se compone de los siguientes pasos:

1. **Escalado de elementos gráficos.**

   El escalado, como se ha mencionado anteriormente, viene definido según los métodos declarados en la interfaz "ClaseGrafica" ya que el escalado nativo de Android no es la mejor opción a implementar en nuestro entorno.

   Estos métodos, toman los datos de la pantalla de la *tablet* que esté ejecutando la aplicación y obtienen una relación de proporción con las métricas definidas en los prototipos visuales creados para facilitar la implementación de las aplicaciones. Estos prototipos están basados en la pantalla de la *tablet* Xiaomi Redmi Pad [33] que tiene unas medidas de 1142dp x 669dp (figura 3.12).

Una vez que se obtienen las proporciones, escalan uno a uno los elementos que contiene la interfaz gráfica de cada una de las pantallas, asegurándonos de esta manera de que los elementos siempre se van a encontrar en las mismas coordenadas, sea cual sea el modelo de *tablet* utilizado.

Figura 3.12: Visualización de las proporciones del modelo en una de las actividades de Athena Creaciones

2. **Asignación e inicialización de elementos de la actividad.**

   Teniendo ya escalados todos los elementos gráficos de la actividad, pasamos a asignarlos a variables definidas en la actividad, y a inicializarlos, para poder hacer uso de ellos en la parte lógica de esta. Aparte de los elementos gráficos, también se inicializan aquellos objetos no gráficos necesarios para el correcto funcionamiento de la actividad.

3. **Obtención de información de la base de datos**

   La gran mayoría de las actividades de las aplicaciones trabajan con datos que están almacenados en la base de datos del servidor local. Para poder trabajar con ellos en ellas, se necesita obtenerlos a través de una solicitud a la API.

   Estas solicitudes, administradas a través de la librería `Retrofit` que se implementa en el lenguaje Kotlin [34], son asíncronas a la ejecución de la actividad. Debido a esto, estas solicitudes tienen que ser coordinadas de forma muy precisa con esta ejecución. Para ello, todos los métodos que obtienen información de la API son ejecutados en unos conjuntos de sentencias, llamados "corrutinas" [35], independientes del hilo principal de la actividad. Las corrutinas se encargan de realizar las llamadas a la API y una vez que se han obtenido los datos necesarios, almacenarlos en el objeto inicializado en la actividad para su posterior uso en ella.

   En las aplicaciones Android del entorno, la información y la coordinación de la API se encuentra en el objeto de la clase `Api`, declarado en todas las actividades de ambas aplicaciones. Los objetos de esta clase, para poder realizar las peticiones a la ruta correcta de la API, implementan los métodos definidos en la interfaz no gráfica `ApiService`, ya que estos métodos llevan definidas dichas rutas en su implementación.

   En la figura 3.13, podemos observar el código de uno de estos métodos de obtención de datos, concretamente, la obtención de los datos de un alumno.

```
private fun obtenerAlumno (){
    CoroutineScope(Dispatchers.IO).launch { this: CoroutineScope
        val llamada = api.getApi().getAlumno(AlumnoRequest(infoSesion.getIdAlumno()))
        val respuesta: Alumno? = llamada.body()
        runOnUiThread {
            if (llamada.isSuccessful) {
                if (respuesta != null) {
                    if (respuesta.foto != "error") {
                        alumno = respuesta
                        cargarAlumno(alumno)
                    } else {
                        mostrarMensaje(
                            contexto: this@EdicionAlumnoActivity,
                            titulo: "Error",
                            mensaje: "Ha habido un error al cargar la información del alumno." +
                                    "\nIntentalo de nuevo o si persiste el error, " +
                                    "comprueba la conexión con el servidor."
                        )
                    }
                }
            }
        }
    }
}
```

Figura 3.13: Fragmento de código donde se obtienen los datos de un alumno desde la API

4. **Carga de la información**

   Teniendo ya toda la información necesaria para el funcionamiento correcto de la actividad, le proporciona a los elementos gráficos de las actividades. En este paso, si la información a mostrar es una cadena de texto, se asigna directamente a la variable de texto del elemento gráfico que lo va a contener. Pero en el caso de que la información a mostrar sea de tipo audiovisual, hay que descargarlo del hosting en la nube. Si esta información audiovisual es una imagen, hay que cargarla en el elemento gráfico con la biblioteca "Picasso", si es de reproducción de audio, hay que cargarlo en un objeto de tipo `MediaPlayer` [36] y si es el caso de información de grabación, en un objeto de tipo `MediaRecorder` [37]. Un ejemplo de esta funcionalidad se puede observar a continuación, en la figura 3.14.

```
private fun cargarAlumno(alumno: Alumno) {
    nombreEdicionAlumno.text = alumno.nombre + " " + alumno.apellidos

    val imageRef = firebase.child( pathString: "Users/${alumno.foto}")

    imageRef.downloadUrl

        .addOnSuccessListener { uri ->
            Picasso.get().load(uri).into(imagenEdicionAlumno)
        }
        .addOnFailureListener { it: Exception
            Picasso.get().load(R.drawable.fondo_btneliminar)
                .into(imagenEdicionAlumno)
        }
}
```

Figura 3.14: Fragmento de código donde se cargan los datos obtenidos de un alumno

5. **Funcionalidad asociada a la interacción con elementos gráficos**

   Por ultimo, en la implementación de cada actividad, se ejecuta la funcionalidad asociada a la interacción con los elementos gráficos. En Kotlin, estas interacciones se recogen por medio de eventos de entradas, denominados "listeners" [38].

   Estos "listeners", aunque se asocian a las variables gráficas de la actividad del paso 2, dependen de que el usuario interactúe con el elemento gráfico indicado en la variable. Un ejemplo de una posible interacción con la interfaz gráfica de la actividad sería la pulsación de un botón de "Volver", como se puede ver en ejemplo de la figura 3.15, cuya implementación hace que vuelvas a la pantalla anterior de la aplicación.

Figura 3.15: Ejemplo de "listener" en Athena Creaciones

Una vez descrita la funcionalidad e implementación común en ambas aplicaciones, pasamos a describir los aspectos únicos de cada una.

3.3.1.1 Implementación de Athena Creaciones

Como ya se indicó en la introducción de la aplicación (sección 3.1.2), Athena Creaciones proporciona herramientas para la creación, edición y gestión de los elementos que componen la funcionalidad de las aplicaciones Android del entorno.

Para lograr esta funcionalidad, la aplicación se sustenta en una estructura de actividades (figura 3.16), escritas en lenguaje Kotlin e interconectadas entre sí, que permiten realizar la funcionalidad deseada en la gestión de los elementos, así como la navegación entre las distintas pantallas.

A continuación, describiremos el funcionamiento de cada una de las actividades y destacaremos los aspectos únicos de cada una.

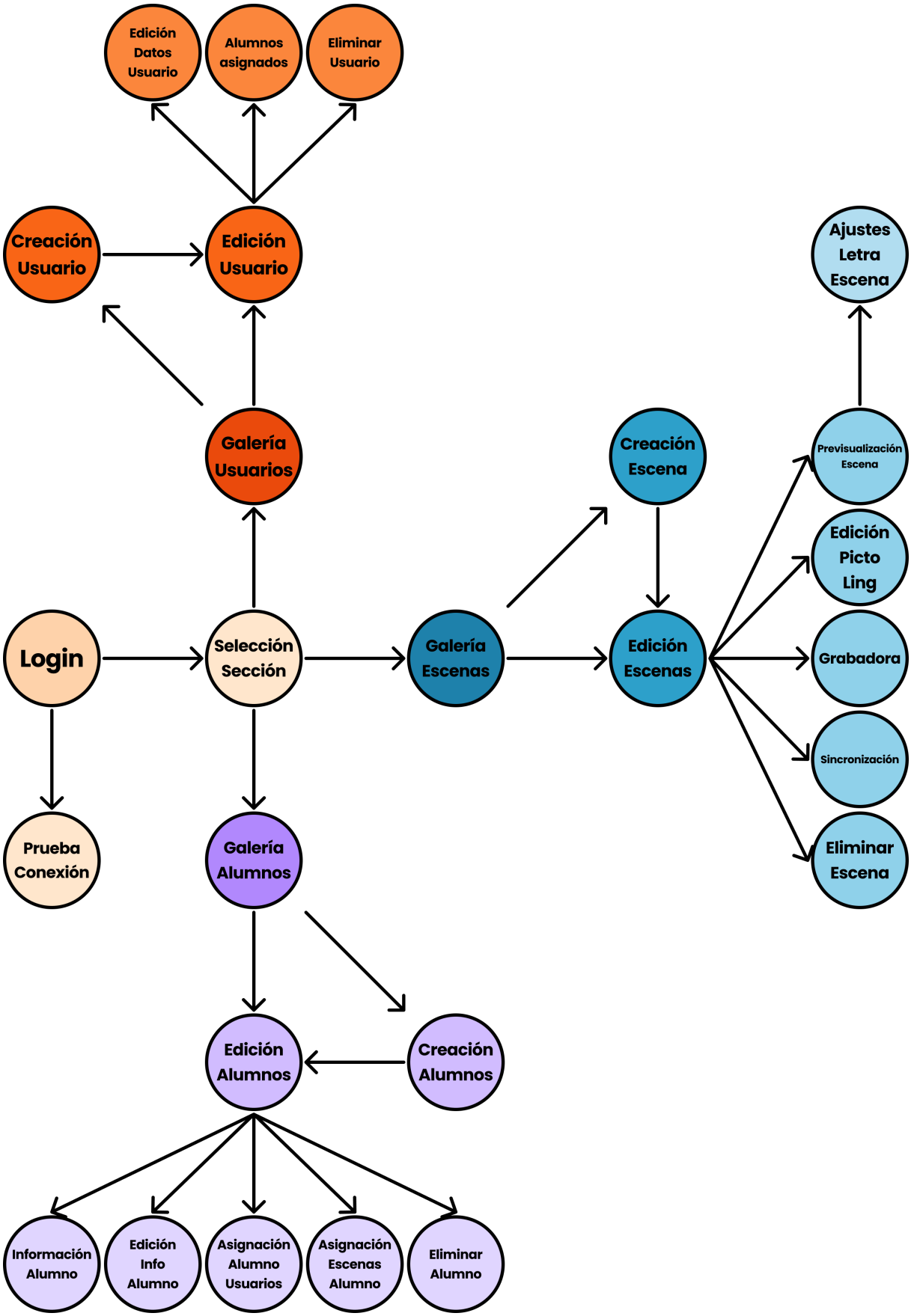

Figura 3.16: Esquema de actividades de Athena Creaciones

- **Login**

  La actividad Login, cuya pantalla podemos ver a continuación en la figura 3.17, nos permite comprobar que el usuario que va a entrar a la aplicación esta registrado en el entorno, mediante el uso de unas credenciales (identificador de usuario y contraseña).

Figura 3.17: Vista de la pantalla de la actividad Login en Athena Creaciones

Como método reseñable, hay que destacar la función `iniciarSesion()`. Este método recoge la información introducida por el usuario en los apartados de introducción de credenciales y envía una solicitud a la API para comprobar si son correctas. En caso afirmativo, la actividad le redirigirá a la actividad Selección Sección.

- **Prueba Conexión**

  La actividad Prueba Conexión, cuya pantalla podemos ver a continuación en la figura 3.18, permite modificar la dirección IP donde está el servidor local, además de comprobar, mediante el envío de una cadena de texto a esa dirección, si la conexión con el servidor local es correcta y la API esta bien configurada.

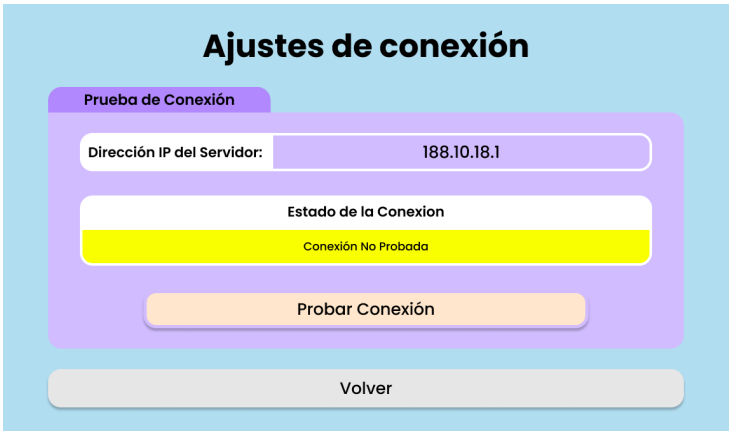

Figura 3.18: Vista de la pantalla de la actividad Prueba Conexión en Athena Creaciones

El principal método de esta actividad es el método `probarConexion`, que realiza la llamada a la API, mostrando mientras un mensaje informativo. Una vez que la llamada y, por tanto, la prueba de conexión, es correcta, muestra información visual en color verde; mientras que en el caso que sea incorrecta, la información visual se muestra en rojo.

- **Selección Sección**

  La actividad Selección Sección es la más sencilla de toda la aplicación, ya que su función es únicamente de nexo entre las diferentes secciones (usuarios, escenas y alumnos). Como podemos ver en la figura 3.19, únicamente alberga cuatro botones: 3 para ir a las distintas secciones y cerrar la sesión y volver a la actividad Login.

Figura 3.19: Vista de la pantalla de la actividad Selección Sección en Athena Creaciones

- **Sección Usuarios**

  La sección de usuarios engloba todas aquellas actividades que gestionan los datos de los usuarios de la aplicación:

  – Galería Usuarios

    Galería Usuarios, cuya pantalla podemos ver en la figura 3.20, permite elegir si se crea un nuevo usuario o se busca y selecciona uno ya existente para usar las herramientas de gestión de datos sobre él.

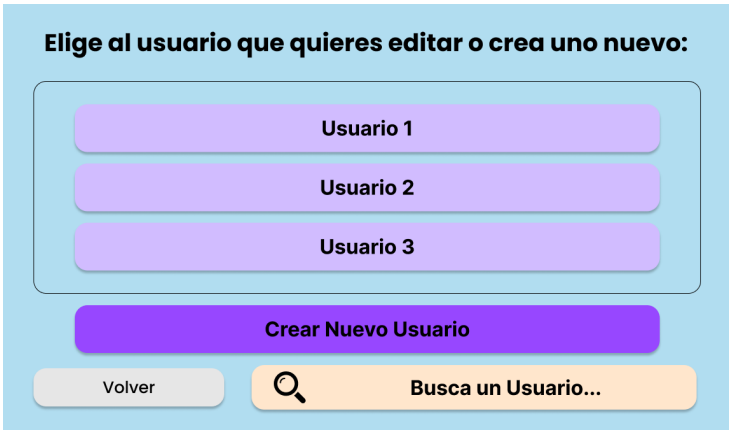

Figura 3.20: Vista de la pantalla de la actividad Galería Usuarios en Athena Creaciones

Su objetivo es controlar y coordinar la visualización de los elementos gráficos que se van a mostrar en la galería, así como controlar los barridos que permiten ver todos los elementos, aunque al inicio no estén en pantalla.

En esta actividad no hay un método principal, si no que hay una funcionalidad principal, mostrar la galería de usuarios, que es una lista dinámica de elementos. Para poder lograr este objetivo, hay que seguir una serie de pasos[39]:

1. Crear el contenedor de la galería.
   La galería esta contenida en un elemento gráfico denominado *Recycler View*, que se declara en el archivo XML asociado a la actividad y se asocia a una variable de la lógica de esta.

2. Crear el adaptador.
   Una vez que tenemos el contenedor, hay que preparar la lógica que se encargará de mostrar los datos en el *Recycler View*. Esa lógica es una clase adaptadora cuyos métodos principales son `onCreateViewHolder()`, para crear una nueva vista para cada elemento del *Recycler View*; `onBindViewHolder()`, para rellenar la vista con los datos del elemento correspondiente; y por último, `getItemCount()`, que devuelve el número de elementos de la lista de llenado del *Recycler View*, para poder contabilizar el número de objetos que componen la galería y facilitar su creación.

3. Asignar el adaptador al contenedor.
   Una vez creado el adaptador, debemos asignarlo al *Recycler View* con su método de inicialización.

4. Crear el *View Holder*.
   Después de haber creado el contenedor y el adaptador, y haberlos asociado, debemos personalizar los elementos que se van a mostrar en la galería, según la información obtenida en la llamada a la API. Para ello, se utiliza la clase *View Holder*, cuyos objetos sirven como recipientes donde se almacenan los datos obtenidos, se personalizan según estos datos y se les da una funcionalidad acorde a lo almacenado.

5. Llenado de la galería con los elementos deseados.
   Para terminar la creación de la galería, solo queda pasarle al adaptador una lista de elementos que contengan la información necesaria para personalizar los *View Holder*. Para ello, se ha configurado un elemento de búsqueda llamado *Search View*, que modela la llamada a la API según la búsqueda que hagamos en él, para que, de esta manera, solo se muestren los elementos deseados en la galería.

– Creación Usuario

La actividad Creación Usuario, como su propio nombre indica, sirve para crear un nuevo usuario en el entorno, a partir de un nombre, un nombre de usuario, una contraseña y un pequeño texto de información adicional.

En su implementación se comprueba que los elementos de introducción de texto con los datos obligatorios (nombre, usuario y contraseña, como podemos ver en la parte derecha de la figura 3.21) no están en blanco o vacíos al pulsar el botón de creación. Acto seguido, se procede a hacer una llamada a la API para completar dicha creación. Si la llamada a la API no ha resultado errónea, la actividad redirige a la actividad de Edición de Usuario.

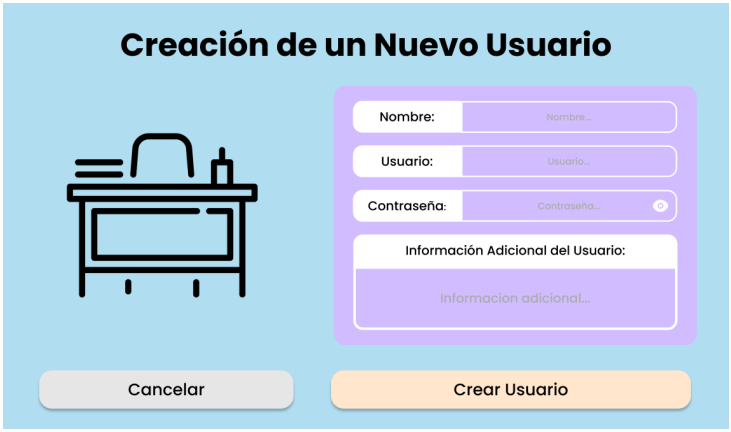

Figura 3.21: Pantalla de la actividad Galería Usuarios en Athena Creaciones.
A la derecha de la pantalla podemos encontrar los elementos de introducción de texto

– Edición Usuario

La actividad Edición Usuario sirve como nexo entre la galería de usuarios y las herramientas para editar el usuario seleccionado (figura 3.22). Muestra en el título el nombre del usuario, que se adquiere al obtener de datos de la actividad (punto 3 de la funcionalidad común), y permite al usuario editar sus datos, ver los alumnos asignados y volver a la galería de usuarios.

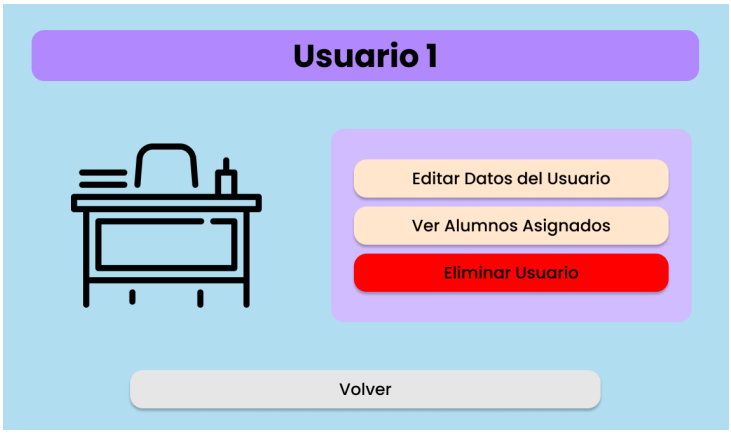

Figura 3.22: Vista de la pantalla de la actividad Edición Usuario en Athena Creaciones

– Edición Datos Usuario

La actividad Edición Datos Usuario le permite modificar sus datos en el entorno. Su funcionalidad y visualización son prácticamente idénticas a Creación Usuario, difiriendo en que hay que cargar los datos del usuario en la actividad al iniciarla.

– Alumnos Asignados

La actividad Alumnos Asignados ofrece la posibilidad de comprobar qué alumnos le aparecerán a cada usuario cuando inicie sesión en la aplicación Athena Escenas, sin necesidad de acceder a esta. Para ello se crea una galería a partir de la información de la base de datos referente a la relación entre el usuario seleccionado y los alumnos creados. En esta galería, si pulsamos sobre cualquier alumno mostrado (en la figura 3.23, podemos ver los *View Holders* que contienen a los alumnos), nos redirige a su pantalla de edición.

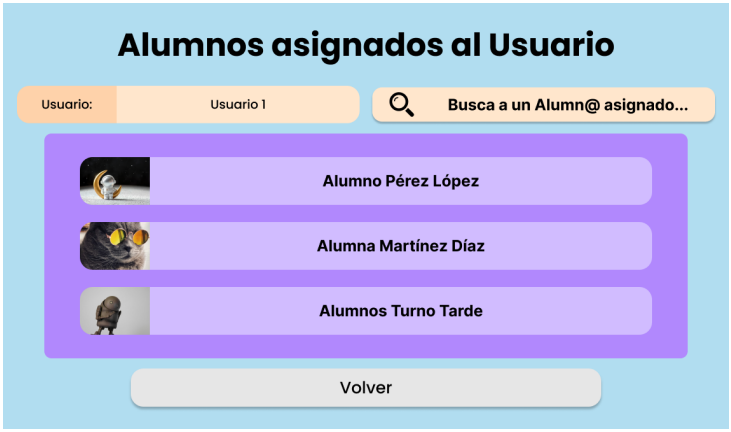

Figura 3.23: Vista de la pantalla de la actividad Alumnos Asignados en Athena Creaciones

– Eliminar Usuario

Por último en la sección de usuarios, la actividad Eliminar Usuario presenta una pantalla que sirve para confirmar la "eliminación" de un usuario del entorno. Esta eliminación consiste en cambiar un bit de información sobre el usuario que permite a las aplicaciones comprobar si debe mostrarse o no, de manera que si se "eliminara" un usuario por error, se podría recuperar sin problema. La implementación de esta actividad consiste únicamente en redirigir al usuario a la actividad de Edición Usuario, en caso de que se cancele la eliminación, o redirigirle a la actividad Galería Usuarios, en caso de que se confirme la eliminación, mediante la pulsación del boton "eliminar" que podemos ver en la figura 3.24, y siempre que la llamada a la API para ello sea correcta.

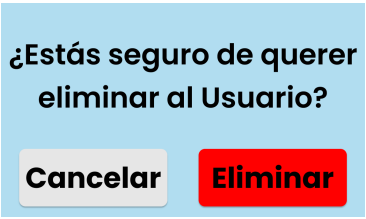

Figura 3.24: Vista de la pantalla de la actividad Eliminar Usuario en Athena Creaciones

- **Sección Escenas**

A continuación, se describe la implementación de las actividades que poseen herramientas de gestión de los datos relativos a las escenas.

– Galería Escenas

La actividad Galería Escenas permite elegir si se crea una nueva escena o se busca y selecciona una ya creada para usar las herramientas de gestión de datos sobre esta. Su implementación y funcionamiento son prácticamente idénticos a los de la actividad Galería Usuarios, mostrando en los *View Holder* información relativa a las escenas.

– Creación Escena

La actividad Creación Escena permite crear una escena a partir de un archivo de imagen, un nombre, una palabra, un conjunto de sílabas y un conjunto de fonemas (tanto las letras que representen cada sílaba, como las que representen cada fonema, tienen que estar separadas por

el carácter "_"). Su implementación es prácticamente idéntica a la de la actividad Creación Usuario, solo que en este caso, como datos obligatorios aparecen el archivo de imagen (se selecciona pulsando la imagen que se puede apreciar en la parte izquierda de la figura 3.25), el nombre y la palabra de la escena y los conjuntos de sílabas y fonemas de esta.

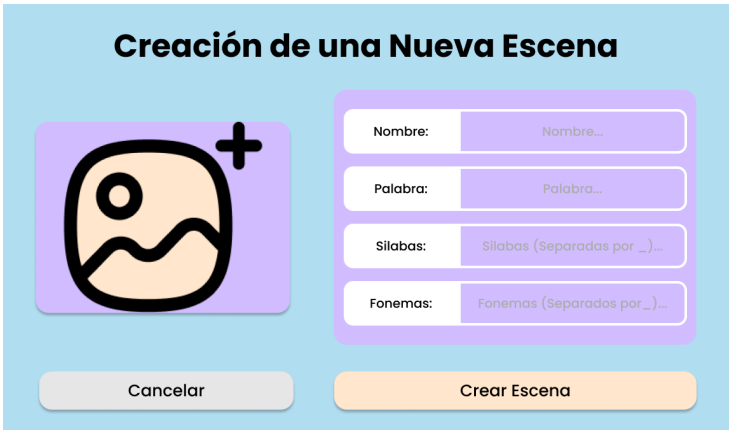

Figura 3.25: Vista de la pantalla de la actividad Creación Escena en Athena Creaciones

– Edición Escena

La actividad Edición Escena sirve de nexo entre la galería de escenas y las herramientas asociadas a ellas. Su implementación y visualización es idéntica a la de la actividad Edición Usuario, pero como podemos ver a continuación en la figura 3.26, se han añadido el archivo de imagen relativo a la escena, la representación de la palabra de la escena y nuevos botones para acceder a las herramientas de las escenas.

Figura 3.26: Vista de la pantalla de la actividad Edición Escena en Athena Creaciones

– Previsualización Escena

La actividad Previsualización Escena simula la reproducción de una escena en la aplicación Athena Escenas. Como ambas aplicaciones pueden ser ejecutadas en el mismo dispositivo, podemos comprobar que todos los parámetros asociados a la reproducción son válidos en él. Como dato a reseñar, no se puede acceder a esta herramienta hasta que la escena no tenga los datos de audio y de sincronización asociados a ella.

Para describir la implementación de esta actividad, vamos a mostrar los pasos que sigue una escena al reproducirse, una vez que se han obtenido los datos desde la API.

1. `reproducirEscena()`
   Este método, de tipo `runBlocking` [40], permite ejecutar los hilos de reproducción de audio y de texto, mientras el hilo principal queda bloqueado, para centrar los recursos en la ejecución de la escena.

2. `reproducirAudio()`
   Este método permite iniciar la reproducción del audio de la escena, utilizando un objeto de tipo `MediaPlayer` [36] que se ha instanciado e inicializado en los métodos de obtención y carga de información de la actividad.

3. `reproducirTexto()`
   Este método coordina la ejecución de los submétodos que lo componen, que van "iluminando" los elementos de la palabra según los valores de sincronización obtenidos en los métodos de obtención de información de la actividad. Estos valores representan los tiempos en milisegundos (ms).

   Como vimos anteriormente en la figura 3.2 y simplificado en la figura 3.27, el ciclo de reproducción de una escena se compone de varias etapas cuyos nombres representan el elemento lingüístico de la palabra que se va a iluminar en ella.

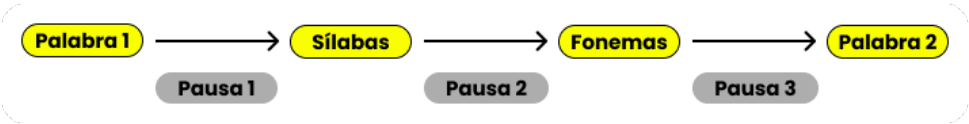

Figura 3.27: Simplificación del ciclo de reproducción de una escena

   Para lograr esa funcionalidad, `reproducirTexto()` llama al submétodo correspondiente siguiendo los valores de sincronización y atendiendo a las pausas entre ellos, para que de manera coordinada estos métodos puedan realizar la misma funcionalidad con los elementos lingüísticos que manejan.

   Los submétodos que componen a este método son: `mostrarFonemas()`, para la funcionalidad de los fonemas; `mostrarSilabas()`, para la funcionalidad de las sílabas; y `mostrarPalabraCompleta()`, para resaltar la palabra completa al final del ciclo e indicar que se ha terminado la reproducción del texto.

4. `comprobarFinalizacion()`
   Este último método tiene como objetivo comprobar tras un periodo de tiempo de medio segundo (500ms), que tanto la reproducción del audio, como la reproducción del texto han finalizado, para poder desbloquear el hilo principal y poder volver a reproducir la escena si así se desea.
   Para lograr esta funcionalidad, este método implementa un objeto de tipo `Handler` [41] (manejador) que está asociado a un objeto de tipo `Looper` [42], que proporcionar al manejador el periodo de refresco deseado. Y por último, para ejecutar el manejador, se utiliza el método `run()` de la interfaz no gráfica de Kotlin, `Runnable` [43], que permite a este poder ejecutarse en un hilo ajeno a los de reproducción.

Una vez descritos los métodos principales de la actividad, hay que añadir que la escena también ofrece redirigirnos a la actividad Ajustes Letra Escena, al pulsar el botón de la parte superior derecha que podemos ver en la figura 3.28.

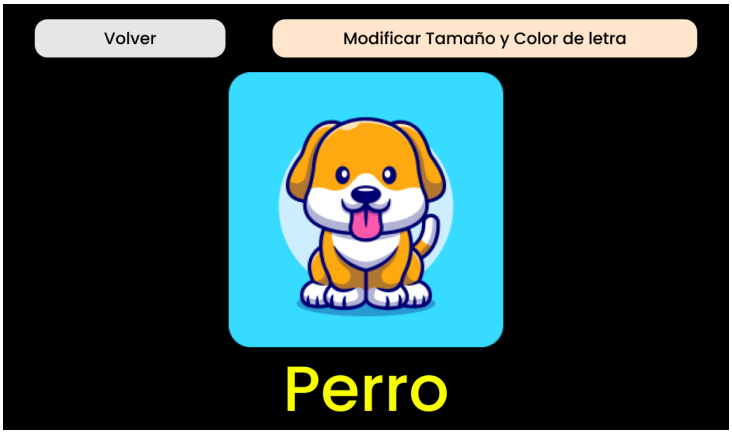

Figura 3.28: Vista de la pantalla de la actividad Previsualización Escena en Athena Creaciones

– Ajustes Letra Escena

La actividad Ajustes Letra Escena permite probar y editar el tamaño y el color del texto de la escena, como podemos ver en la figura 3.29.

Figura 3.29: Vista de la pantalla de la actividad Ajustes Letra Escena en Athena Creaciones

Para lograr esa funcionalidad, aparte de la obtención y carga de datos de todas las actividades, esta actividad tiene que comprobar tanto que lo introducido en el apartado de tamaño sea un número entero o decimal, como que el código de color introducido en formato hexadecimal sea válido en RGB. Una vez comprobados, se asignan esos parámetros al elemento gráfico de la pantalla y se obtiene *feedback* visual de lo introducido.

– Edición Picto-Lingüística

La actividad Edición Picto-Lingüística permite modificar los datos de la escena relativos a su archivo de imagen, nombre, palabra, conjunto de sílabas y conjunto de fonemas. Su funcionalidad y visualización es idéntica a la actividad Creación Escena, añadiendo que antes de proceder a la edición, se deben cargar los datos de la escena.

– Grabadora

La actividad Grabadora, como su propio nombre indica, permite grabar un audio a través del micrófono de la *tablet* donde se ejecuta la aplicación. Aunque, debido a *bugs* con determinados modelos de *tablets*, también permite cargar un archivo de audio almacenado en dicho dispositivo.

Como podemos ver más abajo en la figura 3.30, esta actividad se compone de dos funcionalidades: reproducción, en la parte superior, y grabación/carga de audio, en la parte inferior.

* La funcionalidad de reproducción es exactamente idéntica a la funcionalidad de reproducción de audio y texto de la actividad Previsualización Escena, difiriendo en que hasta que no se ha grabado o cargado un audio, no se activa el botón que permite su ejecución.

* La funcionalidad de grabación consiste en, mediante el uso de un objeto de la clase *MediaRecorder* [37], con los métodos que tiene definidos, y el micrófono de la *tablet* que ejecuta la aplicación, poder almacenar el audio requerido en un archivo y poder utilizarlo en la reproducción de una escena.
  La implementación de esta funcionalidad reside en los métodos `iniciarGrabacion()`, que usa el objeto `MediaRecorder` para iniciar la captura de audio; `pararGrabacion()`, que finaliza la captura de audio y guarda dicha captura en un archivo manejable por la actividad; y, por último, el método `eliminarGrabacion()`, que borra el archivo de audio.

* Debido a la aparición de *bugs* irreparables por parte del entorno, a la hora de grabar ficheros de audio en determinados modelos de *tablet*, se ha incluido la posibilidad de agregar un archivo de audio almacenado en el dispositivo que ejecuta la aplicación. Para ello se almacena el audio guardado en el sistema de archivos nativo de la *tablet*, y se utilizan los métodos `elegirMedioAudio()` para abrir el sistema de archivos y `onActivityResult()` para controlar los datos recibidos.

Figura 3.30: Vista de la pantalla de la actividad Grabadora en Athena Creaciones

– Sincronización

La actividad Sincronización permite dotar a la escena de parámetros numéricos de tiempo para ajustar el periodo de ejecución de los métodos de reproducción de texto y audio una escena. Estos tiempos permitirán que cuando se reproduzca el sonido de una sílaba/fonema determinado, se resalten las letras correspondientes a la vez. Como podemos ver más abajo, en la figura

3.31, la pantalla de la actividad se divide en dos zonas. A la izquierda aparecen los controles de reproducción del audio y del texto de la escena, para poder tener un *feedback* instantáneo de la sincronización; mientras que a la derecha, tenemos los elementos de introducción de tiempos (visto anteriormente en la figura 3.2 y simplificado en la figura 3.27).

La funcionalidad de reproducción es exactamente idéntica a la reproducción de audio y texto de la actividad Previsualización Escena, salvo en que los periodos de ejecución de los distintos submétodos de la reproducción vienen marcados por los valores que se introducen en la parte derecha de la pantalla.

Figura 3.31: Vista de la pantalla de la actividad Sincronización en Athena Creaciones

– Eliminar Escena

La ultima actividad de la sección de escenas, Eliminar Escena, tiene como funcionalidad "desactivar" la escena del entorno. Su implementación es totalmente idéntica a Eliminar Usuario.

- **Sección Alumnos**

Por último en la aplicación Athena Creaciones, se describe la implementación de las actividades que gestionan los datos relativos a alumnos.

– Galería Alumnos

La actividad Galería Alumnos permite elegir si se crea un nuevo usuario o se busca y selecciona uno ya creado. Su implementación y funcionamiento son idénticos a Galería Usuarios, mostrando en los *View Holder* información relativa a los alumnos.

– Creación Alumno

La actividad Creación Alumno permite crear un alumno a partir de un achivo de imagen, un nombre, unos apellidos, una fecha de nacimiento y un texto de información adicional. Su implementación es prácticamente idéntica a la de la actividad Creación Escena, solo que en este caso, los datos obligatorios son el archivo de imagen, el nombre y los apellidos del alumno.

– Edición Alumno

La actividad Edición Alumno sirve de nexo entre la galería de alumnos y las herramientas asociadas a estos. Su implementación y visualización es idéntica a la de la actividad Edición Escena, eliminando la visualización de la palabra y poniendo como título de la actividad la concatenación del nombre y del apellido del alumno.

– Información Alumno

La actividad Información Alumno, como podemos ver a continuación en la figura 3.32, es muy sencilla, ya que solo muestra por pantalla la datos guardados en el entorno relativos al alumno seleccionado.

Figura 3.32: Vista de la pantalla de la actividad Información Alumno en Athena Creaciones

– Edición Info. Alumno

La actividad Edición Info. Alumno permite modificar los datos relativos al alumno seleccionado. Su implementación, funcionalidad y visualización es idéntica a la de la actividad Creación Alumno, difiriendo en que hay que cargar los datos relativos al alumno, antes de editarlos.

– Asignación Alumno Usuarios

La actividad Asignación Alumno Usuarios tiene como objetivo permitir asignar o desasignar al alumno a un usuario. Permite configurar si este alumno le aparece o no al usuario, cuando este ultimo inicie sesión en la aplicación Athena Escenas.

Para conseguir esta funcionalidad, su implementación se basa en la creación de una galería a partir de la información, obtenida desde la base de datos, referente a la relación entre el alumno seleccionado y los usuarios creados. En esta galería, si pulsamos a cualquier usuario mostrado (en la figura 3.33, podemos ver los *View Holders* que contienen los usuarios), aparecerá o desaparecerá, en la parte izquierda del elemento que lo contiene, la confirmación visual si el alumno está asignado a ese usuario.

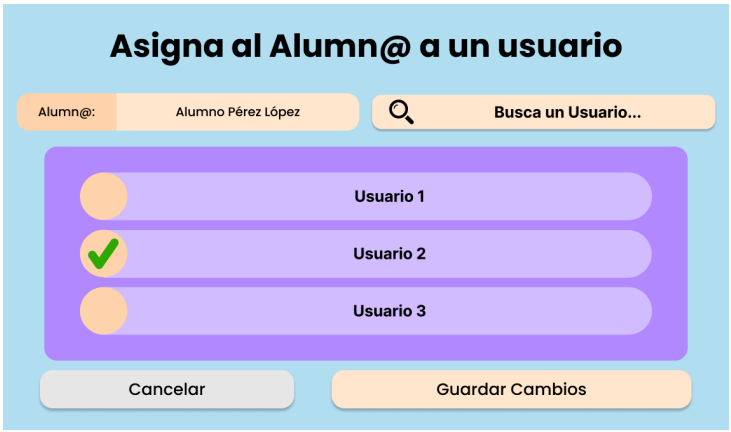

Figura 3.33: Vista de la pantalla de la actividad Alumnos Asignados en Athena Creaciones

– Asignación Escenas Alumno

La actividad Asignación Escenas Alumno tiene como finalidad permitir asignar o desasignar escenas al alumno seleccionado, es decir, configurar las escenas que le aparecerán al alumno una vez que sea seleccionado en la aplicación Athena Escenas.

Su implementación es idéntica a la de Asignación Alumno Usuarios divergiendo en que en los *View Holder* muestran información relativa a las escenas, como podemos ver en la figura 3.34.

Figura 3.34: Vista de la pantalla de la actividad Asignación Escenas Alumno en Athena Creaciones

– Eliminar Alumno

Por último, en la aplicación Athena Creaciones, tenemos la actividad Eliminar Alumno cuyo objetivo es "desactivar" el alumno del entorno. Su implementación es idéntica a la de Eliminar Usuario.

3.3.1.2 Implementación de Athena Escenas

Como ya se indicó en la introducción de la aplicación (sección 3.1.1), Athena Escenas es la aplicación del entorno orientada a que los niños reproduzcan las escenas en ella.

Para lograr esta funcionalidad, al igual que Athena Creaciones, la aplicación se sustenta en una estructura de actividades (figura 3.35), escritas en lenguaje Kotlin e interconectadas entre sí, que permiten la navegación entre las distintas pantallas y la reproducción de las escenas asignadas a los alumnos.

A continuación, describiremos el funcionamiento de cada una de las actividades y destacaremos los aspectos únicos de cada una.

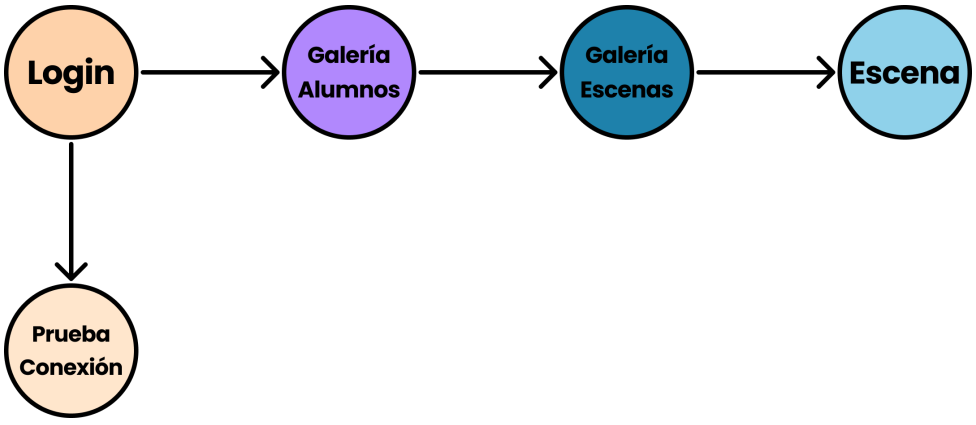

Figura 3.35: Esquema de actividades de Athena Escenas

- **Login**

  La actividad Login, cuya pantalla podemos ver a continuación en la figura 3.36, nos permite comprobar que el usuario que va a entrar a la aplicación esta registrado en el entorno, mediante el uso de unas credenciales (identificador de usuario y contraseña).

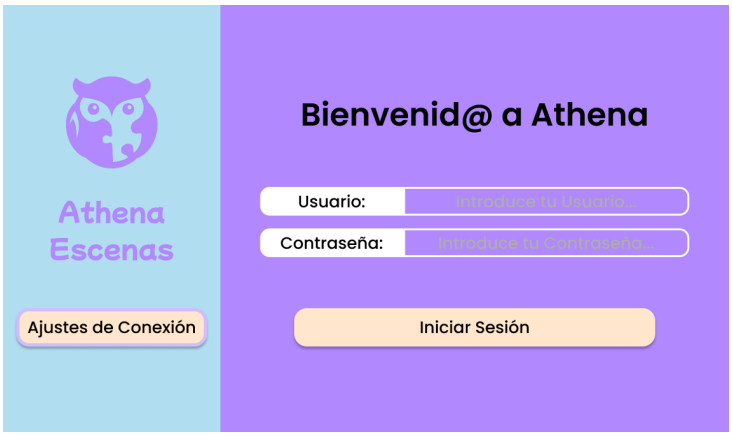

Figura 3.36: Vista de la pantalla de la actividad Login en Athena Escenas

  Su implementación es exactamente idéntica a la actividad Login de Athena Creaciones, variando únicamente en la parte gráfica de la actividad, que cambia de colores al ser una aplicación distinta y en vez de redirigir a la actividad de Selección Sección, redirige a la actividad Galería Alumno de esta aplicación.

- **Prueba Conexión**

  La actividad Prueba Conexión de la aplicación Athena Escenas, es completamente idéntica a la actividad Prueba Conexión de Athena Creaciones.

- **Galería Alumnos**

  La actividad Galería Alumnos de la aplicación Athena Escenas permite buscar y seleccionar un alumno que esté asociado al usuario que ha iniciado sesión.

  Como podemos ver a continuación en la figura 3.37, la pantalla de esta actividad es prácticamente idéntica a la actividad Galería Alumnos de Athena Creaciones, pero tanto esta pantalla como su implementación divergen en que esta actividad muestra únicamente los alumnos asignados al usuario que ha iniciado sesión y no permite la creación de un nuevo alumno.

Figura 3.37: Vista de la pantalla de la actividad Galería Alumnos en Athena Escenas

- **Galería Escenas**

  La actividad Galería Escenas de la aplicación Athena Escenas permite buscar y seleccionar una escena que esté asociada al alumno que hemos elegido en la actividad anterior.

  La implementación de esta actividad comparte similitud con la implementación de la actividad Galería Escenas de Athena Creaciones, pero como hemos visto anteriormente en la figura 3.5 de la presentación, esta galería no esta implementada a modo de lista que se desplaza.

  En la fase de diseño se decidió que cuando se le diera la *tablet* a un niño, este debería ver el mayor número de escenas posibles. Por ello, esta galería está implementada como una matriz de escenas de cinco elementos por fila y, si no caben todas ellas, la matriz se puede desplazar hacia arriba para descubrir aquellas que no se han mostrado.

- **Escena**

  La actividad Escena es la aplicación principal del entorno, aquella por cuya funcionalidad se ha creado el entorno alrededor.

  Esta actividad permite que el niño pueda reproducir una escena de una manera sencilla, ya que solo tiene que pulsar la imagen central en pantalla. Además, como hemos visto anteriormente en la figura 3.1, en la parte superior se ha añadido una herramienta de navegación a la escena anterior y posterior que tiene asignadas el alumno, para que de esta manera, no haga falta volver a la galería de escenas para reproducir las demás.

  La implementación de esta actividad es idéntica a la de la Previsualización Escena de Athena Creaciones, pero como hemos mencionado anteriormente, en vez de implementar la funcionalidad del botón "volver" y del botón de modificar la letra de la escena, se ha incluido un mecanismo que permite navegar a la escena anterior y posterior.

3.3.2 Aplicaciones ejecutadas en el servidor local

Una vez que se ha terminado con la descripción de las aplicaciones ejecutadas en las tablets, es turno de las aplicaciones que se ejecutan en el servidor local: el SGBD de la base de datos que recoge toda la información del entorno, el contenedor de la aplicación Docker que lo encapsula y la API del entorno.

- **Base de Datos**

  Como se mencionó en el apartado del Nivel de Datos de la Arquitectura del entorno, para poder almacenar e implementar el SGBD del entorno de una manera sencilla, se optó por configurar un contenedor gestionado por la aplicación Docker.

  Para configurar un contenedor con la base de datos y el SGBD desde cero, hay que seguir una serie de pasos:

  1. Instalar Docker ya sea en un ordenador que utilice Windows [44] o en un ordenador Apple con MacOS [45].

  2. Ejecutar el siguiente comando en la terminal del sistema operativo del equipo elegido (sea Windows o MacOS):

     ```
     docker run -name Database_Athena -p 3306:3306 -e MYSQL_ROOT_PASSWORD
     = Athena_SuperUser -d mysql
     ```

3. Instalar un programa de administración de bases de datos relacionales (RDBMS). Se recomienda el uso del programa DataGrip de la empresa JetBrains [46], pero unas alternativas gratuitas y de software libre serian HeidiSQL [47] y DBeaver [48].

4. Conectarse a la base de datos del contenedor en el RDBMS con los siguientes datos:
   - host: 'localhost' o host: "IP del equipo que contiene la base de datos"
   - port: '3306',
   - user: 'root',
   - password: 'Athena_SuperUser',
   - database: 'TFG'

5. Por último, haciendo uso de las herramientas del RDBMS, crear las tablas necesarias para el correcto almacenamiento de los datos del entorno. Estas tablas se dividen en dos grupos: tablas transaccionales, las cuales almacenan la información dinámica del entorno; y tablas relacionales, las cuales nos permiten vincular información de las tablas anteriores. El resultado final de la creación tiene que quedar como el esquema que podemos ver en la figura 3.38. A continuación, se describirá la implementación de estas tablas.

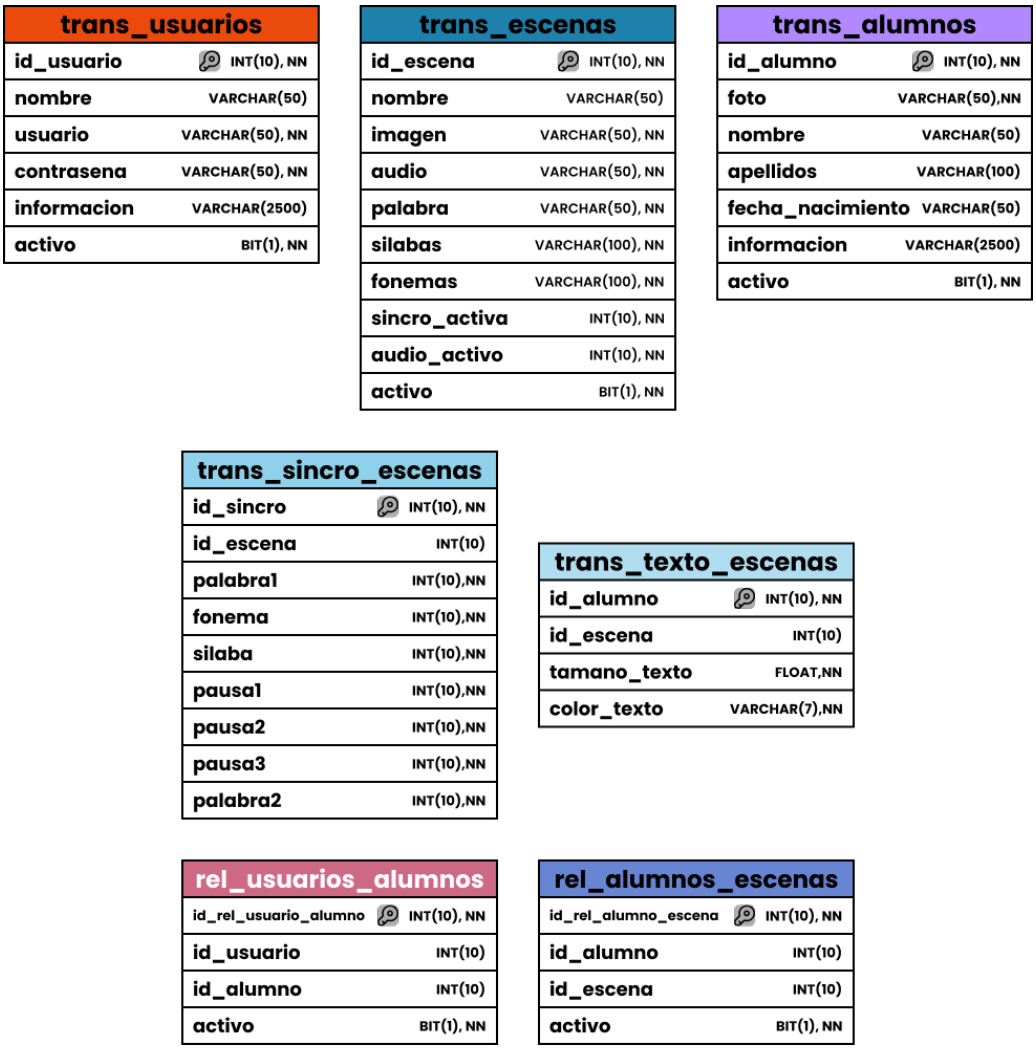

Figura 3.38: Esquema de las tablas de la Base de Datos del entorno

- trans_alumnos

  La tabla `trans_alumnos` es una tabla transaccional que almacena los datos relativos a los alumnos del entorno. Se compone de las columnas:

  * id_alumno: entero que identifica cada registro de alumno.
  * foto: cadena de texto que almacena el nombre del archivo de imagen del alumno y su extensión.
  * nombre: cadena de texto que almacena el nombre del alumno.
  * apellidos: cadena de texto que almacena los apellidos del alumno.
  * fecha_nacimiento: cadena de texto que almacena la fecha de nacimiento del alumno.
  * información: cadena de texto que almacena la información adicional del alumno.
  * activo: bit que permite comprobar si el alumno esta activado en el entorno.

- trans_escenas

  La tabla `trans_escenas` es una tabla transaccional que almacena los datos relativos a los las escenas del entorno. Se compone de las columnas:

  * id_escena: entero que identifica cada registro de escena.
  * nombre: cadena de texto que almacena el nombre de la escena.
  * imagen: cadena de texto que almacena el nombre del archivo de imagen de la escena y su extensión.
  * audio: cadena de texto que almacena el nombre del archivo de audio de la escena y su extensión.
  * palabra: cadena de texto que almacena la palabra de la escena.
  * sílabas: cadena de texto que almacena el conjunto de sílabas que componen la palabra de la escena.
  * fonemas: cadena de texto que almacena el conjunto de fonemas que componen la palabra de la escena.
  * sincro_activa: bit que permite comprobar si la escena tiene parámetros de sincronización.
  * audio_activo: bit que permite comprobar si la escena tiene audio asignado.
  * activo: bit que permite comprobar si la escena esta activa en el entorno.

- trans_usuarios

  La tabla `trans_alumnos` es una tabla transaccional que almacena los datos relativos a los usuarios del entorno. Se compone de las columnas:

  * id_usuario: entero que identifica cada registro de usuario.
  * nombre: cadena de texto que almacena el nombre del usuario.
  * usuario: cadena de texto que almacena la credencial de usuario de inicio de sesión del usuario.
  * contraseña: cadena de texto que almacena la credencial de contraseña de inicio de sesión del usuario.
  * información: cadena de texto que almacena la información adicional del usuario.
  * activo: bit que permite comprobar si el usuario esta activado en el entorno.

- trans_sincro_escenas

  La tabla `trans_sincro _escenas` es una tabla transaccional que almacena los datos relativos a la sincronización entre el audio y el texto de las escenas:

  * id_sincro: entero que identifica cada registro de sincronización de escena.
  * id_escena: entero que almacena el identificador de escena a la que esta asociada la información.
  * palabra1: entero que almacena el valor del tiempo de reproducción de la palabra por primera vez en la reproducción de la escena.
  * fonema: entero que almacena el valor del tiempo de reproducción de cada fonema en la reproducción de la escena.
  * sílaba: entero que almacena el valor del tiempo de reproducción de cada sílaba en la reproducción de la escena.
  * pausa1: entero que almacena el valor del tiempo de la primera pausa en la reproducción de la escena.
  * pausa2: entero que almacena el valor del tiempo de la segunda pausa en la reproducción de la escena.
  * pausa3: entero que almacena el valor del tiempo de la tercera pausa en la reproducción de la escena.
  * palabra2: entero que almacena el valor del tiempo de reproducción de la palabra por segunda vez en la reproducción de la escena.

- trans_texto_escenas

  La tabla `trans_texto _escenas` es una tabla transaccional que almacena los datos relativos a la tipografía de las escenas:

  * id_texto: entero que identifica cada registro de datos de texto de escena.
  * id_escena: entero que almacena el identificador de escena a la que esta asociada la información.
  * tamano_texto: valor numérico con coma flotante que almacena el valor del tamaño del texto de la escena.
  * color_texto: cadena de texto que almacena el código del color del texto de la escena.

- rel_alumnos_escenas

  La tabla `rel_alumnos_escenas` es una tabla relacional que almacena los datos relativos a la vinculación de alumnos con escenas:

  * id_rel_alumno_escena: entero que identifica cada registro de datos de relación entre alumnos y escenas.
  * id_alumno: entero que almacena el identificador de alumno al que esta asociado la información.
  * id_escena: entero que almacena el identificador de escena a la que esta asociada la información.
  * activo: bit que permite comprobar si la relación esta activa en el entorno.

- rel_usuarios_alumnos

  La tabla `rel_usuarios_alumnos` es una tabla relacional que almacena los datos relativos a la vinculación de usuarios con alumnos:

  - id_rel_usuario_alumno: entero que identifica cada registro de datos de relación entre usuarios y alumnos.
  - id_usuario: entero que almacena el identificador de usuario al que esta asociada la información.
  - id_alumno: entero que almacena el identificador de alumno al que esta asociado la información.
  - activo: bit que permite comprobar si la relación esta activa en el entorno.

- **API**

  Una API, o interfaz de programación de aplicaciones, es un conjunto de reglas definidas que permiten a las diferentes aplicaciones comunicarse entre sí. Actúa como una capa intermedia que procesa las transferencias de datos entre sistemas, permitiendo a las aplicaciones del entorno obtener y modificar la información de la base de datos de una manera ordenada y sencilla [49].

  Para implementar la API del entorno, se ha optado por programarla en lenguaje Node.js, ya que es un entorno rápido, escalable y eficiente. Además, gracias al uso de las bibliotecas `Express` y `MySQL` de este lenguaje, la conexión de esta tanto con la base de datos, como con las aplicaciones, se simplifica en gran medida.

  Para crear la API desde cero, se puede seguir el siguiente tutorial [50].

  Una vez que tenemos creada la API, pasamos a definir las comunicaciones necesarias para que el entorno funcione correctamente. Estas comunicaciones se llaman "solicitudes" (*API Requests*).

  Una solicitud API consiste en una petición que realiza una aplicación a dicha API para recuperar datos o ejecutar una acción coordinada por esta. Una solicitud API consta de los siguientes elementos, pero solo se describirán los importantes para el entorno:

  – Método

    El método indica el tipo de acción que se solicita a la API. Por ejemplo, el método *GET* se utiliza para recuperar datos, el método *POST* se utiliza para crear datos nuevos, y el método *PUT* se utiliza para actualizar datos existentes. En el entorno, todas las peticiones son de tipo *POST*, aunque el objetivo de la petición no sea crear datos nuevos, debido a que estas son las peticiones que mejor rendimiento tienen en las aplicaciones del entorno.

  – *URL*

    La URL especifica el recurso al que se solicita la acción. En el entorno, las *URLs* utilizadas se componen de los siguientes elementos, en el orden en el que aparecen:

    1. Dirección del servidor: Si la API esta en el mismo equipo que la base de datos, la *URL* empezará por `localhost:3000`. En su defecto, se deberá sustituir la palabra `localhost` por la dirección IP de dicho equipo (ej. 188.18.10.8:3000).

    2. La cadena `api/v1`: Para indicar que estamos apuntando a la API y a la versión número "1" de esta.

    3. El tipo de solicitud: Puede ser de tipo "conex", para aspectos relacionados con la conexión a la API; de tipo "get", para obtener datos de la base de datos, de tipo "check", para comprobar algún dato en la base de datos; de tipo "add", para añadir información a la

base de datos; de tipo “update”, para actualizar la información de la base de datos; de tipo “activate”, para volver a activar algún dato desactivado en las aplicaciones Android del entorno; o, por último, de tipo “delete”, para desactivar algún dato en las aplicaciones Android del entorno.

4. A que tipo de información esta dirigida la petición: a datos de usuarios, a datos de escenas o a datos de alumnos.

5. Información adicional: Esta puede aparecer o no y sirve para indicar si se quiere gestionar un aspecto concreto de la información.

Un ejemplo de una ruta en el entorno, para la obtención de la lista de alumnos que hay creados, sería: `localhost:3000/api/v1/get/alumno/lista`.

– Cuerpo

Aunque normalmente es un elemento opcional, todas las solicitudes API del entorno poseen cuerpo. Se ha optado por implementarlas de esta maneras ya que facilita en gran cantidad la especificación de las gestiones que se deben hacer en la base de datos del entorno, segun la información enviada en él.

Una vez la aplicación envía la solicitud API, esta la procesa y devuelve una respuesta. Una respuesta API esta compuesta de múltiples elementos, pero debido a que en el entorno la mayoría de estos no se utilizan implícitamente en el código, únicamente vamos a describir los usados: el código de estado y el cuerpo.

– Código de estado

El código de estado de una respuesta API indica el éxito o el fracaso de la solicitud como número de tres dígitos. Permite controlar la ejecución de las aplicaciones según la información que proporciona.

Los códigos de estado más comunes son:

* 200 *OK*: La solicitud se ha completado correctamente.
* 400 *Bad Request*: La solicitud contiene errores.
* 401 *Unauthorized*: La solicitud no está autorizada.
* 404 *Not Found*: El recurso solicitado no se encuentra.
* 500 *Internal Server Error*: Se ha producido un error en el servidor.

– Cuerpo

Al igual que en las solicitudes, el cuerpo es un elemento opcional de las respuestas. Pero en el entorno, se ha decidido que todas las respuestas tengan cuerpo para facilitar la integración de la información en las de estructuras de control de las aplicaciones que la manejan.

Para terminar la implementación de la API, y siguiendo el esquema que podemos ver debajo en la figura 3.39, se van a describir los siete distintos tipos de solicitudes que se usan en el entorno. Estos tipos se van a describir por funcionamiento, según el orden en el que aparecen en la implementación de la API:

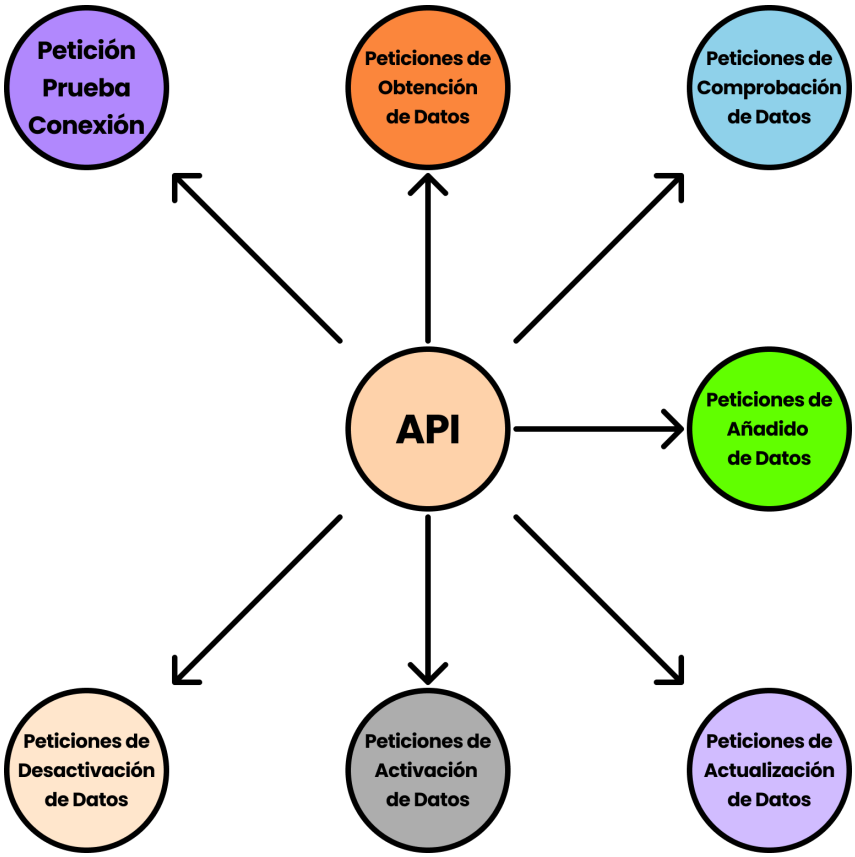

Figura 3.39: Tipos de solicitudes de la API del entorno

– **Petición de Prueba de Conexión**

La llamada de prueba de conexión es la que permite a las aplicaciones Android del entorno comprobar si la ruta que tienen asignada a la API es correcta y está bien configurada.

* Cuerpo Solicitud: "Comprobador" → Cadena de texto que permite comprobar si el uso de los cuerpos de las solicitudes a la API están bien configurados.

* Cuerpo Respuesta: Código → "1" si el "comprobador" es correcto y en su defecto, "-1".

– **Peticiones de Obtención de Datos**

Las llamadas de obtención de datos son las que permiten a las aplicaciones Android del entorno obtener la estructura de información deseada de la base de datos.

* Cuerpo Solicitud: "Filtro" → Información que diferencia a la estructura de datos requerida de las otras almacenadas.

* Cuerpo Respuesta: Estructura de datos → Estructura de datos deseada y que representa los datos introducidos en el cuerpo de la solicitud.

– **Peticiones de Comprobación de Datos**

Las llamadas de comprobación de datos son las llamadas que permiten a las aplicaciones Android del entorno comprobar si hay una estructura de datos con la información que se envía.

* Cuerpo Solicitud: "Filtro" → Información que diferencia a la estructura de datos requerida de las otras almacenadas.

* Cuerpo Respuesta: Información → Si la estructura de datos ya existe, envía los datos relativos a esta. En su defecto, envía un código "0".

– **Peticiones de Añadido de Datos**

Las llamadas de añadido de datos son las llamadas que permiten a las aplicaciones Android del entorno añadir información a la API mediante el envío de una estructura de datos con la información deseada.

* Cuerpo Solicitud: Estructura de Datos → Estructura de datos que representa los datos a añadir a la API.

* Cuerpo Respuesta: Información → Si la estructura de datos se ha añadido correctamente, envía el identificador relativo a esta estructura. En su defecto, envía un código "0".

– **Peticiones de Actualización de Datos**

Las llamadas de actualización de datos son las llamadas que permiten a las aplicaciones Android del entorno sobrescribir información a la API mediante el envío de una estructura de datos con las modificaciones deseadas.

* Cuerpo Solicitud: Estructura de Datos → Estructura de datos que representa los datos a sobrescribir en la API.

* Cuerpo Respuesta: Código → Si la estructura de datos se ha actualizado correctamente, envía un código "1". En su defecto, envía un código "0".

– **Peticiones de Activación de Datos**

Las llamadas de activación de datos son las llamadas que permiten a las aplicaciones Android del entorno activar estructuras de datos con la información deseada en la API mediante el envío de la información que diferencia a la estructura de datos de las demás.

* Cuerpo Solicitud: "Filtro" → Información que diferencia a la estructura de datos requerida de las otras almacenadas .

* Cuerpo Respuesta: Código → Si la estructura de datos se ha activado correctamente, envía un código "1". En su defecto, envía un código "0".

– **Peticiones de Desactivación de Datos**

Las llamadas de desactivación de datos son las llamadas que permiten a las aplicaciones Android del entorno desactivar estructuras de datos con la información deseada en la API mediante el envío información que diferencia a la estructura de datos de las demás.

* Cuerpo Solicitud: "Filtro" → Información que diferencia a la estructura de datos requerida de las otras almacenadas .

* Cuerpo Respuesta: Código → Si la estructura de datos se ha desactivado correctamente, envía envía un código "1". En su defecto, envía un código "0".

3.3.3 Aplicaciones ejecutadas en la nube

Como ultimo punto del entorno, vamos a describir las aplicaciones que se ejecutan en la nube.

Como se ha mencionado en el apartado 3.2.2, en la nube la aplicación que se ejecuta es Firebase.

Firebase es la plataforma de desarrollo de aplicaciones móviles de Google y esto nos permite que su integración con las aplicaciones Android sea óptima gracias a las herramientas que integra, ya que ambos sistemas son de la misma empresa.

En el entorno, debido a que los archivos audiovisuales requieren de un almacenamiento mayor que una base de datos y de una implementación de este bastante compleja, el servidor local no es una opción óptima.

El conjunto de estas premisas, hizo que en la fase de diseño del entorno se optara por el *Cloud Hosting* de Firebase. Este hosting, es una solución robusta y escalable para el almacenamiento de contenido audiovisual en aplicaciones Android. Ofrece una serie de beneficios como escalabilidad, rendimiento, seguridad y facilidad de uso. Además de tener un plan gratuito con suficiente capacidad de almacenamiento para esta versión del entorno.

Para poder implementar esta herramienta hay que seguir los siguientes pasos:

1. Crear una cuenta de Google [51].
2. Una vez que ya esté creada la cuenta, hay que registrar el proyecto en Firebase.
   - Accede a la consola de Firebase [52] y cree un nuevo proyecto.
   - Selecciona la plataforma Android y define el nombre del paquete del entorno en la nube.
   - Modifica las reglas del paquete del entorno en la nube, para que se puedan hacer operaciones de lectura y escritura en él.
   - Define la estructura de carpetas deseada en el paquete.
3. Por ultimo, añadir en el *IDE* Android Studio las aplicaciones a firebase:
   - Navegar con cada una de las aplicaciones por los menús: *Tools >Firebase >Cloud Storage for Firebase >Get Started With Cloud Storage*
   - Seguir las instrucciones del *IDE*

Una vez completados todos los pasos, ya estaría todo el entorno implementado.

Capítulo 4

Conclusiones y líneas futuras

4.1 Conclusiones

La lectura y la escritura son habilidades esenciales para la integración de las personas. Las dificultades para aprender a leer y escribir pueden tener implicaciones a todos los niveles: social, educativo o profesional, entre otros.

Por esta razón, en este TFG se ha diseñado e implementado Athena, un entorno diseñado para brindar a los niños con TEA apoyo en el aprendizaje de la lectoescritura y, a las personas que trabajan con ellos, herramientas para poder personalizar ese apoyo. Está formado por:

- Athena Escenas, una aplicación Android donde los niños pueden reproducir en una *tablet* las escenas visuales de una manera muy sencilla y atractiva para ellos. Cuando se selecciona una escena, se visualiza una imagen y se reproduce secuencialmente audio sincronizado con la palabra completa, después con las sílabas, posteriormente con los fonemas que la componen, y por último con la palabra completa de nuevo. Cada vez que suena la palabra, cada sílaba o cada fonema, se resalta la secuencia de letras que se corresponde con el sonido emitido. De esta forma se llama la atención al niño sobre las letras que representan el sonido concreto que está escuchando.

- Athena Creaciones, herramienta implementada en el sistema operativo Android que permite a los logopedas y familiares introducir la información de los niños en el sistema, y asignarles y adaptarles las escenas de forma intuitiva y fácil. Al introducir las palabras nuevas, permite indicar cual es la división por sílabas y por fonemas y definir los tiempos de cada sílaba/fonema para sincronizar el resaltado de las letras con la emisión de los sonidos.

  Cada profesor podrá crear tantos alumnos como sea necesario y podrá configurar para cada uno independientemente las escenas que más le llamen la atención. De esta forma se facilitará la enseñanza de la lectoescritura en base a la curiosidad del niño, lo que aumentará su motivación.

- Un servidor local donde se encuentran la API y la base de datos del entorno. La API, implementada en lenguaje Node.js, es la herramienta encargada de proporcionar a las aplicaciones el acceso a la información del entorno de una manera sencilla y ordenada a través de operaciones llamadas peticiones. La base de datos, implementada en lenguaje MySQL, es la herramienta encargada de almacenar toda la información relativa al entorno y de proporcionar mecanismos de gestión de esos datos a demanda.

- Un almacenamiento en la nube, gestionado por la plataforma Firebase de Google, para poder almacenar todo el contenido audiovisual que será utilizado por las aplicaciones en sus distintas funcionalidades.

4.2 Líneas futuras

Tras finalizar el desarrollo de esta versión del entorno Athena, quedan pendientes aspectos que pueden mejorar la funcionalidad del entorno, como por ejemplo:

- Implementar la API en un servidor que permita conectarse a él desde internet. De esta manera no se necesitaría estar en la misma red local que el equipo que actúa de servidor local y los niños podrían reproducir las escenas en cualquier lugar.
- Mejorar la lógica de usuarios del entorno, es decir, aportar una jerarquía de permisos de usuarios para que no todos puedan acceder a datos sensibles del entorno.
- Implementar el almacenamiento de los contenidos audiovisuales en el mismo servidor externo que la API y que todas las conexiones estén más centralizadas y sin depender de planes gratuitos de aplicaciones.
- Implementar una aplicación web que permita la asignación de escenas a los alumnos y estos alumnos a usuarios, desde cualquier dispositivo que disponga de conexión a internet.
- Implementar un sistema de recogida de datos del uso de la aplicación, como por ejemplo, el número de veces se ha reproducido una escena, para que el logopeda pueda valorar la evolución del niño a partir de los datos.
- Implementar un sistema de evaluación que permita comprobar si el alumno es capaz de reconocer la imagen que ve en las escenas, una vez que la haya reproducido un número determinado de veces.
- A medida que la conversión de texto en voz vaya avanzando y desarrolle reproducción de sílabas y fonemas, desarrollar un algoritmo que permita separar directamente la palabra en sílabas y fonemas, para que no sea necesario grabar voces ni sincronizar el audio, sino que se sincronice automáticamente, facilitando de esta forma la inserción de escenas nuevas.
- Implementar la posibilidad de resaltar varios objetos en una misma imagen y que se puedan reproducir los audios y textos asociados a cada uno independientemente. Por ejemplo, si una imagen tiene dos animales, al pulsar encima de cada uno , que se resalte su nombre y se oiga la transcripción fonética de este.
- Desarrollar aplicaciones para los dispositivos móviles que utilicen IOS.
- Mejorar la creación de escenas para que se pueda crear una directamente, haciendo una foto con el dispositivo y etiquetando la/las palabras a reproducir.
- Permitir la configuración de colores en la aplicación según las necesidades de los niños con TEA, ya que en ciertos espectros, pueden ser muy sensibles a ciertos colores.
- Implementar en Athena Escenas que se pueda acceder directamente a las escenas del alumno deseado a través de un código particular para cada uno de ellos, o con la lectura de un código QR generado por el entorno.

Bibliografía

[1] M. King, M. A. Romski y M. A. Sevcik, "Growing up with AAC in the digital age: a longitudinal profile of communication across contexts from toddler to teen", *Augmentative and Alternative Communication*, vol. 36, n.º 2, págs. 128-141, 2020.

[2] J. Light y D. McNaughton, "Literacy intervention for individuals with complex communication needs", *Topics in Language Disorders*, vol. 40, n.º 4, págs. 306-322, 2020.

[3] *What is Autism Spectrum Disorder? | CDC*, https://www.cdc.gov/ncbddd/autism/facts.html [Último acceso 31 de enero de 20242].

[4] K. Erikson, "Reading Comprehension in AAC", *The ASHA Leader*, vol. 8, n.º 12, págs. 6-9, 2003.

[5] A. Hynan, J. Murray y J. Goldbart, "'Happy and excited': Perceptions of using digital technology and social media by young people who use augmentative and alternative communication", *Child Language Teaching & Therapy*, vol. 30, n.º 2, págs. 175-186, 2014.

[6] J. Light, K. M. Wilkinson, A. Thiessen, D. R. Beukelman y S. K. Fager, "Designing effective AAC displays for individuals with developmental or acquired disabilities: State of the science and future research directions", *Alternative Communication*, vol. 35, n.º 1, págs. 42-55, 2019.

[7] K. Drager, J. Light, N. Muttiah et al., "AAC Technologies with Visual Scene Displays and "Just in Time" Programming and Symbolic Communication Turns Expressed by Students with Severe Disability", *Journal of Intellectual Developmental Disabilities*, vol. 44, n.º 3, págs. 321-336, 2019.

[8] C. Holyfield, J. Caron y J. Light, "Programing AAC just-in-time for beginning communicators: the process", *Augmentative and Alternative Communication*, vol. 35, n.º 4, págs. 309-318, 2019.

[9] O. Leonet, M. Orcasitas-Vicandi, A. Langarika-Rocafort, N. Idoiaga y G. Román, "A Systematic Review of Augmentative and Alternative Communication Interventions for Children Aged From 0 to 6 Years", *Language, Speech, and Hearing Services in Schools*, vol. 53, n.º 3, págs. 894-920, 2022.

[10] J. Light, D. McNaughton y E. Jakobs, "RERC on AAC: Rehabilitation Engineering Research Center on Augmentative and Alternative Communication", *Design of transition to literacy (T2L) decoding feature*, 2019.

[11] J. Caron, J. Light y D. McNaughton, "Effects of an AAC App with Transition to Literacy Features on Single-Word Reading of Individuals with Complex Communication Needs", *Research and Practice for Persons with Severe Disabilities*, vol. 45, n.º 2, págs. 115-131, 2020.

[12] K. A. Erickson, P. Hatch y S. Clendon, "Literacy, assistive technology, and students with significant disabilities", *Focus on Exceptional Children*, vol. 42, n.º 5, págs. 1-16, 2010.

[13] B. Fossett y P. Mirenda, "Sight word reading in children with developmental disabilities: A comparison of paired associate and picture-to-text matching instruction", *Research in Developmental Disabilities*, vol. 27, n.º 4, págs. 411-429, 2006.

[14] J. G. Caron, J. Light, C. Holyfield y D. McNaughton, "Effects of dynamic text in an AAC app on sight word reading for individuals with autism spectrum disorder", *Augmentative and Alternative Communication*, vol. 34, n.º 2, págs. 143-154, 2018.

[15] J. Caron, J. Light, C. Holyfield, C. Knudtson y D. McNaughton, "Grid Displays to Literacy: Effect of dynamic text on word reading for individuals with ASD [Presentation]", *American Speech-Language-Hearing Association (ASHA)*, 2016.

[16] J. Light, J. Caron y D. McNaughton, "Evidence-based intervention & apps to improve literacy outcomes for children with autism who require AAC [Presentation]", *American Speech-Language-Hearing Association (ASHA) Conference*, 2016.

[17] J. Light, D. McNaughton, T. Jakobs y D. Hershberger, "Investigating AAC technologies to support the transition from graphic symbols to literacy.", *RERC on AAC: Rehabilitation Engineering Research Center on Augmentative and Alternative Communication*, 2014.

[18] *What is three-tier architecture? | IBM*, Disponible en: https://www.ibm.com/topics/three-tier-architecture [Último acceso 27 de enero de 2024].

[19] *Diseño | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/topics/ui/declaring-layout?hl=es-419 [Último acceso 27 de enero de 2024].

[20] *What is MySQL? | MySQL 8.0 Reference Manual*, Disponible en: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html [Último acceso 27 de enero de 2024].

[21] *Docker: Accelerated Container Application Development*, Disponible en: https://www.docker.com/ [Último acceso 27 de enero de 2024].

[22] *What is a container? | Docker*, Disponible en: https://www.docker.com/resources/what-container/ [Último acceso 27 de enero de 2024].

[23] *Firebase | Google's Mobile and Web App Development Platform*, Disponible en: https://firebase.google.com/?hl=es [Último acceso 27 de enero de 2024].

[24] *Enfoque de prioridad de Kotlin en Android | Android Developers*, Disponible en: https://developer.android.com/kotlin/first?hl=es-419 [Último acceso 27 de enero de 2024].

[25] *Node.js — About Node.js*, Disponible en: https://nodejs.org/en/about [Último acceso 27 de enero de 2024].

[26] *Figma: The Collaborative Interface Design Tool*, Disponible en: https://www.figma.com/ [Último acceso 28 de enero de 2024].

[27] *Descripción general de los recursos de las apps | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/topics/resources/providing-resources?hl=es-419 [Último acceso 28 de enero de 2024].

[28] *Capa de datos | Desarrolladores de Android*, Disponible en: https://developer.android.com/topic/architecture/data-layer?hl=es-419 [Último acceso 28 de enero de 2024].

[29] *Introducción a las actividades | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/components/activities/intro-activities?hl=es-419 [Último acceso 28 de enero de 2024].

[30] *Working with JSON - Learn web development | MDN Mozilla*, Disponible en: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON [Último acceso 28 de enero de 2024].

[31] *Classes | Kotlin Documentation*, Disponible en: https://kotlinlang.org/docs/classes.html [Último acceso 28 de enero de 2024].

[32] *Interfaces | Kotlin Documentation*, Disponible en: https://kotlinlang.org/docs/interfaces.html [Último acceso 28 de enero de 2024].

[33] *Redmi Pad - La Pad más fun | Xiaomi España*, Disponible en: https://www.mi.com/es/product/redmi-pad/ [Último acceso 29 de enero de 2024].

[34] *Retrofit*, Disponible en: https://square.github.io/retrofit/ [Último acceso 29 de enero de 2024].

[35] *Corrutinas de Kotlin en Android | Android Developers*, Disponible en: https://developer.android.com/kotlin/coroutines?hl=es-419 [Último acceso 29 de enero de 2024].

[36] *Descripción general de MediaPlayer | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/topics/media/mediaplayer?hl=es-419 [Último acceso 30 de enero de 2024].

[37] *Descripción general de MediaRecorder | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/topics/media/mediarecorder?hl=es-419 [Último acceso 30 de enero de 2024].

[38] *Descripción general de los eventos de entrada | Desarrolladores de Android*, Disponible en: https://developer.android.com/develop/ui/views/touch-and-input/input-events?hl=es-419 [Último acceso 29 de enero de 2024].

[39] *Cómo crear listas dinámicas con RecyclerView | Desarrolladores de Android*, Disponible en: https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=es-419 [Último acceso 29 de enero de 2024].

[40] *runBlocking | Kotlin Language API*, Disponible en: https://kotlinlang.org/api/kotlinx.coroutines/kotlinx-coroutines-core/kotlinx.coroutines/run-blocking.html [Último acceso 30 de enero de 2024].

[41] *Handler | Android Developers*, Disponible en: https://developer.android.com/reference/kotlin/android/os/Handler [Último acceso 31 de enero de 2024].

[42] *Looper | Android Developers*, Disponible en: https://developer.android.com/reference/kotlin/android/os/Looper [Último acceso 31 de enero de 2024].

[43] *Runnable | Android Developers*, Disponible en: https://developer.android.com/reference/java/lang/Runnable?hl=en [Último acceso 31 de enero de 2024].

[44] *Install Docker Desktop on Windows | Docker Docs*, Disponible en: https://docs.docker.com/desktop/install/windows-install/ [Último acceso 31 de enero de 2024].

[45] *Install Docker Desktop on Mac | Docker Docs*, Disponible en: https://docs.docker.com/desktop/install/mac-install/ [Último acceso 31 de enero de 2024].

[46] *DataGrip: The Cross-Platform IDE for Databases & SQL by JetBrains*, Disponible en: https://www.jetbrains.com/datagrip/ [Último acceso 31 de enero de 2024].

[47] *HeidiSQL - MariaDB, MySQL, MSSQL, PostgreSQL and SQLite made easy*, Disponible en: https://www.heidisql.com/ [Último acceso 31 de enero de 2024].

[48] *DBeaver Community | Free Universal Database Tool*, Disponible en: https://dbeaver.io/ [Último acceso 31 de enero de 2024].

[49] *¿Qué es una interfaz de programación de aplicaciones (API)? | IBM*, Disponible en: https://www.ibm.com/es-es/topics/api [Último acceso 31 de enero de 2024].

[50] *Desarrollando una API con Express desde cero - Midudev | YouTube*, Disponible en: https://www.youtube.com/watch?v=YmZE1HXjpd4 [Último acceso 31 de enero de 2024].

[51] *Crear una cuenta de Google - Ayuda de Cuenta de Google*, Disponible en: https://support.google.com/accounts/answer/27441?hl=es [Último acceso 31 de enero de 2024].

[52] *Firebase Console*, Disponible en: https://console.firebase.google.com/ [Último acceso 31 de enero de 2024].

Apéndice A

Pliego de condiciones

A.1 Introducción

En este apartado se van a evaluar las condiciones necesarias para poder desarrollar el entorno que se ha presentado y descrito en los apartados anteriores.

A.2 Requisitos de hardware

A.2.1 Requisitos mínimos

- Un PC con Sistema operativo Windows, macOS, Linux o ChromeOS de 64 bits.
- Una mínimo de 16 GB de RAM.
- Un mínimo de 16 GB de espacio en disco.

A.2.2 Requisitos de hardware recomendados

Estos requisitos son necesarios para que la experiencia con los archivos audiovisuales, emuladores de dispositivos Android y el *debugger* de Android Studio, sea lo mas agradable posible.

- Procesador de 64 bits con 4 Cores o más.
- Memoria RAM de 32 GB.
- Tarjeta Gráfica de nivel superior a una NVIDIA GeForce 970 2GB.
- *Tablet* que posea el sistema operativo Android, para tener *feedback* visual que no solo provenga de un emulador.

A.3 Requisitos de software

- Sistema operativo Windows 10.
- Sistema operativo MacOS Ventura 13.0.

- Sistema operativo Android 11.
- Andorid Studio.
- Docker.
- Nodejs.
- Un SGBD, siempre que sea para bases de datos relacionales.
- Un RDBMS.
- Figma.

Apéndice B

Presupuesto

En este capítulo se incluye una estimación del coste total para la realización del proyecto.

B.1 Recursos Hardware

Estos costes son los referentes a las herramientas que se han utilizado para el desarrollo del proyecto.

- Ordenador y periféricos
- *Tablet* Android

Descripción	Cantidad	Precio por Unidad	Subtotal
Ordenador y Periféricos	1	1.099,99 €	1.099,99 €
Tablet Android	1	229,99 €	229,99 €
TOTAL			1.329,98 €

Tabla B.1: Costes Hardware

B.2 Recursos Software

El coste software incluye los gastos en licencias, se incluirá la licencia de Windows y la el pago mensual de la licencia de Figma Professional.

- Windows 11
- Figma

Descripción	Cantidad	Precio por Unidad	Subtotal
Windows 11	1	130 €	130,00 €
Figma Professional	15 meses	12,00 €/mes	180,00 €
TOTAL			310,00 €

Tabla B.2: Costes Software

B.3 Coste de la Mano de Obra

En este apartado se especifican los gastos del personal para el desarrollo de este proyecto. Estas son las tareas que se van a llevar a cabo:

- Análisis de la idea.
- Diseño de la interfaz.
- Desarrollo del código.
- Pruebas.
- Documentación.

Todas las tareas se reparten entre dos perfiles profesionales, que cada uno cuenta con un coste diferente. El tiempo empleado en realizar el proyecto es de unos quince meses.

Descripción	Horas	Precio por Hora	Subtotal
Analista	1.500	75,00 €	112.500,00 €
Programador	2.100	65,00 €	136.500,00 €
TOTAL			249.000,00 €

Tabla B.3: Coste de la Mano de Obra

B.4 Presupuesto Total

A continuación, se realiza la suma de todos los conceptos del presupuesto tenidos en cuenta para el desarrollo del proyecto. Incluyendo gastos variables que se detallan a continuación y cuya cuantía se muestra en la tabla.

- Costes Extras: Se corresponden a gastos "imprevistos".
- Gastos Generales: En estos gastos se incluyen los gastos de desplazamiento, gastos de oficina, etc.
- IVA: Impuesto sobre el valor añadido, establecido en España en un 21 %.

Descripción	Coste Total
Costes Hardware	1.329,00 €
Costes Software	310,00 €
Coste de la Mano de Obra	249.000,00 €
Costes Extras	650,00 €
Gastos Generales	2.300,00 €
IVA (21 %)	11.253,69 €
TOTAL	264.842,69 €

Tabla B.4: Coste total

Apéndice C

Manual de Usuario de Athena Escenas

C.1 Introducción

Athena Creaciones es una aplicación que se ejecuta en *tablets* Android, diseñada para que niños y niñas con Trastorno del Espectro Autista (TEA) puedan reproducir escenas visuales personalizadas para ellos.

C.2 Funcionamiento

Una vez que tenemos la aplicación en nuestro dispositivo, al pulsar en el icono de esta, el cual se puede ver en la figura C.1, se iniciará su funcionamiento.

Figura C.1: Icono de la aplicación Athena Escenas

C.2.1 Inicio de Sesión en la aplicación

Abierta la aplicación, aparecerá en el dispositivo la pantalla de inicio de sesión de la aplicación. Como se puede ver a continuación en la figura C.2, en la parte izquierda de la pantalla se puede acceder a los ajustes de conexión de la aplicación, pulsando el botón con el mismo nombre; o, introduciendo unas credenciales correctas y pulsando el boton “Iniciar Sesión” a la derecha de la pantalla, se puede iniciar sesión en la aplicación y esta se redirigirá a la pantalla de selección del alumno que usará la aplicación.

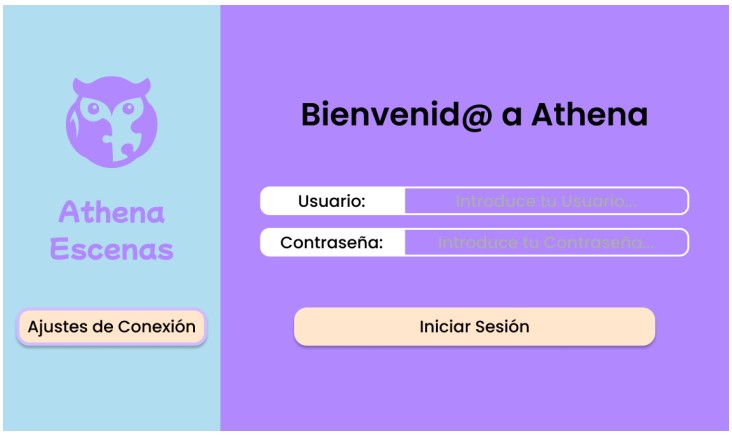

Figura C.2: Pantalla de inicio de sesión de Athena Escenas

C.2.2 Ajustes de Conexión de la aplicación

En la pantalla de los ajustes de conexión de la aplicación, como podemos ver en la figura C.3, se encuentra la herramienta de la aplicación que permite comprobar si la conexión con el servidor local, es correcta. Para hacer uso de esa funcionalidad, hay que introducir la dirección IP del servidor local en el hueco que se indica, y, a continuación, pulsar el botón "Probar Conexión".

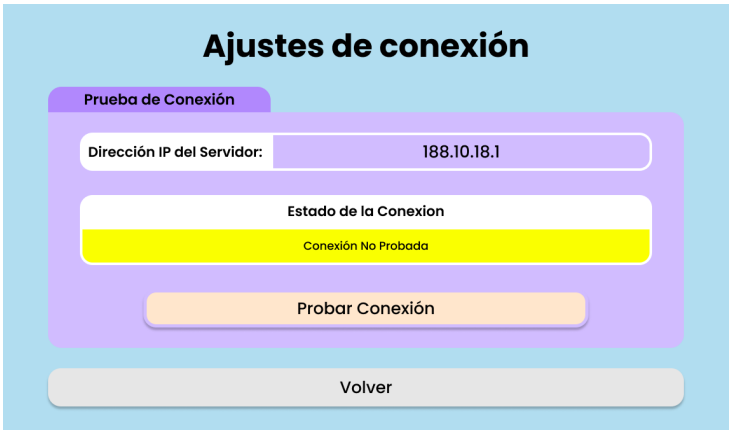

Figura C.3: Pantalla de ajustes de conexión de Athena Escenas

En el caso que la conexión sea correcta, el recuadro de "Estado de la Conexión" cambiará su color a verde y aparecerá el texto "Conexión Correcta", como podemos ver en la figura C.4.

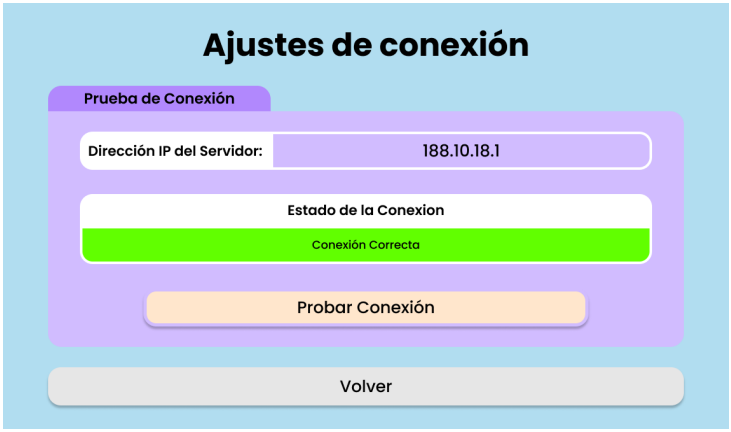

Figura C.4: Pantalla de ajustes de conexión de Athena Escenas, una vez que la prueba de conexión ha resultado correcta

En su defecto, si la conexión sea incorrecta, el recuadro de "Estado de la Conexión" cambiará su color a rojo y aparecerá el texto "Conexión Fallida", como podemos ver en la figura C.5.

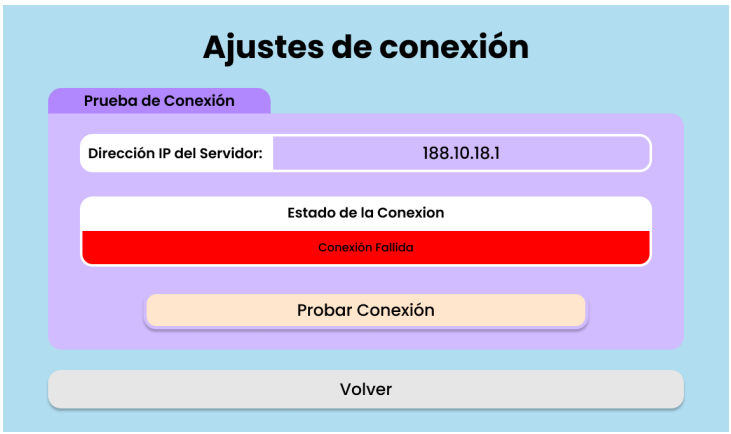

Figura C.5: Pantalla de ajustes de conexión de Athena Escenas, una vez que la prueba de conexión ha resultado fallida

Además, si se pulsa el botón "Volver", el cual aparece en la parte inferior de la pantalla, la aplicación se redirigirá a la pantalla de inicio de sesión de esta.

C.2.3 Selección del Alumno que usará la aplicación

Después de iniciar sesión en la aplicación con unas credenciales validas, se llega a la pantalla de la galería de alumnos de la aplicación, la cual se puede observar en la figura C.6.

Figura C.6: Pantalla de selección de alumno de Athena Escenas

En esta pantalla se debe elegir al alumno o a la alumna que utilizará la aplicación, ya que si se pulsa en la "tarjeta de la galería" que contiene su imagen y nombre, la aplicación se redirigirá hacia la pantalla de selección de escena, para que se seleccione la escena asignada a este alumno que se desea reproducir.

Además, en la parte de abajo aparecen tanto un botón de "Cerrar Sesión", el cual permite volver a la pantalla de inicio de sesión de la aplicación; como un "buscador de alumnos", el cual nos permite buscar al alumno que se quiere que utilice la aplicación.

C.2.4 Selección de la Escena que se reproducirá en la aplicación

Tras haberse elegido un alumno, se llega a la pantalla de selección de la escena que se reproducirá en la aplicación. Como podemos ver en la figura C.7, en esta pantalla aparecen todas las escenas que el alumno tiene asignadas.

Figura C.7: Pantalla de selección de escena de Athena Escenas

En esta pantalla, la cual ya debe utilizar un niño o niña con TEA, si se pulsa en una "tarjeta" de la galeria, la aplicación se redirige a la pantalla de reproducción de escenas para reproducir la escena elegida en esta galería.

Además, en esta pantalla se puede hacer uso del "buscador de escenas" que aparece en la parte inferior de la pantalla, para poder encontrar la escena deseada.

Como dato importante de esta pantalla, hay que indicar que no existe botón de “vuelta a atrás” para que los alumnos no salgan de ella de una manera sencilla y solo se centren en las escenas que se pueden reproducir.

C.2.5 Reproducción de la Escena seleccionada

Por último, una vez que se ha elegido una escena, se llega a la pantalla de reproducción de la escena. Esta pantalla realiza la funcionalidad principal de la aplicación y también del entorno: reproducir escenas visuales.

Como podemos ver en la figura C.8, esta pantalla esta compuesta principalmente por una imagen central. Esta imagen, al ser pulsada, hace que se inicie la reproducción de la escena, cuya funcionalidad se verá en el texto de debajo de esta imagen central.

Figura C.8: Pantalla de reproducción de escena de Athena Escenas

Además, las imágenes que aparecen en la parte superior de la pantalla sirven para que el alumno que esta reproduciendo la escena pueda navegar entre las distintas escenas que tiene asignadas, sin necesidad de volver a la pantalla de selección de escena.

Apéndice D

Manual de Usuario de Athena Creaciones

D.1 Introducción

Athena Creaciones es una aplicación que se ejecuta en *tablets* Android, diseñada para proporcionar herramientas de creación, edición y gestión tanto las escenas visuales y los distintos elementos del entorno Athena.

D.2 Funcionamiento

Una vez que tenemos la aplicación en nuestro dispositivo, al pulsar en el icono de esta, el cual se puede ver en la figura D.1, se iniciará su funcionamiento.

Figura D.1: Icono de la aplicación Athena Creaciones

D.2.1 Inicio de Sesión en la aplicación

Abierta la aplicación, aparecerá en el dispositivo la pantalla de inicio de sesión de la aplicación. Como se puede ver a continuación en la figura D.2, en la parte izquierda de la pantalla se puede acceder a los ajustes de conexión de la aplicación, pulsando el botón con el mismo nombre; o, introduciendo unas credenciales correctas y pulsando el boton “Iniciar Sesión” a la derecha de la pantalla, se puede iniciar sesión en la aplicación y esta se redirigirá a la pantalla de selección de la sección que usará la aplicación.

Figura D.2: Pantalla de inicio de sesión de Athena Creaciones

D.2.2 Ajustes de Conexión de la aplicación

En la pantalla de los ajustes de conexión de la aplicación, como podemos ver en la figura D.3, se encuentra la herramienta de la aplicación que permite comprobar si la conexión con el servidor local, es correcta. Para hacer uso de esa funcionalidad, hay que introducir la dirección IP del servidor local en el hueco que se indica, y, a continuación, pulsar el botón "Probar Conexión".

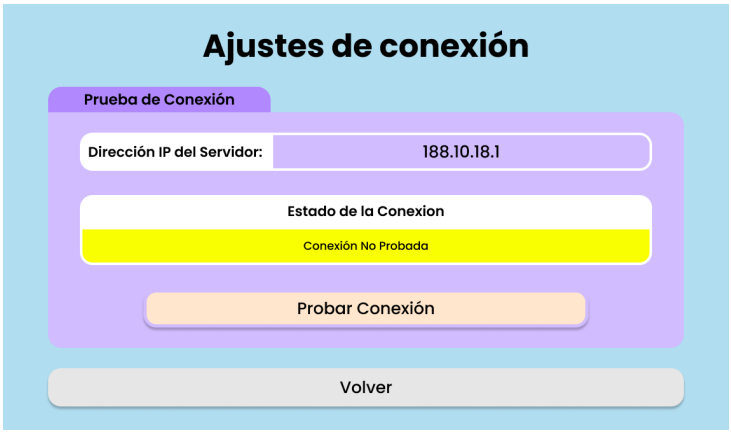

Figura D.3: Pantalla de ajustes de conexión de Athena Creaciones

En el caso que la conexión sea correcta, el recuadro de "Estado de la Conexión" cambiará su color a verde y aparecerá el texto "Conexión Correcta", como podemos ver en la figura D.4.

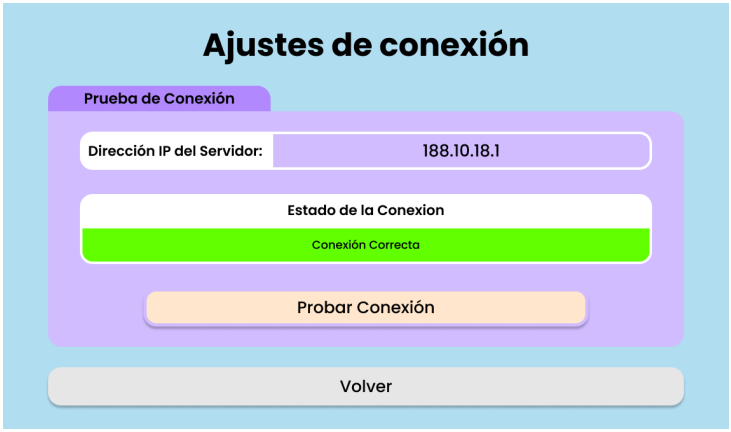

Figura D.4: Pantalla de ajustes de conexión de Athena Escenas, una vez que la prueba de conexión ha resultado correcta

En su defecto, si la conexión sea incorrecta, el recuadro de "Estado de la Conexión" cambiará su color a rojo y aparecerá el texto "Conexión Fallida", como podemos ver en la figura D.5.

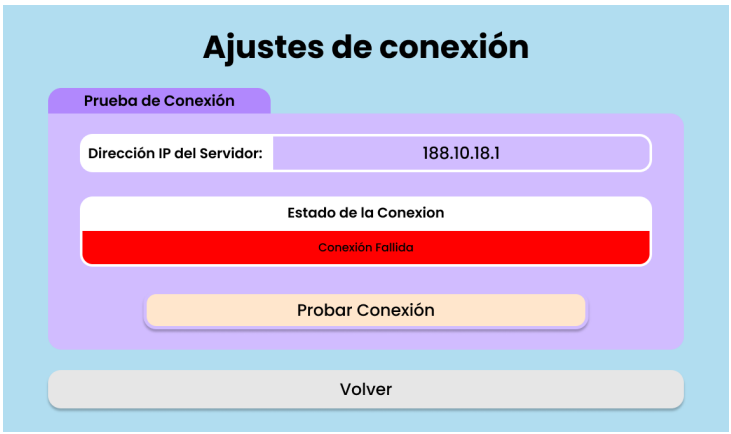

Figura D.5: Pantalla de ajustes de conexión de Athena Escenas, una vez que la prueba de conexión ha resultado fallida

Además, si se pulsa el botón "Volver", el cual aparece en la parte inferior de la pantalla, la aplicación se redirigirá a la pantalla de inicio de sesión de esta.

D.2.3 Selección de la sección que contiene las herramientas de gestión

Después de iniciar sesión en la aplicación con unas credenciales validas, se llega a la pantalla de la selección de sección de la aplicación, la cual se puede observar en la figura D.6.

Figura D.6: Pantalla de selección de sección de Athena Escenas

En esta pantalla se elegir la sección de la aplicación a la que se quiere acceder. Existen tres tipos de secciones:

- Sección de edición de usuarios: Sección que agrupa todas las herramientas de creación, gestión y edición de usuarios. Si se pulsa el botón de esta sección, la aplicación se redirigirá a la pantalla de muestra de los usuarios creados en el entorno.
- Sección de edición de escenas: Sección que agrupa todas las herramientas de creación, gestión y edición de escenas. Si se pulsa el botón de esta sección, la aplicación se redirigirá a la pantalla de muestra de las escenas creadas en el entorno.
- Sección de edición de alumnos: Sección que agrupa todas las herramientas de creación, gestión y edición de alumnos. Si se pulsa el botón de esta sección, la aplicación se redirigirá a la pantalla de muestra de los alumnos creados en el entorno.

Además, en la parte de abajo aparece un botón de "Cerrar Sesión", el cual permite volver a la pantalla de inicio de sesión de la aplicación.

D.2.4 Pantalla de muestra de todos los usuarios creados en el entorno

En esta pantalla, como podemos ver en la figura D.7, aparece una galería con todos los usuarios creados en el entorno.

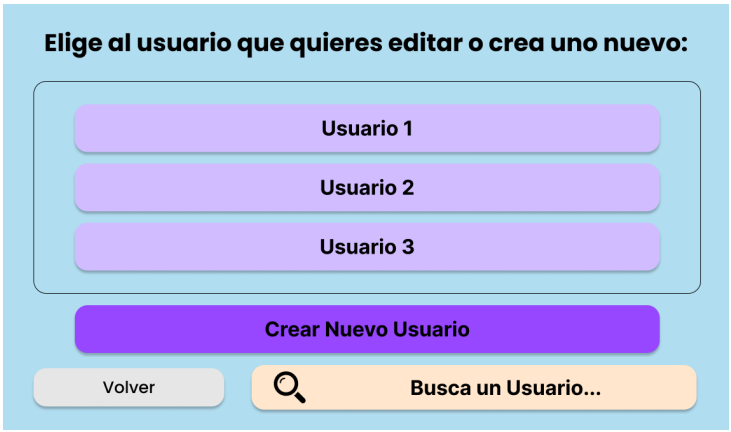

Figura D.7: Pantalla de selección de usuario de Athena Creaciones

Si se pulsa en el nombre de cualquier usuario ya creado, la aplicación se redirigirá al menú de edición de usuario para poder editar al usuario seleccionado.

Además, en la parte de abajo aparecen un botón "Crear Nuevo Usuario", el cual nos permite acceder al menú de creación de usuario; un botón de "Volver", el cual permite volver a la pantalla de selección de sección de la aplicación; y, por último, un "buscador de usuarios", el cual nos permite buscar al usuario que se desea editar.

D.2.5 Menú de creación de usuario

En la pantalla de este menú, como se puede ver en la figura D.8, se puede crear un nuevo usuario en el entorno mediante la introducción de los datos requeridos en la parte derecha de esta.

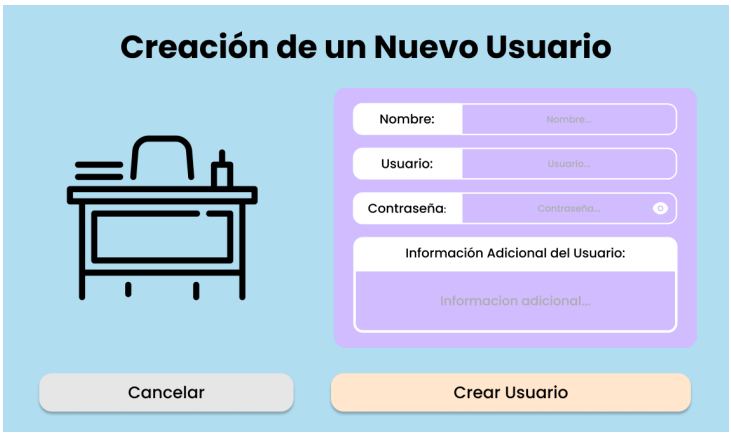

Figura D.8: Pantalla del menú de creación de usuario de Athena Creaciones

Una vez que se han introducido los datos obligatorios, Nombre, Usuario y Contraseña, al pulsar el botón "Crear Usuario" se procederá a la creación del nuevo usuario y, si esta resulta exitosa, la aplicación se redirigirá al menú de edición de usuario para gestionar al usuario recién creado.

Además, esta pantalla permite volver a la pantalla de muestra de los usuarios creados en el entorno si se presiona el botón "Cancelar".

D.2.6 Menú de edición de usuario

La pantalla de este menú, como podemos ver en la figura D.9, sirve como nexo de unión entre las distintas herramientas disponibles en la aplicación, para la gestión de los usuarios creados.

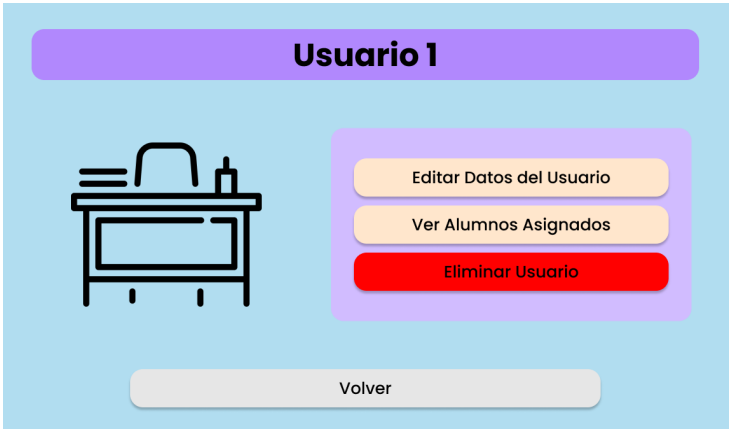

Figura D.9: Pantalla del menú de edición de usuario de Athena Creaciones

Los botones de la parte derecha de la pantalla de este menú nos permiten redirigir la aplicación al menú de edición de datos del usuario, al menú de alumnos asignados al usuario y al menú de eliminación del usuario.

Además, esta pantalla permite volver a la pantalla de muestra de los usuarios creados en el entorno si se presiona el botón "Volver".

D.2.7 Menú de edición de datos del usuario

En este menú, cuya pantalla podemos ver en la figura D.10, nos permite editar los datos de un usuario ya creado.

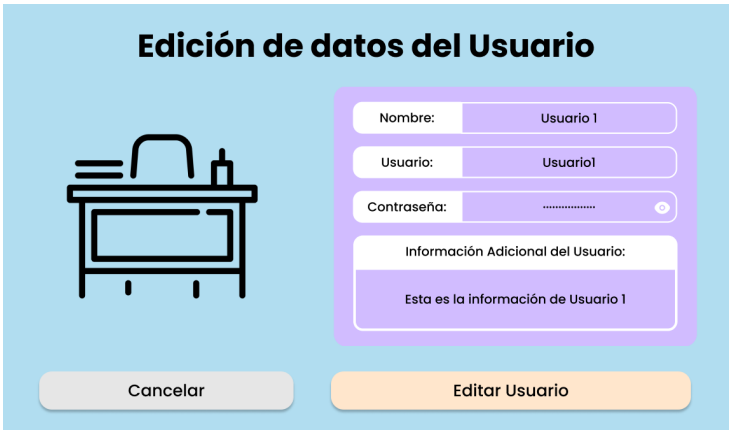

Figura D.10: Pantalla del menú de edición de datos de usuario de Athena Creaciones

Una vez que se tienen los datos editados y los campos obligatorios (nombre, usuario y contraseña) están rellenos, al pulsar el botón "Editar Usuario" se reemplazan los datos del usuario por los introducidos.

Además, esta pantalla permite volver al menú de edición de usuario si se presiona el botón "Cancelar".

D.2.8 Menú de alumnos asignados al usuario

En este menú, cuya pantalla podemos ver en la figura D.11, nos permite ver una lista con los alumnos que el usuario tiene asignados. Si se pulsa alguno de los alumnos que aparecen en la lista, la aplicación se redirige al menú de edición del alumno seleccionado.

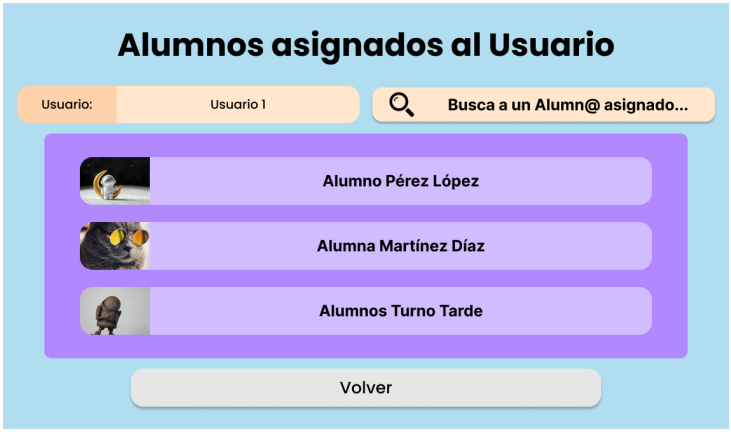

Figura D.11: Pantalla del menú de alumnos asignados al usuario de Athena Creaciones

Además, esta pantalla permite volver al menú de edición de usuario si se presiona el botón "Volver".

D.2.9 Menú de eliminación de usuario

En este menú, cuya pantalla podemos ver en la figura D.12, nos permite confirmar la eliminación del usuario seleccionado del entorno. Si se confirma la eliminación pulsado el botón "Eliminar" y esta es correcta, la aplicación se redirige a la pantalla de muestra de los usuarios creados en el entorno.

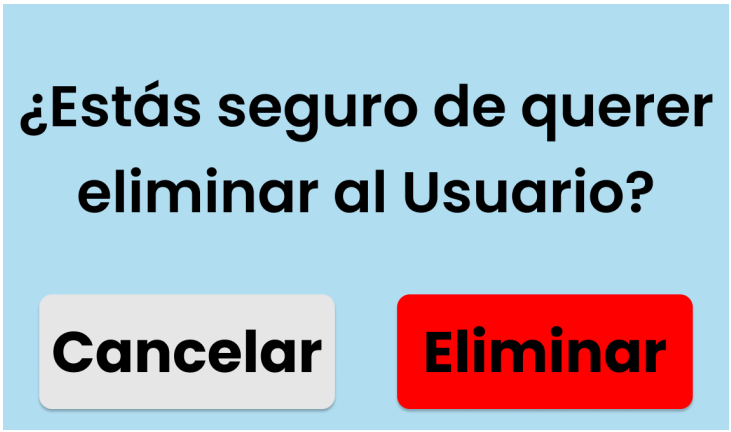

Figura D.12: Pantalla del menú de eliminación de usuario de Athena Creaciones

En caso de que se pulse el botón "Cancelar", esta pantalla permite volver al menú de edición de usuario, sin realizar ningún cambio.

D.2.10 Pantalla de muestra de todos las escenas creadas en el entorno

En esta pantalla, como podemos ver en la figura D.13, aparece una galería con todos las escenas creadas.

Figura D.13: Pantalla de selección de escena de Athena Creaciones

Si se pulsa en la "tarjeta" que representa una escena ya creada, la aplicación se redirigirá al menú de edición de escena para poder editar a la escena seleccionada.

Además, en la parte de la izquierda de la pantalla, aparece otra "tarjeta" con el texto "Nueva Escena", la cual, al ser pulsada, nos permite acceder al menú de creación de escena.

Por último, en la parte inferior aparecen un botón de "Volver", el cual permite volver a la pantalla de selección de sección de la aplicación; y un "buscador de escenas", el cual nos permite buscar la escena que se desea editar.

D.2.11 Menú de creación de escena

En la pantalla de este menú, como se puede ver en la figura D.14, se puede crear una nueva escena en el entorno mediante la introducción de los datos requeridos en la parte derecha de esta y la selección de un archivo de imagen en la parte izquierda.

Figura D.14: Pantalla del menú de creación de escena de Athena Creaciones

Una vez que se han rellenado todos los huecos de introducción de datos en la mencionada parte derecha y se ha seleccionado una archivo de imagen en la parte izquierda, al pulsar el botón "Crear Escena" se procederá a la creación de la nueva escena y, si esta resulta exitosa, la aplicación se redirigirá al menú de edición de escena para gestionar la escena recién creada.

Además, esta pantalla permite volver a la pantalla de muestra de las escenas creadas en el entorno si se presiona el botón "Cancelar".

D.2.12 Menú de edición de escena

La pantalla de este menú, como podemos ver en la figura D.15, sirve como nexo de unión entre las distintas herramientas disponibles en la aplicación, para la gestión de las escenas creadas.

Figura D.15: Pantalla del menú de edición de una escena de Athena Creaciones

Los botones de la parte derecha de la pantalla de este menú nos permiten redirigir la aplicación al menú de previsualizacion de la escena, al menú de modificación pictolingüística de la escena, al menú de grabación de audio de la escena, al menú de sincronización del audio y del texto de la escena y al menú de eliminación de la escena.

Además, esta pantalla permite volver a la pantalla de muestra de las escenas creadas en el entorno si se presiona el botón "Volver".

D.2.13 Menú de previsualización de la escena

En la pantalla de este menú, como podemos ver en la figura D.7, aparece una simulación de la pantalla de reproducción de una escena en Athena Escenas.

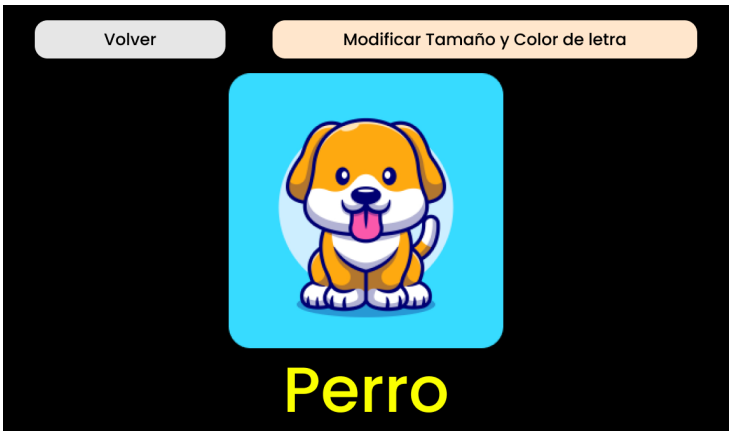

Figura D.16: Menú de previsualización de una escena de Athena Creaciones

Su funcionamiento es exactamente idéntico a la pantalla de reproducción de escena de Athena Escenas, pero en vez de permitir la navegación entre escenas, este menú posee dos botones en la parte superior. El botón de la parte izquierda, "Volver", redirige la aplicación de vuelta al menú de edición de escena;

mientras que el botón de la derecha, “Modificar Tamaño y Color de letra”, redirige la aplicación al menú de edición tipográfica de la escena.

D.2.14 Menú de edición tipográfica de una escena

En la pantalla de este menú, como podemos ver en la figura D.17, se puede probar y modificar la tipografía de una escena cambiando el valor del tamaño de letra y el código de color del texto que aparece en la reproducción de esta.

Figura D.17: Menú de edición tipográfica de una escena de Athena Creaciones

Para poder cambiar tanto el tamaño, como el color, se tiene que introducir en los campos correspondientes, los valores deseados. Si se pulsa en el botón “Probar Letra”, se puede ver una prueba de la tipografía introducida. En el caso de que se quieran guardar los datos introducidos, se debe pulsar el botón “Guardar Cambios”, situado en la parte inferior derecha de la pantalla.

Por parte del botón de la parte izquierda, “Volver”, este botón redirige la aplicación de vuelta al menú de edición de escena.

D.2.15 Menú de edición picto-lingüística de una escena

En la pantalla de este menú, como podemos ver en la figura D.18, se pueden modificar los datos de imagen y lingüísticos de una escena.

Figura D.18: Menú de modificación picto-lingüística de una escena de Athena Creaciones

Para poder modificar el archivo de imagen de la escena, se tiene que pulsar la propia imagen de la escena y elegir un nuevo archivo de imagen del sistema de archivos de la *tablet* en la que se ejecuta la aplicación.

Para guardar todos los cambios, una vez que se han introducido todas las modificaciones deseadas y hay un archivo de imagen seleccionado, se tiene que pulsar el botón "Guardar Cambios".

Por parte del botón de la parte izquierda, "Volver", este botón redirige la aplicación de vuelta al menú de edición de escena.

D.2.16 Menú de grabación de audio de una escena

En la pantalla de este menú, como se puede ver en la figura D.19, se puede tanto grabar o adjuntar un archivo de audio del sistema de archivos de la *tablet* por primera vez a una escena, como modificar el audio en el caso de que la escena ya tenga uno vinculado. Además, se puede probar el audio antes de guardar los cambios.

Figura D.19: Pantalla del menú de grabación de audio de Athena Creaciones

En la parte inferior, se pueden encontrar los controles de grabación, siendo el botón de la izquierda el botón de grabación o de selección de archivo de audio; el botón del medio, el botón de parada de grabación; y el botón de la derecha, el botón de eliminación del audio vinculado.

En la parte superior, se encuentran los controles de grabación, botón de reproducción a la izquierda y botón de detención a la derecha, los cuales permiten comprobar el funcionamiento del audio antes de guardar las modificaciones.

Una vez que la modificación de audio es la deseada, al pulsar el botón "Guardar Cambios" se procederá a la vinculación del audio con la escena, si esta resulta exitosa, la aplicación se redirigirá al menú de edición de escena.

Además, esta pantalla permite volver a la pantalla de muestra de las escenas creadas en el entorno, sin guardar los cambios realizados, en el caso que se presione el botón "Cancelar".

D.2.17 Menú de sincronización del audio y del texto de una escena

En la pantalla de este menú, como se puede ver en la figura D.20, se puede sincronizar el audio y el texto de una escena mediante la introducción de valores de tiempo que serán asignados a los periodos de reproducción de la escena.

Figura D.20: Pantalla del menú de sincronización del audio y del texto de una escena de Athena Creaciones

Al introducir valores numéricos, que representen las duraciones de los periodos de reproducción de la escena, se puede calibrar la sincronización entre el audio y el texto de la escena seleccionada, para que esta sea lo mas precisa posible.

En la parte izquierda, utilizando los controles de reproducción (botón de la derecha, reproducción y botón de la izquierda, detención), se puede reproducir una prueba de la reproducción, con los valores que se han introducido en la parte derecha de la pantalla.

Si la prueba verifica que los valores son los deseados, al pulsar el botón "Guardar Cambios", situado en la parte inferior derecha de la pantalla, se guardan los valores introducidos. Y, en el caso de que el guardado sea exitoso, la aplicación se redirige a la pantalla de muestra de las escenas creadas en el entorno.

Además, esta pantalla permite volver a la pantalla de muestra de las escenas creadas en el entorno, sin guardar los valores introducidos, si se presiona el botón "Cancelar".

D.2.18 Menú de eliminación de escena

En este menú, cuya pantalla podemos ver en la figura D.21, nos permite confirmar la eliminación de la escena seleccionada del entorno. Si se confirma la eliminación pulsado el botón "Eliminar" y esta es correcta, la aplicación se redirige a la pantalla de muestra de las escenas creadas en el entorno.

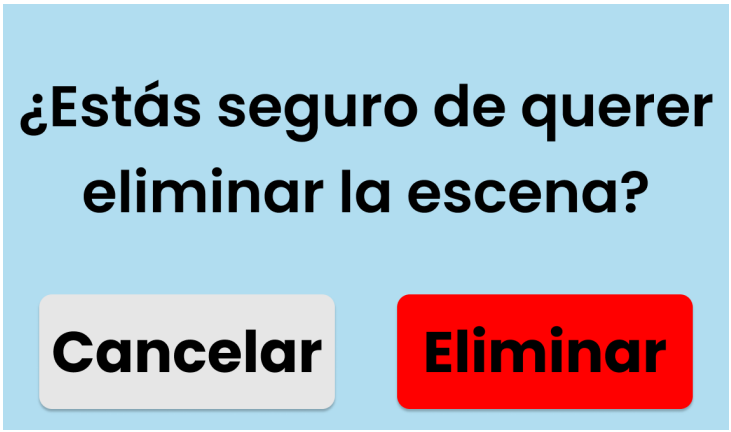

Figura D.21: Pantalla del menú de eliminación de escena de Athena Creaciones

En caso de que se pulse el botón “Cancelar”, esta pantalla permite volver al menú de edición de escena, sin realizar ningún cambio.

D.2.19 Pantalla de muestra de todos los alumnos creados en el entorno

En esta pantalla, como podemos ver en la figura D.22, aparece una galería con todos los alumnos creados en el entorno.

Figura D.22: Pantalla de selección de alumno de Athena Creaciones

Si se pulsa en la “tarjeta” que representa un alumno ya creado, la aplicación se redirigirá al menú de edición de alumno para poder editar al alumno seleccionado.

Además, en la parte de la izquierda de la pantalla, aparece otra “tarjeta” con el texto “Nuevo Alumno”, la cual, al ser pulsada, nos permite acceder al menú de creación de alumno.

Por último, en la parte inferior aparecen un botón de “Volver”, el cual permite volver a la pantalla de selección de sección de la aplicación; y un “buscador de alumnos”, el cual nos permite buscar el alumno que se desea editar.

D.2.20 Menú de creación de alumno

En la pantalla de este menú, como se puede ver en la figura D.23, se puede crear un nuevo alumno en el entorno mediante la introducción de los datos requeridos en la parte derecha de esta y la selección de un archivo de imagen en la parte izquierda.

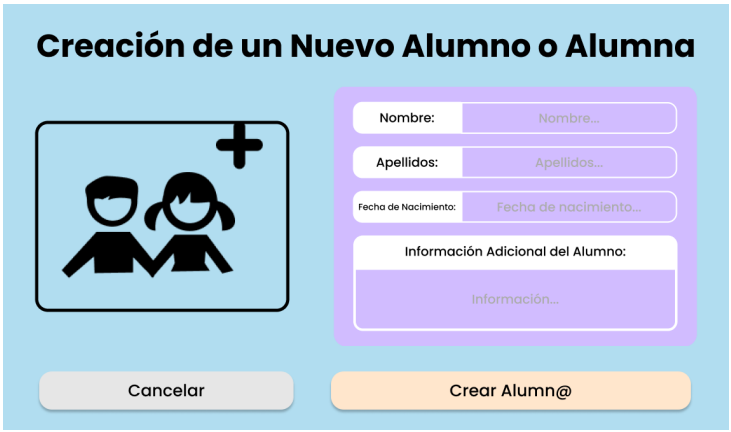

Figura D.23: Pantalla del menú de creación de alumno de Athena Creaciones

Una vez que se han rellenado los huecos de introducción de datos que son obligatorios (nombre y apellidos) en la mencionada parte derecha y se ha seleccionado una archivo de imagen en la parte izquierda, al pulsar el botón "Crear Alumno" se procederá a la creación del nuevo alumno y, si esta resulta exitosa, la aplicación se redirigirá al menú de edición de alumno para gestionar el alumno recién creado.

Además, esta pantalla permite volver a la pantalla de muestra de los alumnos creados en el entorno si se presiona el botón "Cancelar".

D.2.21 Menú de edición de alumno

La pantalla de este menú, como podemos ver en la figura D.24, sirve como nexo de unión entre las distintas herramientas disponibles en la aplicación, para la gestión de los alumnos creados.

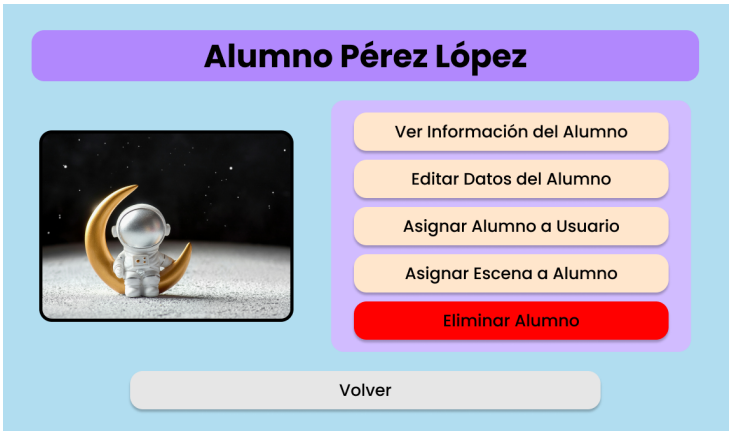

Figura D.24: Pantalla del menú de edición de un Alumno de Athena Creaciones

Los botones de la parte derecha de la pantalla de este menú nos permiten redirigir la aplicación al menú de información del alumno, al menú de edición de los datos del alumno, al menú de asignación del alumno a usuarios, al menú de asignacion de escenas al alumno y al menú de eliminación del alumno.

Además, esta pantalla permite volver a la pantalla de muestra de los alumnos creados en el entorno si se presiona el botón "Volver".

D.2.22 Menú de visualización de datos del alumno

En este menú, cuya pantalla podemos ver en la figura D.25, nos permite visualizar la información de un alumno ya creado.

Figura D.25: Pantalla del menú de visualización de los datos de un alumno de Athena Creaciones

En esta pantalla solo se muestra la información almacenada que esta vinculada al alumno seleccionado, por si hay que tener en cuenta parte de la información adicional que aparece en su registro.

Además, este menú permite volver al menú de edición de alumno si se presiona el botón “Volver”.

D.2.23 Menú de edición de datos del alumno

En este menú, cuya pantalla podemos ver en la figura D.21, nos permite editar los datos almacenados en el entorno acerca del alumno seleccionado.

Figura D.26: Pantalla del menú de edición de los datos de un alumno de Athena Creaciones

Para poder modificar el archivo de imagen del alumno, se tiene que pulsar la propia imagen del alumno y elegir un nuevo archivo de imagen del sistema de archivos de la *tablet* en la que se ejecuta la aplicación.

Para guardar todos los cambios, una vez que se han introducido todas las modificaciones deseadas, los campos nombre y apellidos no están vacíos y hay un archivo de imagen seleccionado, se tiene que pulsar el botón “Editar Alumn@”.

En caso de que se pulse el botón “Cancelar”, este menú permite volver al menú de edición de alumno, sin realizar ningún cambio.

D.2.24 Menú de asignación del alumno a usuarios

En la pantalla de este menú, como podemos ver en la figura D.27, se puede asignar el alumno seleccionado a los distintos usuarios que se han creado en el entorno, de manera que cuando estos inicien sesión en Athena Escenas, puedan seleccionar al alumno para que reproduzca escenas.

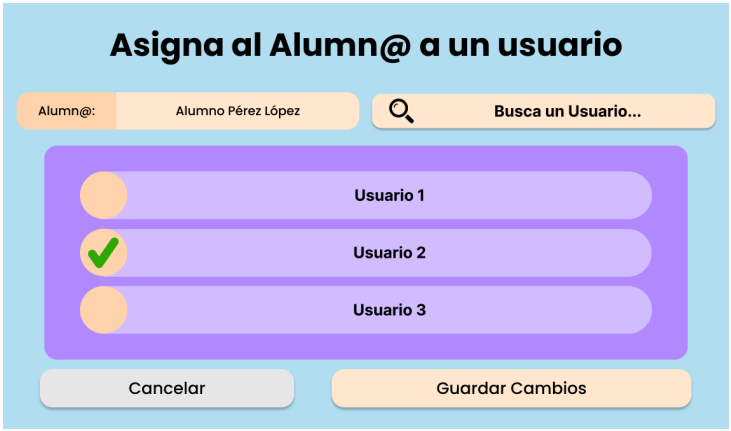

Figura D.27: Menú de asignación de un alumno a usuarios de Athena Creaciones

En la parte central, se puede decidir si se asigna o se revierte la asignación del alumno al usuario cuyo nombre aparece en el elemento de la lista. Para asignar al usuario el alumno, se ha de pulsar el elemento donde aparece el nombre del usuario hasta que un *tick* verde aparezca a la izquierda de este elemento. En el caso de querer revertir la asignación, se ha de pulsar el elemento hasta que este *tick* no aparezca.

Una vez que el alumno esté asignado a todos los usuarios deseados, se ha de pulsar el botón "Guardar Cambios".

Por el contrario, al pulsar el botón "Cancelar", el menú permite volver al menú de edición de alumno, sin realizar ningún cambio.

D.2.25 Menú de asignación de escenas al alumno

En la pantalla de este menú, como podemos ver en la figura D.28, se puede asignar al alumno seleccionado, las distintas escenas que se han creado en el entorno, de manera que cuando los alumnos sean seleccionados en Athena Escenas, puedan seleccionar la escena asignada para que se reproduzca.

Figura D.28: Menú de asignación de un alumno a usuarios de Athena Creaciones

En la parte central, se puede decidir si se asigna o se revierte la asignación de la escena que aparece en la lista, al alumno. Para asignar la escena al alumno, se ha de pulsar el elemento donde aparece la escena deseada hasta que un *tick* verde aparezca a la izquierda de este elemento. En el caso de querer revertir la asignación, se ha de pulsar el elemento hasta que este *tick* no aparezca.

Una vez que el alumno tenga asignadas las escenas deseadas, se ha de pulsar el botón "Guardar Cambios", para que estos queden confirmados.

Por el contrario, al pulsar el botón "Cancelar", el menú permite volver al menú de edición de alumno, sin realizar ningún cambio.

D.2.26 Menú de eliminación de alumno

En este menú, cuya pantalla podemos ver en la figura D.29, nos permite confirmar la eliminación del alumno seleccionado del entorno. Si se confirma la eliminación pulsado el botón "Eliminar" y esta es correcta, la aplicación se redirige a la pantalla de muestra de los alumnos creados en el entorno.

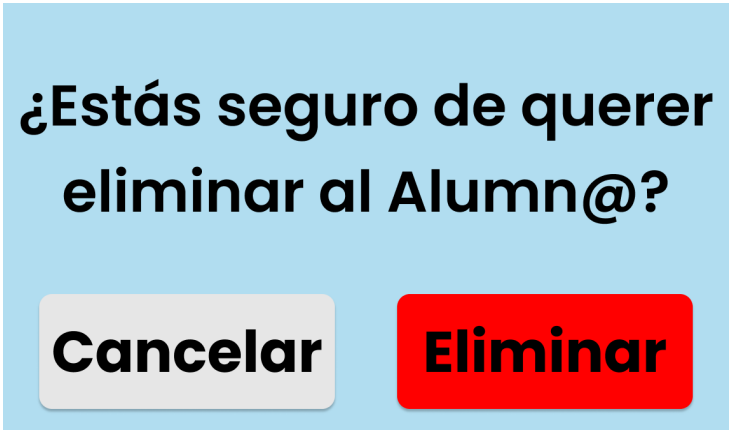

Figura D.29: Pantalla del menú de eliminación de alumno de Athena Creaciones

En caso de que se pulse el botón "Cancelar", esta pantalla permite volver al menú de edición de alumno, sin realizar ningún cambio.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá