UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

Profesor Patrocinante:

Ph. D. Pamela Guevara A.

Informe de Memoria de Título
para optar al título de:

Ingeniero Civil Biomédico

"Desarrollo de Aplicación Android para Estimulación del Lenguaje en Edad Preescolar"

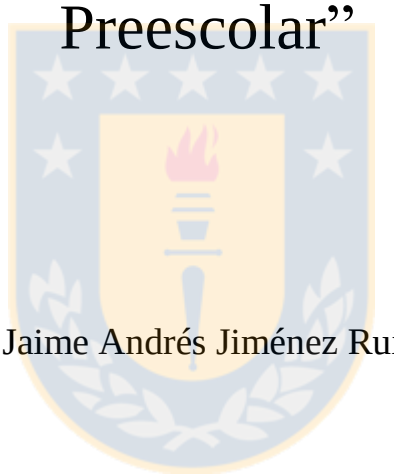
Concepción, Agosto 2017

Jaime Andrés Jiménez Ruiz

UNIVERSIDAD DE CONCEPCIÓN
Facultad de Ingeniería
Departamento de Ingeniería Eléctrica

Profesor Patrocinante:
Ph. D. Pamela Guevara A.

“Desarrollo de Aplicación Android para Estimulación del Lenguaje en Edad Preescolar”



Jaime Andrés Jiménez Ruiz

Informe de Memoria de Título
para optar al Título de

Ingeniero Civil Biomédico

Agosto 2017

Resumen

Este trabajo se enmarca dentro del proyecto FONDEF IDEA ID 16I10210, denominado “Sistema Interactivo de Promoción del Lenguaje y la Comunicación (SIPLYC)”, que busca desarrollar una aplicación móvil para la estimulación lingüística de niños en edad pre-escolar. Para esto, se requiere del desarrollo de dos aplicaciones en Android; una principal de estimulación (o juego) y otra secundaria de control (o de Tutor).

El presente trabajo corresponde al diseño e implementación de la aplicación secundaria del proyecto, que permite la evaluación del habla, generación de recreos y control de pausas a través de comunicación Bluetooth con la aplicación principal. Además, permite la visualización de las estadísticas e historial de evaluación a partir de la información almacenada en una base de datos.

Se revisó bibliografía referida a los procesos cognitivos y fisio-anatómicos en el desarrollo del lenguaje de niños pequeños, además de identificar y analizar otros softwares existentes con este mismo fin o similares.

Se analizó la aplicación principal (actualmente en desarrollo) para una correcta integración, identificando sus funcionalidades, lógica de juego y comunicación Bluetooth. Para la base de datos, se optó por implementar un sistema basado en Firebase Realtime Database, debido a su capacidad de generar persistencia en el disco para funcionar ante desconexión de internet y capacidad de manejar múltiples clientes sincronizados.

Se integraron las funcionalidades necesarias para el uso de Firebase en la aplicación principal, además de las configuraciones necesarias para la generación de recreos, pausas e intercambio de datos entre aplicaciones, mediante comunicación Bluetooth, sin alterar la lógica de ejercicio implementada.

Se evaluó el funcionamiento de la aplicación principal para almacenar datos y responder a las instrucciones de control de la aplicación de Tutor, instalada en diez dispositivos, y se evaluó la capacidad de esta última para leer la base de datos, para generar y visualizar las estadísticas. La lectura no presentó tiempos de carga perceptibles para el usuario y fueron entregadas en tiempo real cuando la conexión a red de internet estaba disponible. En el caso de evaluaciones generadas de forma offline, los datos fueron almacenados de forma local y posteriormente sincronizados de forma automática al recuperar la conexión, cumpliendo con las características requeridas.

“Winter is Here”

Agradecimientos

Agradezco a mi familia por su amor y apoyo incondicional.

Agradezco a los amigos que conocí durante este período y a los que se han mantenido hasta el día hoy; por esos buenos momentos donde compartir y distraerse es tan necesario.

Finalmente, agradezco a esos héroes anónimos y no tan anónimos de *Stack Overflow, Git-Hub* y otros foros, especialmente a quienes no tenían vergüenza de preguntar algo que "ya tenía respuesta", como para quienes se dieron el tiempo de idear nuevas soluciones: muchos trabajos universitarios, en conjunto al presente, fueron posibles gracias a ustedes.

¡Gracias!

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

Abreviaciones

Mayúsculas

RAE : Real Academia de la Lengua Española.
SIPLYC : Sistema Interactivo de Promoción del Lenguaje y la Comunicación.
DES : Ireland's Department of Education and Skill.
(Departamento de Educación y Habilidades de Irlanda).
NIH : National Institute of Health.
(Instituto Nacional de Salud, Estados Unidos).
NIDOCD : National Institute of Deafness and Other Communication Disorders.
(Instituto Nacional de Sordera y Otros Desórdenes de Comunicación, Estados Unidos).
TEA : Trastorno de Espectro Autista
IDE : Interactive Development Environment.
(Entorno de Desarrollo)
API : Application Programming Interface
(Interfaz de Programación de Aplicaciones)
UI : User Interface
(Interfaz de Usuario)
ID : Identificador.
LRU : Least Recently Used
(Menos Utilizado Recientemente)
UUID : Universally Unique Identifier
(Identificador Único Universal)

Minúsculas

app : aplicación (entendido como software para dispositivos móviles)
apps : aplicaciones
fMRI : functional Magnetic Resonance Imaging.
(Imagen de Resonancia Magnética Funcional)

Capítulo 1. Introducción

1.1. Introducción General

Los seres humanos nos comportamos de forma social: desde la aparición de los primeros homínidos, nuestros instintos primitivos de supervivencia nos llevaron a fortalecer nuestros lazos de cooperación con nuestros pares, formando grupos con diferentes propósitos, como cuidado de crías, caza y posteriormente de recolección. Con el avance de los tiempos, estos grupos que inicialmente eran constituidos por familiares, formaron comunidades, pueblos, ciudades, civilizaciones, castillos, e imperios, así, como paralelamente, se complejizaba su lenguaje y se desarrollaba su cultura. Mucho ha evolucionado nuestro lenguaje desde que el primer hombre primitivo se vio en la necesidad de dar un significado a los sonidos que emitía, y este se sigue transformando con el actual mundo globalizado y de alta tecnología que genera nuevos medios y formas de comunicación. Sin embargo, la conclusión es clara, *"La comunicación es una condición necesaria para la existencia del hombre y uno de los factores más importantes de su desarrollo social"* [1].

El lenguaje es nuestra forma principal de comunicación. Mientras la Real Academia de la Lengua Española (RAE) lo define como "*conjunto de sonidos articulados con que el hombre manifiesta lo que piensa o siente*" [2], desde un punto de vista lingüístico se define como una capacidad del ser humano, que le permite expresar un mensaje a través de símbolos, que pueden ser tanto orales como escritos [3][4]. Este conjunto de símbolos, bajo los cuales subyace un significado definido e integrado por una sociedad, pueblo o comunidad a través de una convención social, es lo que conocemos como Lengua, y por esto, nos referimos a los diferentes idiomas como Lengua Española, Lengua Inglesa, Lengua Italiana, entre muchas otras. [5]. El habla corresponde a la forma o acto individual en la que se hace uso de la Lengua para comunicar un mensaje [5].

El origen del lenguaje humano no se encuentra claro. ¿Cuándo apareció? Si bien hoy se sabe que el neandertal poseía laringe similar a la de los simios y un cerebro de mayor tamaño al hombre actual, se desconoce si este tenía la capacidad de "hablar" o no. ¿Cómo se originó? Algunos autores creen que éste se fue formando a partir de la imitación de sonidos de la naturaleza, o sonidos propios de dolor o auxilio, o quizás fue un invento consciente, es posible que nunca se sepa [6]. Entonces, en el tiempo actual donde el Lenguaje y la Lengua se dan por hecho ¿Cómo desarrolla el habla un niño?

Mientras algunos autores tienen una visión enfocada en el lenguaje como una capacidad innata que "crece" dentro de la cabeza del niño, otros lo atribuyen a un proceso de imitación de la Lengua a

partir de la interacción con otras personas [6]. Considerando ambos enfoques, una conclusión es clara, tanto el desarrollo biológico del niño como la estimulación del medio cumplen un rol fundamental.

¿Cómo es el medio actual dónde se desarrolla el lenguaje de los niños de hoy? Con la llegada de la era digital, en estos últimos veinte años el escenario ha cambiado radicalmente con la masificación de los smartphones y tablets con gran penetración en nuestro país. Un estudio del año 2015 realizado por la empresa de telecomunicaciones VTR indicó que un tercio de los niños chilenos de tres años *"ha tenido acceso a un celular o tablet"* [7] y a los cinco años un 40% tendría *"acceso a un celular con internet"* [7]. La Encuesta Casen 2015, por otro lado, indicó que 2 de cada 3 niños menores de 12 años poseen un celular, además de detectar un uso temprano transversal entre los diferentes grupos socioeconómicos, donde cerca de un 20% de los niños entre 5-8 años utilizaban frecuentemente y eran dueños de un celular [8]. La encuesta CHILE3D 2017, aplicada por Gfk Adimark, identificó que de los niños entre 8-14 años que se declararon con acceso a internet, las dos principales actividades asociadas eran "*ver videos de YouTube*" (63 %) y "*para jugar*" (57%) [9].

Pero, ¿se ha visto afectado el aprendizaje y el desarrollo de los niños con estos dispositivos al alcance de la mano? En realidad, no es la primera vez que exponer a los pequeños a una pantalla se pone en tela de juicio; la televisión, por décadas, ha sido blanco de muchos estudios al respecto. Los menores que pasan su tiempo frente al televisor obtendrían menor estimulación en palabras promedio/hora, que por interacción con sus padres lo que tendría una traducción directa en un menor aprendizaje del lenguaje [10]. ¿Y qué ocurre con los niños expuestos a tablets y celulares a temprana edad? Un artículo del periódico británico *The Independent* [11] se contactó con diferentes investigadores para tener una visión más general sobre este tema. Algunos investigadores indicaron una cierta correlación entre el uso de estos dispositivos y el desarrollo de hiperactividad a partir de una investigación con ratas [12] o con patrones de sueño alterados en niños entre 6 y 36 meses determinados a partir de una encuesta aplicada a 715 padres [13]. Otras investigaciones indicaron que incluso frente al uso de aplicaciones educativas de resolución de puzzles, el aprendizaje no es comparable con el logrado con experiencias en el mundo real. Sin embargo, esto no significaba que los niños no pudieran aprender con estos dispositivos, un estudio de la Universidad de Stanford [14] concluyó que dotando estos aparatos con contenido de alta calidad pueden ser una medida efectiva para apoyar en el aprendizaje de menores en familias de escasos recursos, aumentando la estimulación temprana y disminuyendo la brecha en el desarrollo del lenguaje con niños de familias más acomodadas [11].

La presente memoria de título se enmarca dentro del proyecto *"Sistema Interactivo de Promoción del Lenguaje y la Comunicación (SIPLYC)"* [15], que busca desarrollar una aplicación para el estímulo del aprendizaje del lenguaje para niños entre 2 y 3 años y medio, esto con el fin de mejorar la literacidad en la edad pre-escolar. Para futuras menciones, se entenderá literacidad de acuerdo a la definición del Ireland's Department of Education and Skills (DES), que lo define como *"...la capacidad de leer, entender y apreciar críticamente diferentes formas de comunicación, incluyendo lenguaje oral, texto escrito, medios audiovisuales y medios digitales"* (Traducción propia) [16] y que es la definición utilizada en el marco del proyecto de investigación.

Los resultados del presente trabajo son; el desarrollo de componentes de una aplicación de estimulación del lenguaje para su uso en la investigación descrita, en conjunto a una aplicación secundaria que permitirá al profesional examinador controlar las actividades, realizar evaluación del Habla y analizar la curva de aprendizaje de cada menor. Para lo anterior, se implementó una base de datos que guarda la información de uso de la aplicación por cada usuario.

1.2. Objetivos

1.2.1 Objetivo General

Desarrollar módulos para aplicación en Android, para estimulación del lenguaje en niños entre dos y tres años y medio.

1.2.2 Objetivos Específicos

(i) Estudiar versión anterior de software, desarrollada también para Android.

(ii) Diseñar e implementar base de datos para almacenar el progreso del/la niño/a e historial de evaluación.

(iii) Desarrollar componentes de la aplicación del Tutor para evaluar pruebas de vocalización y visualizar estadísticas generadas.

(iv) Integrar características desarrolladas en aplicación principal.

1.3. Alcances y Limitaciones

(i) Este trabajo se encuentra enmarcado dentro del proyecto FONDEF IDEA ID 16I10210 [15]

(ii) Se trabajará en colaboración con ingeniero que es parte del proyecto y el encargado de desarrollar la aplicación final del proyecto.

1.4. Metodología

Para realizar el presente trabajo, se realizó una revisión bibliográfica por un lado a las bases psicológicas y biológicas en el desarrollo cognitivo del lenguaje en el ser humano, y por el otro de los actuales softwares presentes y los desafíos asociados al uso de dispositivos móviles para estimulación cognitiva.

Se revisó el software previo dentro del marco de la misma investigación, en conjunto a la retroalimentación a considerar en esta versión. Se analizó la aplicación principal de estimulación y su actual avance, para la integración correcta de las funcionalidades entre ésta y la que se desarrolló en este trabajo.

A partir de lo anterior, se estudió el diseño de la aplicación secundaria y base de datos asociada.

Posteriormente se procedió a desarrollar el prototipo final, y a evaluar su comportamiento en diferentes dispositivos y escenarios.

1.5. Temario

(i) Capítulo 1: Introduce de forma general al presente trabajo, señalando sus objetivos y alcances.

(ii) Capítulo 2: Se presenta el marco teórico que fundamenta el trabajo, que se divide en cuatro partes: una revisión de las bases psicológicas y fisiológicas en el desarrollo del lenguaje en niños; un análisis de las aplicaciones existentes en el mercado actual, en conjunto a la versión previa de la misma investigación; una presentación de las características del sistema Android, comunicación Bluetooth y base de datos analizadas; y, finalmente, un análisis de la aplicación principal de estimulación SIPLYC.

(iii) Capítulo 3: Se presenta el diseño de la aplicación secundaria, junto a la descripción de sus diferentes módulos.

(iv) Capítulo 4: Se presenta el desarrollo e implementación de las funcionalidades descritas en el capítulo anterior.

(v) Capítulo 5: Se presenta la evaluación del sistema, mediante pruebas de las diferentes funcionalidades requeridas.

(vi) Capítulo 6: Se presenta la discusión del trabajo, en conjunto a sus conclusiones. Se describe el trabajo futuro enmarcado dentro de la actual investigación, para mejorar el software desarrollado.

Capítulo 2. Marco Teórico

2.1. Introducción

Como indica el *National Institute of Health* (NIH) de Estados Unidos a través del *National Institute of Deafness and Other Communication Disorders* (NIDCD), durante sus primeros años de vida, el bebé tiene un acercamiento a la comunicación al aprender que puede pedir compañía y alimento a través del llanto, lo cual tiene una progresión natural, a medida que reconoce sonidos de la lengua materna y luego comprende que son asociados a ciertos objetos y mensajes. Durante los tres primeros años, su cerebro se encuentra en proceso de maduración, y se trata de períodos claves en los que el niño debe ser expuesto a estimulación para "absorber el lenguaje" [17].

En el presente capítulo, se revisa de forma general como se desarrolla el lenguaje del niño en sus primeros años. También se presenta información sobre otras aplicaciones que busquen un objetivo similar o relacionado al propuesto para SIPLYC. Se finaliza con una revisión del sistema Android, el entorno de desarrollo, en conjunto a las tecnologías utilizadas para comunicación inalámbrica y Base de datos.

2.2. Desarrollo del Lenguaje en Niños

2.2.1 Anatomía a cargo del lenguaje

Para comprender el desarrollo del proceso del habla en el ser humano, es necesario entender de forma general el funcionamiento del cerebro. A grandes rasgos, se habla de una literacidad de las funciones cerebrales, asociando el hemisferio izquierdo a una dominancia en tareas como reconocimiento de palabras, sonidos lingüísticos y del habla. Esto, que es observable en la Fig. 2.1, corresponde a los resultados de la investigación realizada por Bavelier (1997) en la cual se analizó la actividad asociada a la lectura en varios individuos mediante *Imagen de Resonancia Magnética funcional* (fMRI), detectándose gran activación del hemisferio izquierdo en donde se encuentran las estructuras anatómicas tradicionalmente relevantes con estas funciones [18].

De acuerdo al modelo clásico de Wernicke y Geschwind, la corteza cerebral cuenta con siete áreas encargadas del proceso de comprensión y expresión del habla: área de Broca, corteza motora primaria, fascículo arqueado, corteza auditiva primaria, área de Wernicke, circunvolución angular y corteza visual primaria (ver Fig. 2.2).

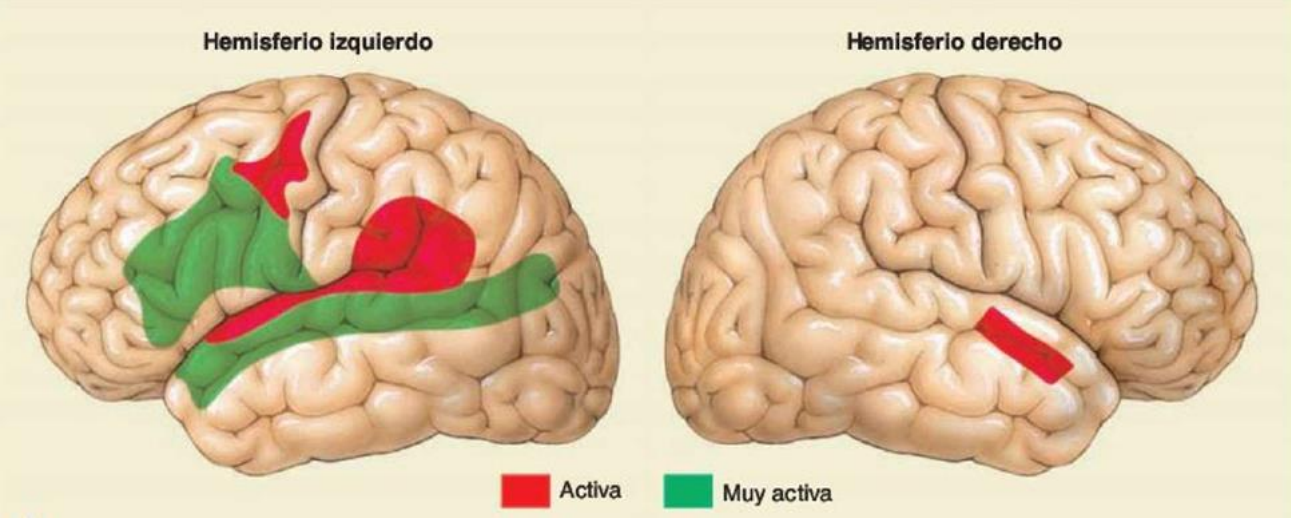


Fig. 2.1. Áreas con aumento de actividad asociada a la lectura, de acuerdo a un análisis por fMRI.

Fuente. (Pinel J.) Biopsicología [18]

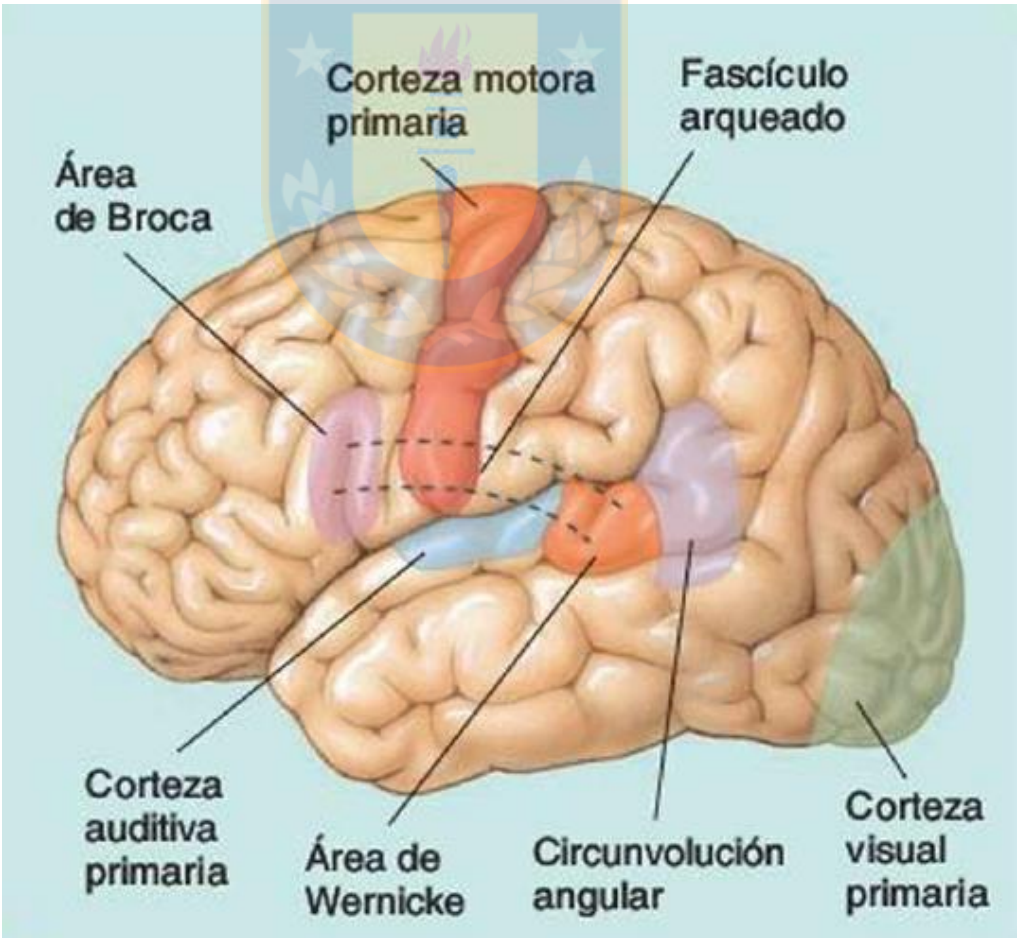


Fig. 2.2. Zonas involucradas en Proceso del habla, según Wernicke y Geschwind.

Fuente. (Pinel J.) Biopsicología [18].

Cada una de estas regiones, de acuerdo al modelo clásico, cumple una tarea específica. Por ejemplo, en el caso del ingreso de un estímulo auditivo, este es recibido en la corteza auditiva, comprendido en el área de Wernicke, la respuesta es generada y transmitida vía fascículo arqueado al área de Broca, que es la encargada de activar a la corteza motora primaria para articular el habla. Si bien este enfoque clásico es apoyado en el campo académico y clínico, hoy los investigadores han girado hacia el enfoque de la neurociencia cognitiva, este plantea que las funciones descritas por Wernicke y Geschwind, pueden descomponerse en tres procesos cognitivos asociados al lenguaje: análisis fonológico (del sonido), gramatical (de su estructura) y semántico (de su significado). Esto se contrapone a la idea del modelo tradicional en el que se describen áreas amplias, homogéneas y circunscritas. El nuevo enfoque propone que estos procesos cognitivos corresponden a áreas pequeñas, especializadas y distribuidas, considerando investigaciones actuales que indican que las siete áreas descritas previamente participan también en otros sistemas funcionales. Es importante señalar que mientras el modelo clásico nace a partir del estudio de pacientes con afasias de lenguaje, el enfoque en la neurociencia cognitiva se basa en el uso de técnicas de neuroimagen (como fMRI) en el estudio de individuos sanos, lo que podría explicar sus diferentes acercamientos [18].

A nivel global, el recién nacido cuenta con tres herramientas para el desarrollo del habla. El sistema sensorial (principalmente ligado a la audición, visión y propiocepción), el sistema motor (referido a los músculos de la fonación y gestos no verbales) y finalmente la potencia intelectual (define el ritmo de adquisición del lenguaje). Los dos primeros fueron ampliamente descritos por Launay y Borel-Maisonny (1975), quienes los denominaron "*vertientes de aprendizaje verbal*", identificando al sistema sensorial como las "*gnosias perceptivas*" y al motor como las "*praxias articulatorias*" [19]. Es importante indicar que estas herramientas del bebé se ven enfrentadas inmediatamente al medio sociocultural, y de su interacción se producirá el fenómeno de adquisición del lenguaje [19].

2.2.2 Etapas en el desarrollo del lenguaje

En la presente sección se describen las etapas en el desarrollo del lenguaje para niños sanos, desde una visión psicolingüística y desde la psicología evolutiva. Para esto, se utilizó la descripción planteada por Castañeda P. F. (1999), quien a su vez se basa en aportes de diferentes autores como Brown y Frazer (1964), Lenneberg (1967), Bateson (1975); Einsenson (1979); Bruner (1976) y otros, por lo que representa un resumen de los aspectos más importantes [19].

A. Etapa Pre-Lingüística o Pre-Verbal

Hoy en día se sabe que esta etapa es muy importante para el desarrollo del lenguaje y comprende el primer año de vida del recién nacido. La comunicación establecida por el niño es de tipo gestual y afectiva, por lo que se espera que la madre o tutor estimule el lenguaje a través de sonidos y palabras aisladas en las actividades con el lactante [19].

(i) **Del nacimiento a dos meses de edad:** El bebé se comunica a través del llanto. Si bien en un principio es un sonido indiferenciado, a los dos meses comienza a variar el tono manifestando diferentes sentimientos como dolor, hambre o malestar.

(ii) **Tres a cuatro meses de edad:** Bebé produce sonidos que expresan placer a través de sonidos guturales (*gaga*) o disgusto por sonidos nasales (*nga nga*), puede distinguir y reaccionar a las entonaciones de la voz de sus padres. Para estimular el lenguaje, se debe responder a estos sonidos.

(iii) **Cinco a seis meses de edad:** Bebé es capaz de imitar sonidos. La /a/ es la primera vocal adquirida, mientras que /i/, /u/ son las últimas. Las consonantes se clasifican en labiales (/p/, /m/, /b/), dentales (/d, /t/) y velopalatales (/g/, /j/), apareciendo en este orden.

(iv) **Siete a ocho meses de edad:** Se desarrolla una "*protoconversación*", donde el niño pasa a un modo de intercambio entre interacción tutor-niño, que puede estimularse nombrando e intercambiando objetos, como entrenamiento del habla.

(v) **Nueve a diez meses de edad:** Niño puede vocalizar palabras cortas, todavía como imitación repetivita que puede ser reforzado por sus padres. El desarrollo de músculos accesorios del habla favorece la articulación.

(vi) **Once a doce meses de edad:** Niño puede vocalizar sus "*primeras palabras"* de dos sílabas ("papá" "mama" "tata"), a las cuales aún no le atribuye un significado. Se observa síntesis (dice 'sopa', en vez de 'mamá, dame sopa'), sustitución ('topa' en vez de 'sopa') o supresión ('opa' en vez de 'sopa').

B. Etapa Lingüística

La primera palabra con un propósito de comunicación (voluntario y no por imitación) marca el inicio de esta etapa. Se estima que la primera palabra en la mayoría de los niños será entre los 15 y 18 meses, sin embargo, esta afirmación se basa en observaciones de padres [19].

(i) **Doce a catorce meses de edad:** Se denomina etapa “holofrásica”, en la cual una palabra reemplaza a una frase y puede tener diferentes significados. Por ejemplo, ‘abe’ en reemplazo de “abrir la puerta” o “abrir la fruta”. Los padres deben estimular asociando sonidos con significados, para que el niño adquiera estas relaciones.

(ii) **Quince a dieciocho meses de edad:** A esta edad el niño cuenta con un vocabulario de entre 5 a 20 palabras y empieza a utilizar combinaciones de dos palabras. En esta edad se recomiendan ejercicios de identificación de objetos, figuras y partes del cuerpo.

(iii) **Dieciocho a veinticuatro meses de edad:** El niño cuenta con un repertorio por sobre las 50 palabras y es capaz de combinar dos a tres en una frase simple, entre sustantivos (“zapato mamá”), entre nombre y verbo (“tata come”) y con adjetivos (“pelota bonita”). A los dos años su vocabulario aumenta a unas 300 palabras, además de utilizar pronombres personales y de pertenencia (‘mío’). Aparece la capacidad simbólica, que le permite expresar cosas que no estén realmente presentes (expresar constructos en su mente, que le permitirá dar palabras a realidades abstractas y emociones). El estímulo de los padres se puede dar a través de cuentos y narraciones.

(iv) **Dos a tres años de edad:** Incremento rápido del repertorio de palabras cercano a las 1200 al final de este período. El niño puede utilizar verbos auxiliares (‘haber’, ‘ser’). Se denomina el período de competencia sintáctica, en el cual el niño cuenta con un lenguaje más compresible, incluso por personas fuera del núcleo familiar.

(v) **Cuatro a cinco años:** Vocabulario cercano a las dos mil palabras. El niño puede responder y comprender preguntas asociadas al comportamiento social que se le ha enseñado. La capacidad simbólica desarrollada, le permite aumentar su necesidad de comunicación, lo cual tiene un efecto positivo en el desarrollo cognitivo y del lenguaje. Esta edad se caracteriza por una retórica propia dominada por el ‘yo’.

(vi) **Seis a siete años:** Se inicia la edad escolar, se supera la etapa egocentrista para dar lugar a un estilo lógico y concreto. El niño toma conciencia de sí mismo, genera una autoimagen y toma en cuenta los comentarios hacia él.

La propia NIDCD dispone de diferentes cuestionarios o test online para padres, con un set de pruebas que les permiten identificar retrasos en alguno de los procesos en el desarrollo del lenguaje, con el fin de que los padres sean agentes capaces detectar a tiempo trastornos del habla o del lenguaje de forma temprana [17].

El proyecto SIPLYC se enfoca específicamente en el período entre los dos y tres años y medio, donde se observa un gran aumento en el vocabulario de los niños. A esta edad el niño debería ser capaz de "*dar una palabra a casi todo*", "*hacerse a entender a los miembros de la familia y amigos*", ser capaz de armar frases para "*nombrar objetos y pedir la atención*", de acuerdo al NIH [17]. Corresponde a un período clave en el desarrollo del lenguaje, previo a la edad escolar, por su relación e importancia en el desarrollo cognitivo.

2.3. Aplicaciones como Herramientas de Apoyo y Estimulación del Lenguaje

2.3.1 Algunas aplicaciones disponibles en el mercado

La masificación de los dispositivos móviles cada día con mayor capacidad de procesamiento o más "inteligentes", en conjunto a su gran penetración en la sociedad, desencadenó una explosión en la cantidad de aplicaciones o apps disponibles en el mercado. Android con su *PlayStore* e iOS con su *iTunes AppStore*, cuentan con *apps* para ayudarnos con nuestras tareas diarias, entretención e incluso para aprender una nueva Lengua. Pero, ¿qué ocurre a nivel de estimulación temprana del lenguaje?

La organización privada *Children's Speech Therapy* del Norte de Irlanda, liderada por Linda Wilson, especialista en terapia del lenguaje y el habla, recomienda una colección de aplicaciones enfocadas en desarrollo del lenguaje para niños pequeños menores a 6 años [20]. Entre estas se destaca *Articulation Station Español* [21] (ver Fig. 2.3) que, como su nombre lo indica, cuenta con una versión que soporta el idioma español. Esta app posee módulos por sonidos, que a su vez cuentan con actividades de práctica, juego de pares, y generación de frases. La aplicación permite a los tutores, padres o profesores grabar al niño, para que pueda escucharse y calificar su desempeño directamente en la aplicación, generando, además, un módulo para la personalización, con compilación del progreso del niño (o múltiples usuarios), indicando las palabras correctas, incorrectas y aproximadas. Permite crear listas de palabras personalizadas y añadir sus propias imágenes. En los aspectos negativos, pese a que es un software desarrollado por padres, además de profesionales en patologías del lenguaje y habla de Utah y Texas, no cuenta con investigación de respaldo en su efectividad. Recomiendan el uso de la app en conjunto a un adulto pues así está diseñada. La aplicación se encuentra disponible sólo para iPad, iPhone y iPod. En su formato gratuito sólo contiene la consonante /p/, y los demás fonemas tienen un valor entre 2.99-4.99 (USD) cada uno, lo cual aumenta su valor a cerca de 60 dólares, lo cual, en conjunto a su exclusividad para dispositivos Apple, la hace muy restrictiva.

Fig. 2.3. Captura de "Articulation Station Español".
(a) Actividad de vocabulario. **(b)** Actividad de generación automática de frases.
Fuente. Sitio web oficial de la aplicación [21].

Fig. 2.4. Captura de "iTouchiLearn Words"
(a) Actividad de asociación de palabras. **(b)** Menú de inicio.
Fuente. Canal oficial del desarrollador en YouTube [22].

Otra aplicación interesante es *"iTouchiLearn Words"* [22] (ver Fig. 2.4). Esta aplicación cuenta con diferentes actividades de juego y material didáctico. Contiene videos que dan contexto a palabras (por ejemplo. 'huevo, pollo, granja'), asociar palabras con su imagen y canciones interactivas. Sin embargo, esta aplicación muestra poca actualización desde que se dejó disponible en 2011. Se encuentra disponible exclusivamente para dispositivos Apple y en inglés. Otras apps presentes son [20]:

(i) "100 words for Babies & Toddlers", que incluye un gran vocabulario con animaciones, pero no permite la evaluación ni genera estadísticas.

(ii) "ABC Alphabet Phonics", que permite la personalización de sonidos, con la pronunciación y voz del padre, pero solo incluye sonidos de letras, no palabras.

(iii) "Speech With Milo: Sequencing", que permite identificar secuencias de acciones a través de animaciones.

Estas aplicaciones mencionadas previamente, pese a estar enfocadas al rango de edad de interés de este trabajo, no cuentan con publicaciones o investigación asociada que respalde su efectividad, más allá del que entregan sus creadores desde su visión como especialistas o padres.

Es posible encontrar otras aplicaciones similares en la PlayStore de Android, sin embargo, tampoco presentan respaldo científico dentro de su descripción, además de poseer quejas de parte de los usuarios por contenido de pago o por exceso de publicidad.

2.3.2 Estudios asociados

A partir de lo presentado en la sección anterior, nace el cuestionamiento de si existen investigaciones asociadas a la efectividad de las apps educativas para dispositivos móviles. La respuesta es afirmativa: como se planteó en la introducción a este trabajo, se han realizado múltiples investigaciones debido a la preocupación de la comunidad científica ante el acceso temprano de los niños a dispositivos portátiles y contenidos digitales. Si bien algunos investigadores asociaron el uso de celulares y tablets al desarrollo de hiperactividad [12] o con patrones de sueño alterados [13], otros estudios (como el de la Universidad de Stanford) indicaron que, dotados de contenido pedagógico de alta calidad y acompañados por la estimulación del tutor, profesor o padres, los dispositivos móviles se pueden transformar en herramientas de transformación social en el apoyo de estimulación temprana y en la disminución de la brecha educativa en familias de escasos recursos [14].

Existe bastante bibliografía asociada al uso de apps en tablets dirigidas a niños con Trastorno de Espectro Autista (TEA). Como bien señala una investigación desarrollada por la Southern Illinois University Edwardsville Research (King A., Brady K., Voreis G. 2017) los dispositivos son utilizados como videos de instrucciones, instrumentos de comunicación (como apoyo en el habla) o como herramientas de aprendizaje escolar (como en el desarrollo del lenguaje). Sin embargo, los resultados de su estudio señalaron que era necesaria una mayor participación de docentes y del uso de apps para TEA en la propia sala de clases, para evaluar y desarrollar este tipo de aplicaciones de forma más efectiva [23].

Durante la propuesta del proyecto *SIPLYC* [15], se destacó la relevancia del software GraphoGame, aplicación desarrollada en Finlandia con un enfoque científico basado en evidencia y dirigido a una edad escolar. A diferencia de otras propuestas, este software cuenta con cerca de veinte años de respaldos en investigación y con publicaciones asociadas, demostrando efectividad en el aumento de literacidad para diferentes tipos de ortografía [24]. Lamentablemente, se encuentra dirigido a un grupo etario diferente.

Para el presente trabajo, es interesante revisar los resultados del estudio presentado por investigadores de la Universidad de Leiden (Bus A. Takacs Z. Swart E. 2015). En este trabajo se realizaron pruebas para determinar la efectividad de historias y cuentos interactivos en el desarrollo cognitivo y del vocabulario de niños. Se observó que elementos multimedia como animaciones, músicas y efectos de sonido fueron positivos para mantener la concentración y mejorar el aprendizaje de palabras. Al mismo tiempo, se determinó que otros elementos interactivos como juegos y diccionarios fueron distractores y sacaron los niños de la inmersión del relato. Aquellos niños con desventaja lingüística (que los investigadores asociaron a familias de escasos recursos, padres con el inglés como segunda lengua o a un atraso en el desarrollo con respecto a sus pares), fueron quienes presentaron una mayor distracción frente a elementos interactivos [25]. Esta última conclusión es importante, debido al grupo al que se busca enfocar el software a desarrollar.

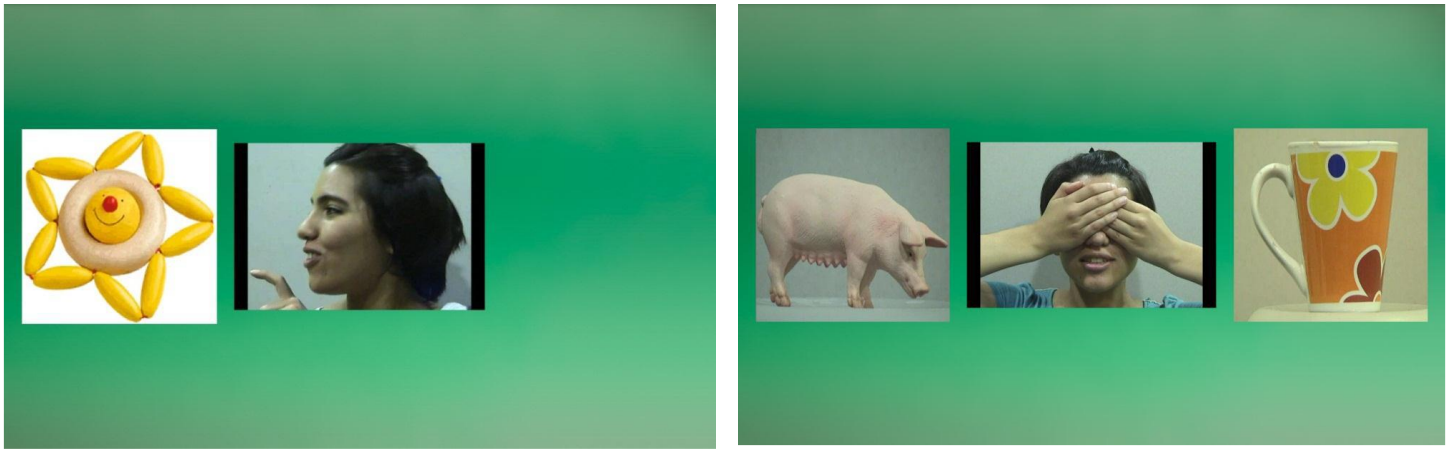
2.3.3 Sistema de Estimulación del Lactante (SIEL)

Dentro de lo que es el trabajo previo de los investigadores a cargo del *SIPLYC*, se desarrolló un software denominado Sistema de Estimulación del Lactante (*SIEL*), enfocado a niños entre los 18 y 24 meses [15]. A continuación, se describirá de forma global esta experiencia, ya que se considera como un insumo relevante para la actual investigación.

A. Descripción de aplicación SIEL

La aplicación *SIEL* poseía actividades en las cuales se enfrentaba al niño a palabras frecuentes (que debería aprender en su casa), infrecuentes y nuevas. Como parte de este proyecto, se realizó una Memoria de Título titulada "*Sistema interactivo de Estimulación del lactante para plataformas Android*" [26], en la cual se implementó el prototipo de la app para tablets.

Desde el punto de vista técnico, esta aplicación permitía el ingreso de diferentes usuarios, contaba con una Base de datos local mediante SQlite y archivos de texto. Contaba con un modo de Vocabulario, que correspondía a la actividad de juego, y un módulo de Estadísticas, en el cual se podía visualizar tiempo promedio de respuesta, respuestas correctas/incorrectas por sesión y cantidad de palabras aprendidas por sesión (contando con la opción de generar un archivo .pdf de reporte con los resultados). El acceso a la aplicación y a los datos era restringido con el uso de contraseña, con el fin de proteger la privacidad de los datos [26].



(a) (b)

Fig. 2.5. Captura de prototipo en Android de aplicación SIEL
(a) en estado de 'Testing' **(b)** en estado de 'Asking'.
Fuente. Cid R. Informe de Memoria de Título [26]

El módulo de Vocabulario contaba con tres estados:

(i) **Training:** donde se le explica al niño lo que se hará durante la actividad.

(ii) **Testing:** donde se le presenta una palabra al niño, y se le pide que seleccione la imagen (siendo la única presente), de acuerdo a la Fig. 2.5.a.

(iii) **Asking:** donde se le presentan dos imágenes y se le solicita que selecciona la que representa a una palabra dada, de acuerdo a la Fig. 2.5.b.

La aplicación seleccionaba al azar diferentes palabras para cada sesión. Se contaba con una actividad de Recreo, que interrumpía la actividad de juego cada cinco minutos, mostrando un video, con el fin de mantener la atención del menor en la pantalla.

B. Resultados de SIEL

La aplicación desarrollada se evaluó en 128 niños entre 18 y 24 meses de edad, durante 6 sesiones de veinte minutos (una sesión, día por medio). Los niños tenían diferentes orígenes, de acuerdo a la distribución de TABLA 2.1.

TABLA 2.1. DISTRIBUCIÓN DE INDIVIDUOS EVALUADOS CON SIEL

Origen	Total de individuos
En laboratorio	87
Fundación INTEGRA	21
SENAME	20

Fuente. Datos entregados en Formulación Proyecto SIPLYC [15].

Se observó que los niños con mayor estado de vulnerabilidad presentes en los centros de protección de la infancia (SENAME), se mostraron menos interesados en participar que los individuos evaluados en el centro de cuidado de Fundación INTEGRA y en el laboratorio. Si bien en los tres grupos se observaron comportamientos de "respuesta contingente" y aprendizaje "a corto y largo plazo de palabras infrecuentes", los niños en mayor riesgo socioeconómico tuvieron un tiempo efectivo de juego cercano al 20% [15].

C. Comentarios y consideraciones

Algunos aprendizajes y observaciones que se pueden obtener a partir de comentarios con respecto a la experiencia previa son:

- **(i)** Es importante considerar el aspecto visual. Con el fin de hacer más atractiva la aplicación para niños pequeños.
- **(ii)** La aplicación funcionaba sólo en un computador con pantalla táctil, lo que permitía el uso de un solo profesor, lo cual limitaba el análisis de los datos a un solo dispositivo.
- **(iii)** La aplicación tenía algunos problemas de ejecución, los que pudieron ser provocados por el tamaño de los archivos multimedia, por lo que es necesario contar con una buena compresión de los archivos a utilizar.
- **(iv)** La aplicación evaluaba vocabulario a través del contacto con la pantalla. Sin embargo, no consideraba ni evaluaba el habla del niño, el cual es un aspecto significativo en el desarrollo del lenguaje.
- **(v)** La aplicación generaba recreos de forma automática cada 5 minutos. Sin bien, era posible definir esos intervalos dentro de la aplicación, el tutor no podía generar uno de forma manual cuando identificara distracción del niño.

A partir de la experiencia adquirida con el desarrollo de *SIEL*, el equipo de investigadores planteó la necesidad del desarrollo de una aplicación que funcione en múltiples dispositivos, y una aplicación de Tutor, encargada de la visualización de las estadísticas y del control de las actividades, para poder generar recreos al momento de identificar intervalos de distracción, así como permitir evaluar el Habla. Estas necesidades fueron consideradas en la realización del presente trabajo para el diseño e implementación del *SIPLYC*.

2.4. Consideraciones de Software

2.4.1 Android Studio

Para el desarrollo de las aplicaciones se utilizará *Android Studio* (v2.3.2 abril 2017) [27]. Este es el entorno de desarrollo (IDE) oficial para el sistema operativo Android, empleado por gran cantidad de dispositivos móviles, y por los tablets disponibles para esta investigación.

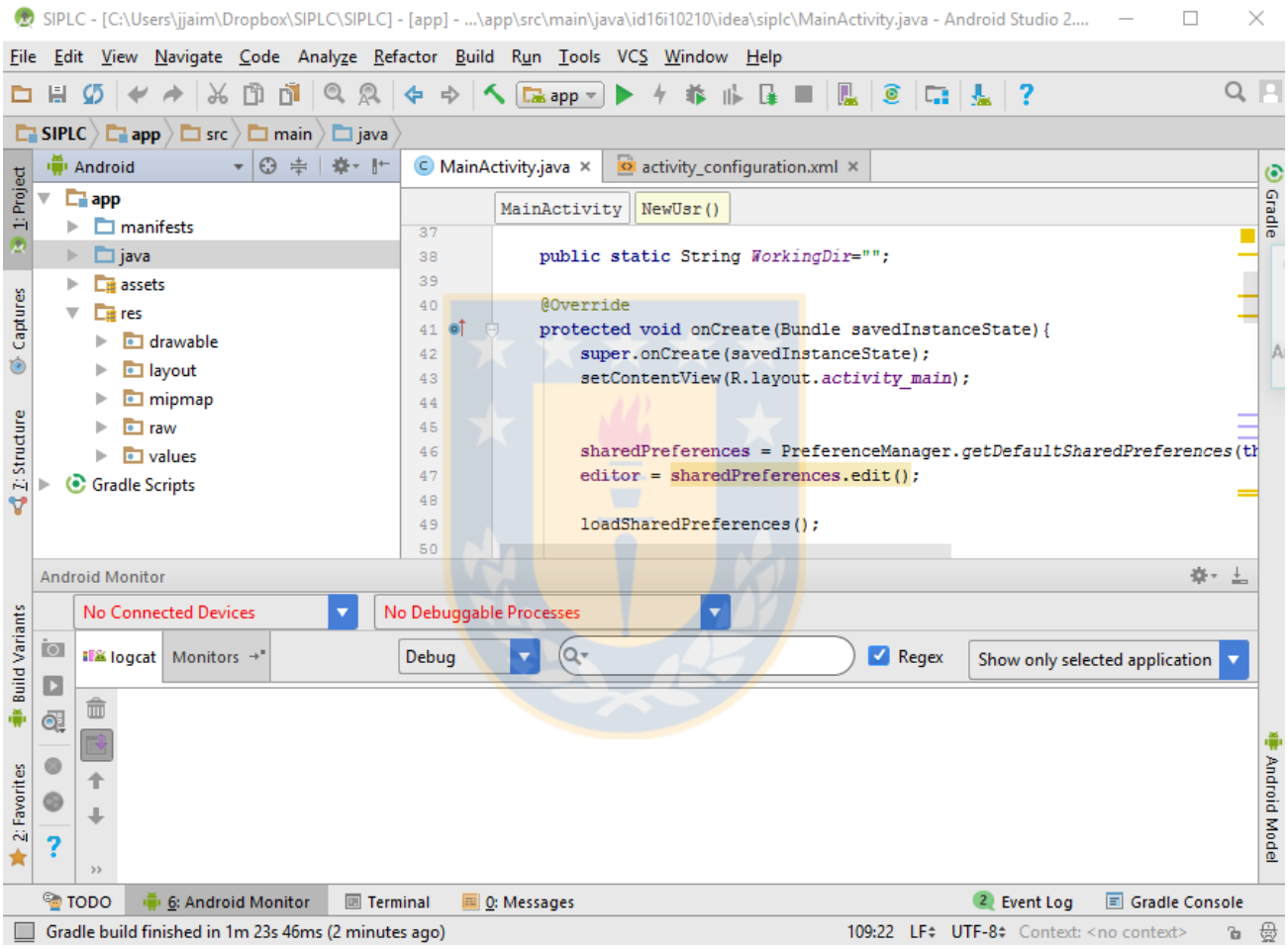


Fig. 2.6. Captura de pantalla de Android Studio v2.3.2.
Fuente: Software disponible para descarga en sitio web oficial de Android Developers [27]

De forma general, la programación en Android, es a través de "*Actividades*", cada actividad corresponde a una pantalla (o Screen) y a lo que ocurre en ella. Esta es programada en lenguaje *JAVA*. Además, la distribución de elementos es definido a través de diseños en *XML* (los cuales se pueden definir para diferentes tamaños de pantalla).

Android Studio maneja las aplicaciones como "*Proyectos*", y posee tres carpetas con los archivos a compilar [27]:

(i) **Manifest:** que contiene el archivo de manifest, que permite administrar los permisos para la aplicación.

(ii) **Java:** Que contiene los archivos de Java de las diferentes "Actividades".

(iii) **Res:** Que contiene los recursos de la aplicación, como medios y archivos *XML*.

En mayo de 2017, durante la conferencia para desarrolladores *Google I/O*, se anunció que el lenguaje de programación *Kotlin*, pasaría a ser el nuevo lenguaje oficial para Android. Este lenguaje fue construido por *JetBrains* (desarrolladores de Android Studio), y se perfila en la actualidad como el posible sucesor de *JAVA*, ya que *Kotlin*, pese a ser un lenguaje de programación estático y orientado a objeto, posee una sintaxis más natural y funcional, entre otras características [28].

Como este anuncio se dio en medio del desarrollo de este trabajo, además que el soporte oficial se dará con la nueva versión de *Android Studio v3.0* (disponible actualmente solo como *preview*), la realización de la app se hará con la versión anterior.

2.4.2 Descripción general de una aplicación en Android

A. Actividad

Como se mencionó anteriormente, las aplicaciones diseñadas para sistemas operativos Android se encuentran compuestas por "*Actividades*" que corresponden a las diferentes pantallas o "*Screen*" con las que el usuario interactúa. Para que una actividad pueda ser accedida, debe ser declarada en el archivo de *Manifest* (Fig. 2.7.a) y ser iniciada llamando al método *startActivity(),* mediante un *Intent*. Como se observa en la Fig. 2.7.b, un *intent* especifica la nueva actividad que se quiere iniciar y permite transferir datos a esta mediante el método *putExtra()* [29].

```
<manifest ... >
  <application ... >
      <activity android:name=".ExampleActivity" />
      ...
  </application ... >
  ...
</manifest >
```

(a)

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.putExtra(Intent.EXTRA_EMAIL, recipientArray);
startActivity(intent);
```

(b)

Fig. 2.7. Declaración y llamado a inicio de una Actividad.
(a) Declaración de Actividad en archivo Manifest del proyecto
(b) Declaración de Intent de Actividad y su llamado a inicio.
Fuente: Android Developers, "Actividades" [29].

(a) **(b)** **(c)**

Fig. 2.8. Tipos de diseño de Layout
(a) LinearLayout **(b)** RelativeLayout **(c)** ListView
Fuente: Android Developers, "Diseño" [30].

Los elementos de la Interfaz de Usuario (UI) como botones, imágenes y textos se denominan *Views*, y son definidos dentro de un *GroupView* o *Layout*, que permite distribuirlos en la pantalla. Los diseños más comunes son [30]:

(i) **LinearLayout:** ordena los elementos en una sola fila o columna (Fig. 2.8.a).

(ii) **RelativeLayout**: permite organizar los elementos en relación a otros elementos secundarios (ej. Botón a la derecha de la imagen) o a un contenedor (ej. Botón alineado al borde superior del *Layout*) (Fig. 2.8.b).

(iii) **ListView:** permite generar una lista de objetos. Utiliza un *Adapter*, que es un objeto que permite leer una fuente de datos (ej. Una lista de nombres de usuario) y agregarlos al *ListView*, de forma programática (Fig. 2.8.c).

Si bien los elementos de la UI pueden ser inicializados programáticamente desde la Actividad, es recomendado declararlos y organizarlos como XML. Esto ofrece la ventaja de generar diseños para diferentes orientaciones, pantallas o idiomas, sin necesidad de hacerlo mediante códigos.

Los objetos View y Layout poseen atributos que permiten definir parámetros de diseño (tamaño, backgroud, color, orientación) directamente en XML (ej. Fig. 2.9.a). Se puede, además, definir un ID (identificador) que permite generar una instancia al objeto desde la aplicación para modificar sus propiedades o comportamiento (ej. Fig. 2.9.b).

```
<Button android:id="@+id/my_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/my_button_text"/>
```

(a)

```
Button myButton = (Button) findViewById(R.id.my_button);
```

(b)

Fig. 2.9. Declaración e Instancia a View
(a) Declaración de un Botón en XML **(b)** Instancia al objeto Button mediante su ID.
Fuente: Android Developers, "Diseño" [30].

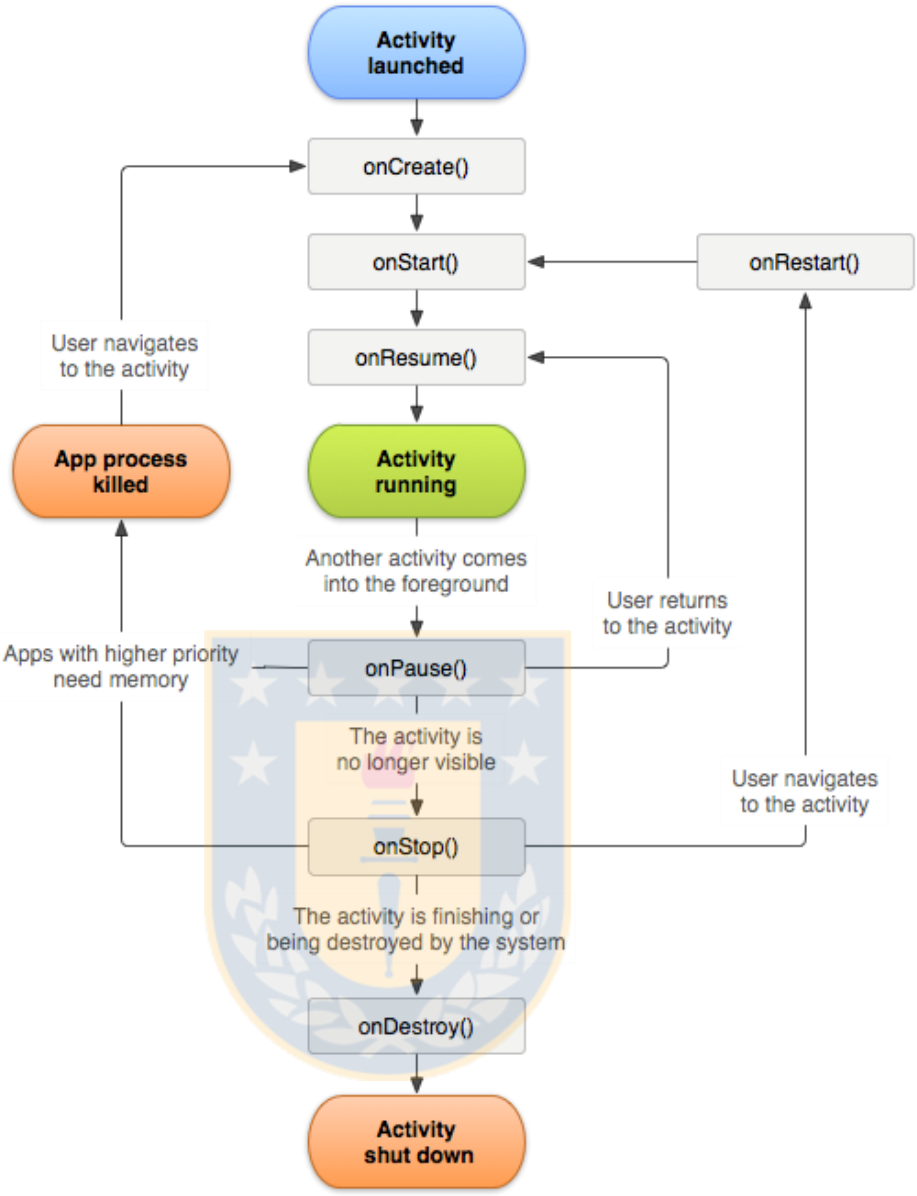


Fig. 2.10. Ciclo de vida de una Actividad.
Fuente: Android Developers, "Actividades" [29].

Una Actividad en Android cumple con un ciclo de vida definido de acuerdo a la Fig. 2.10. Al ser llamada, sobrescribiendo el método *onCreate()*, es posible cargar un recurso XML de layout para definir el diseño de la actividad (ver Fig. 2.11). En este método generalmente se realiza la declaración de instancias a las Views y se definen las interacciones que se tendrá con las mismas.

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main_layout);
}
```

Fig. 2.11. Carga de recurso XML con layout de la actividad
Fuente: Android Developers, "Diseño" [30].

Android posee diferentes formas para identificar y responder a eventos generados por interacción del usuario. Para esto, las *Views* cuentan con *Event Listeners* para "escuchar" estos eventos, como, por ejemplo [31]:

(i) **View.onClickListener:** llama al método *onClick()* cuando detecta que el usuario tocó el objeto.

(ii) **View.onLongClickListener:** llama al método *onLongClick()* cuando detecta que el usuario tocó y mantuvo presionado el objeto.

(iii) **View.onTouchListener:** llama al método *onTouch(),* cuando se detecta que el usuario realizó un evento táctil, presionar, soltar, y diferentes tipos de movimientos (*MotionEvent*) como desplazamientos, rotaciones, zoom-in, zoom-out, entre otros.

Existen extensiones de estos *Listeners* para views más complejas, como *onItemSelectedListener(),* que identifica el ítem seleccionado en un *ListView* o un *Spinner* (lista desplegable) [31]. Finalmente, estos eventos "escuchados" llaman a los métodos correspondientes, los cuales pueden ser editados para programar las respuestas deseadas.

2.4.3 Base de Datos

Para el desarrollo de la aplicación principal se utilizará una base de datos local por cada dispositivo, que almacenará los datos de los usuarios que utilicen la app en el mismo. Estos datos deberán sincronizarse con una base de datos a través de internet, que serán utilizados por la aplicación del tutor para acceder a la información recopilada por los diferentes dispositivos.

A. Diseño por Arquitectura centrada en SyncAdapter

A continuación, se describe una arquitectura centrada en *SyncAdapter,* con el uso de *SQlite* como Base de datos local, y *MySQL,* montando un servidor web.

(i) **SQlite:** Android Studio cuenta con el *API* (Interfaz de Programación de Aplicaciones) en el paquete: *android.database.sqlite,* que contiene lo requerido para el manejo de una Base de datos local [32].

(ii) **SyncAdapter:** Android Studio cuenta con una clase que permite funcionar como un intermediario para la sincronización de archivos con un servidor web. Esta puede ejecutarse en el segundo plano, en un intervalo de tiempo, cuando la red esté disponible o por orden del usuario [32].

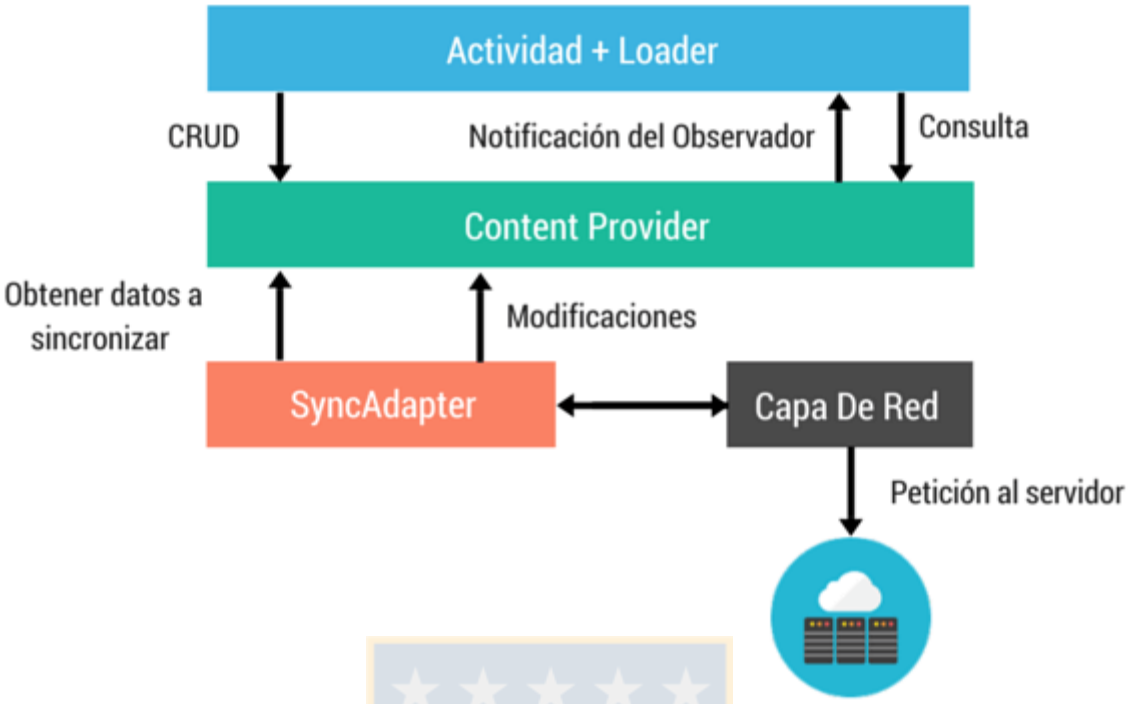


Fig. 2.12. Esquema de arquitectura centrada en un SyncAdapter.
Fuente. Tutorial de arquitectura centrada en SyncAdapter [32].

(iii) **MySQL:** Permite la implementación de un servidor web, cuenta con una versión gratuita (*MySQL Community Edition*) para uso no comercial. Se define a sí misma como la "*Base de datos de open source más popular del mundo*"[33] y utiliza un sistema basado en peticiones y consultas en lenguaje *SQL* .

La arquitectura propuesta, se presenta en la Fig. 2.12, donde es posible identificar la interacción entre la aplicación (*Actividad*, *Loader*), un proveedor de contenido (que sería la base de datos local), el *SyncAdapter* y el servidor web. La aplicación principal de juego genera contenido que es almacenado en un servidor web y que posteriormente puede ser revisado por el tutor en la aplicación secundaria.

B. Arquitectura basada en Firebase Realtime Database

Firebase Realtime Database o *Firebase* [34] es una base de datos en tiempo real con datos alojados en la nube y desarrollada por Google, que permite la sincronización instantánea de cada cliente conectado a internet y el funcionamiento sin conexión a través de una caché local con datos persistentes en el disco. La base de datos es accesible directamente desde dispositivos móviles Android, iOS o de un navegador (Fig. 2.13), sin necesidad de un servidor propio. La API cuenta con

diferentes bibliotecas para responder a las necesidades del desarrollador como *Analytics* (Estadísticas), *Realtime Database* (Base de datos en tiempo real), *Storage* (Almacenamiento), *Crash Reporting* (Reporte de Errores), *Remote Config* (Configuración Remota) y *Authentication* (Autenticación) [34].

La API de *Realtime Database* se encuentra optimizada para permitir operaciones de ejecución rápida y respuesta en tiempo real. Se basa en un lenguaje *NoSQL* pues no utiliza tablas relacionales, sino que utiliza objetos JSON (Ver.Fig. 2.14) . Esta se comporta como un "árbol JSON alojado en la nube"[34], tiene una limitación de anidación de 32 niveles de profundidad. Las recomendaciones de diseño indican que lo más eficiente es utilizar estructuras simples, pues el acceso de un cliente a la lectura de un nodo, le da acceso a todos los datos anidados como hijos de éste.

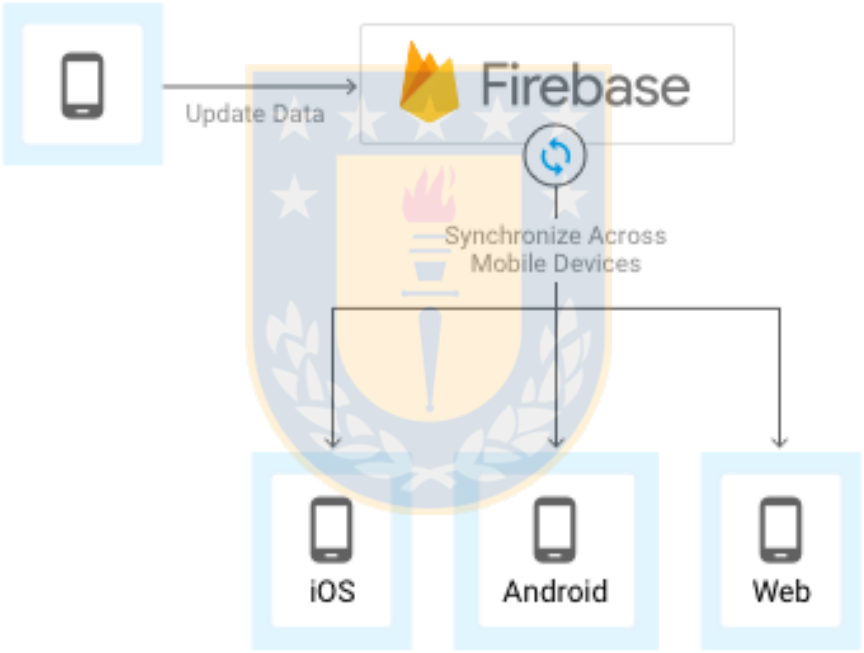


Fig. 2.13. Estructura de aplicación con el uso de Firebase
Fuente. Google Cloud Platform. Mobile App Backend Services [35].

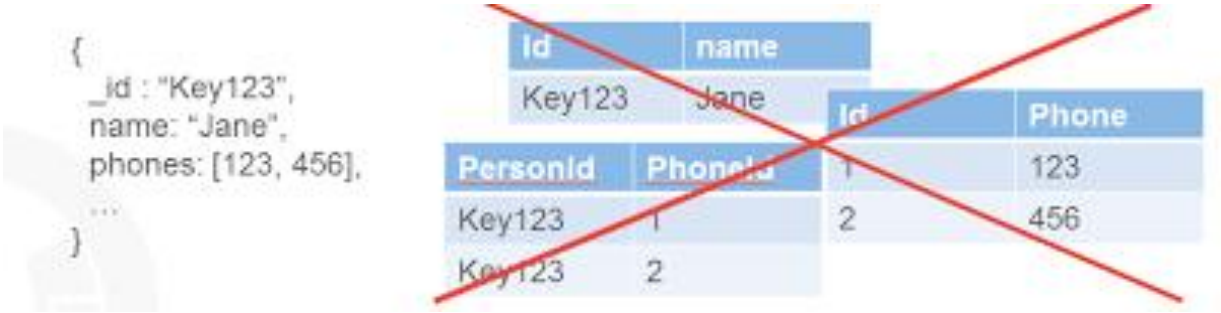


Fig. 2.14. Formato JSON y formato de tabla relacional.
Fuente. Blog comparativo entre bases de datos SQL y NoSQL[36].

Las características de *Firebase* la hacen indicada para la implementación de la Base de datos con el comportamiento deseado, sin la necesidad de recurrir a una arquitectura más compleja.

Para su uso es necesario considerar aspectos de instalación, además de referencias para el envío de datos y consultas (*Query*).

C. Configuración de Firebase en Aplicación

Aspectos a considerar para incluir la API de *Firebase* al proyecto de Android [37].

(i) **Requisitos Previos:** Se requiere un dispositivo con mínimo Android 4.0 y tener la versión de *Servicios de Google Play 10.2.1* o superior. Se debe instalar en *Android Studio* el SDK de *Servicios de Google Play*.

(ii) **Añadir la aplicación al Firebase Project:** *Firebase* cuenta con una consola mediante su interfaz web, que permite gestionar diferentes proyectos. Cada proyecto representa una Base de datos que se vincula a la aplicación en Android Studio. Se descarga un archivo *google-services.json*, el cual es único para cada aplicación.

(iii) **Agregar el SDK:** Se requiere incluir el complemento de google-services en el archivo *build.gradle* en la carpeta raíz del proyecto (Fig. 2.15). En el archivo "*app/build.gradle"*, (en módulo *app*) se incluye la línea de *apply* que activa el complemento, además de las dependencias de los SDK a utilizar (Fig. 2.16).

```
buildscript {
    // ...
    dependencies {
        // ...
        classpath 'com.google.gms:google-services:3.0.0'
    }
}
```

Fig. 2.15. Dependencias en /build.gradle.
Fuente: Documentación Firebase Realtime Database [37].

```
dependencies {
  // ...
  compile 'com.google.firebase:firebase-core:10.2.1'

  // Getting a "Could not find" error? Make sure you have
  // the latest Google Repository in the Android SDK manager
}

// ADD THIS AT THE BOTTOM
apply plugin: 'com.google.gms.google-services'
```

Fig. 2.16. Dependencias en app/build.gradle
Fuente: Documentación Firebase Realtime Database [37].

TABLA 2.2. BIBLIOTECAS DISPONIBLES DE FIREBASE

Línea de dependencia de Gradle	Servicio
com.google.firebase:firebase-core:10.2.1	Analytics
com.google.firebase:firebase-database:10.2.1	Realtime Database
com.google.firebase:firebase-storage:10.2.1	Storage
com.google.firebase:firebase-crash:10.2.1	Crash Reporting
com.google.firebase:firebase-auth:10.2.1	Authentication
com.google.firebase:firebase-messaging:10.2.1	Cloud Messaging y Notifications
com.google.firebase:firebase-config:10.2.1	Remote Config
com.google.firebase:firebase-invites:10.2.1	Invites y Dynamic Links
com.google.firebase:firebase-ads:10.2.1	AdMob
com.google.firebase:firebase-appindexing:10.2.1	App Indexing

Fuente. Documentación Firebase Realtime Database [37].

(iv) **Incluir bibliotecas en dependencia de Gradle:** Estas bibliotecas se ofrecen de forma gratuita con capacidad limitada (Tabla 2.2), *Realtime Database*, por ejemplo, con un límite de uso de 1GB en datos almacenados y 10GB descargados, además de un máximo de 100 conexiones simultáneas, características que permiten la implementación del actual proyecto.

(v) **Escritura de datos:** Con los pasos anteriores, se logra la conexión del proyecto en Android a una base de datos específica que se puede acceder desde la consola en la Interfaz Web de Firebase. Para que la aplicación interactúe con los datos es necesario agregar una instancia a *Firebase Database* en la Actividad y una referencia al nodo específico. En la Fig. 2.17 se define la instancia como *database*, y la referencia *myRef* como referencia al nodo "*message*". Esto permite escribir el mensaje "*Hello, World*", mediante el método *setValue()*.

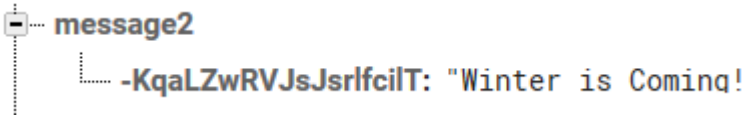
```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");
```

Fig. 2.17. Instancia y Referencia en Firebase Realtime Database.
Fuente: Documentación Firebase Realtime Database [37].

```
mFirebaseDatabase.getReference("message").setValue("Hello, World!");
```

(a)

```
mFirebaseDatabase.getReference("message2").push().setValue("Winter is Coming!");
```



(b)

Fig. 2.18. Generación automática de Keys o Claves únicas.
(a) Método setValue **(b)** Método push().setValue.
Fuente: Basado en Documentación Firebase Realtime Database [37][38].

(vi) **Firebase Key:** El método *setValue()* permite agregar un elemento directamente a un nodo, sobreescribiendo valores anteriores (Fig. 2.18.a). *Firebase* cuenta con el método *push()*, que permite agregar un elemento al nodo, con una key o clave única de identificación (Fig. 2.18.b). Esta *PushID*, es una clave de 120 bits, con 48 bits de TIMESTAMP (guarda el tiempo con precisión de milisegundos en el que se ingresó el valor), en conjunto a 72 bits aleatorios, que permite asegurar que dos datos enviados en el mismo milisegundo no generen la misma clave, e identificarlos por separado en la base de datos [38].

(vii) **ChildEventListener:** Para la lectura de elementos es posible generar un *Listener* a una referencia, que responda a cambios a los hijos del nodo referido. En la Fig. 2.19 se observan los cuatro métodos a los que puede generar respuesta. Estos métodos recuperan al objeto modificado como un objeto *DataSnapshot*. Tanto los datos enviados como leídos pueden ser datos en diferentes formatos u objetos personalizados[39].

```
ref.addChildEventListener(new ChildEventListener() {
    @Override
    public void onChildAdded(DataSnapshot dataSnapshot, String prevChildKey) {}

    @Override
    public void onChildChanged(DataSnapshot dataSnapshot, String prevChildKey) {}

    @Override
    public void onChildRemoved(DataSnapshot dataSnapshot) {}

    @Override
    public void onChildMoved(DataSnapshot dataSnapshot, String prevChildKey) {}

    @Override
    public void onCancelled(DatabaseError databaseError) {}
});
```

Fig. 2.19. Método ChildEventListener.
Fuente: Documentación Firebase, Retrieve Data [39].

```
Query query = dinosaursRef.orderByChild("height").endAt(favoriteDinoHeight).limitToLast(2);
query.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        // Data is ordered by increasing height, so we want the first entry
        DataSnapshot firstChild = dataSnapshot.getChildren().iterator().next();
        System.out.println("The dinosaur just shorter than the stegosaurus is: " + firstChild.getKey());
    }

    @Override
    public void onCancelled(DatabaseError databaseError) {
        // ...
    }
});
```

Fig. 2.20. Implementación de Querys.
Fuente: Documentación Firebase, Retrieve Data [39].

(viii) **Query:** Para la realización de consultas complejas, *Firebase* cuenta con la Clase *Query* que incorpora diferentes métodos: *endAt, equalTo, isEqual, limitToFirst, limitToLast, orderByChild, orderByKey, OrderByValue, startAt* y *toString*. Estos se asocian a una referencia para realizar la consulta, agregando los elementos recuperados en el método *onChangeData()* mediante un DataSnapshot. Por el contrario, si ningún elemento es recuperado se ejecuta el método *onCancelled()* (Fig. 2.20) [39].

(ix) **Persistencia en Disco:** *Firebase* permite habilitar funciones sin conexión. De esta forma, la aplicación seguirá funcionando ante interrupciones temporales de red, realizando consultas a la caché almacenada de forma local. La caché por defecto tiene una capacidad de 10 MB (configurable a un máximo de 100MB). Si esta llega al límite de su capacidad, utiliza una política de reemplazo LRU (Least Recently Used). En el caso de escrituras, los datos serán persistentes en el disco hasta que se conecte nuevamente. La persistencia es activada mediante una línea de código (Fig. 2.21), además permite seleccionar los nodos a mantener sincronizados para su uso offline mediante *keepSynced* (Fig. 2.21) [40].

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

(a)

```
DatabaseReference scoresRef = FirebaseDatabase.getInstance().getReference("scores");
scoresRef.keepSynced(true);
```

(b)

Fig. 2.21. Habilitación de funciones sin conexión.
(a) Habilitar persistencia en disco. (b) Habilitar sincronización de nodo específico.
Fuente: Documentación Firebase, Retrieve Data [40].

(x) **Autenticación:** Conocido también como autentificación, permite a la Base de datos comprobar la información del usuario y sus credenciales. Como se indicó en TABLA 2.2, *Firebase* ofrece una biblioteca de Autenticación. A través de esta es posible generar procesos de identificación de usuarios con diferentes niveles de acceso de edición y lectura. Dispone de una interfaz de usuario que incluye la opción de identificarse mediante email o diferentes proveedores como Google, Twitter y Facebook. Esto genera un *token de identificación* único para la sesión, que vincula a las lecturas y escrituras a la identidad del usuario. Este proceso requiere de conexión a la red. Pese a que el uso de persistencia en disco permite mantener este *token* por un tiempo, este se elimina luego de un tiempo por motivos de seguridad, deteniendo la lectura/escritura y solicitando re-autenticar al usuario [40]. Sin bien, el uso de Autenticación es recomendado por los desarrolladores para proteger los datos, para el presente trabajo se requiere de una aplicación que pueda funcionar de forma independiente a la red, por lo cual se protegerá el acceso a las aplicaciones a través de contraseña, pero no se utilizará la biblioteca de Autenticación de *Firebase*.

(xi) **Post-procesamiento:** El uso de *Firebase* permite un sencillo desarrollo de nuevas estadísticas, además de su exportación completa como archivo JSON, posteriormente analizable en planillas electrónicas como archivos CSV (valores separados por coma)[37].

2.4.4 Comunicación Inalámbrica entre aplicaciones

Como se presentó en la sección previa uno de los comentarios con respecto a la aplicación *SIEL*, fue la generación de recreos de forma automática cada cierto tiempo, sin el input del tutor. *SIPLYC* contará con una aplicación secundaria para el tutor, con la cual pueda "*activar recreos*" a la aplicación de juego. Para la implementación de esto, se utilizará comunicación inalámbrica a través de Bluetooth. Esta tecnología viene disponible en los dispositivos a utilizar, y es de sencilla implementación para Android.

A. Comunicación Bluetooth en Android

Android Studio da acceso a la funcionalidad Bluetooth mediante la API de Android Bluetooth, que permite identificar otros dispositivos, consultar por equipos sincronizados, conectarse y establecer un canal de comunicación, transferir y recibir datos entre estos. Sus principales interfaces son [41]:

(i) **BluetoothAdapter:** Permite consultar la lista de dispositivos sincronizados, puede crear una instancia de un dispositivo Bluetooth remoto y crear un servidor para escuchar comunicación de estos.

(ii) **BluetoothDevice:** Lo utiliza para solicitar e identificar una conexión con un equipo remoto, almacenando su información de nombre, dirección, estado y dirección MAC única.

(iii) **BluetoothSocket:** Permite el intercambio de datos, representa al punto de conexión.

(iv) **BluetoothServerSocket:** Representa al socket de servidor, que se encuentra a la escucha de solicitudes de conexión, y generará un *BluetoothSocket* al aceptarla que enviará al otro dispositivo.

(v) **BluetoothProfile.ServiceListener:** Identifica y responde a eventos de conexión y desconexión de dispositivos.

Para habilitar las funciones de Bluetooth, es necesario incluir los permisos en el *AndroidManifest* (Fig. 2.22.a), establecer una instancia de *BluetoothAdapter* para identificar si el equipo cuenta con la función (Fig. 2.22.b), solicitar al usuario su habilitación (Fig. 2.22.c) y posteriormente hacer visible el dispositivo para iniciar una conexión (Fig. 2.22.d).

```
<manifest ... >
  <uses-permission android:name="android.permission.BLUETOOTH" />
  ...
</manifest>
```

(a)

```
BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (mBluetoothAdapter == null) {
    // Device does not support Bluetooth
}
```

(b)

```
if (!mBluetoothAdapter.isEnabled()) {
    Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
}
```

(c)

```
Intent discoverableIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
discoverableIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
startActivity(discoverableIntent);
```

(d)

Fig. 2.22. Configuración inicial para habilitar funciones de Bluetooth
(a) Permisos de Bluetooth. **(b)** Definición de BluetoothAdapter.
(c) Solicitud para habilitar Bluetooth. **(d)** Solicitud de visibilidad de dispositivo.
Fuente: Documentación Oficial de Android Developers [41].

Para conectar dos dispositivos, es necesario identificar uno como cliente y otro como servidor. Una técnica de implementación es que ambos se inicien con un socket de servidor, de forma que cualquiera pueda realizar el intento de conexión y convertirse en cliente [41].

B. Conexión como Servidor

El dispositivo que sea servidor, debe contar con un *BluetoothServerSocker* abierto a conexiones. En primer lugar, se crea uno generando un identificador único universal (UUID) de 128 bits que identificará al servicio. Mediante el método *accept(),* se acepta solicitudes de conexión, y al realizarse una de forma exitosa se obtiene un *BluetoothSocket.* Esto genera un canal de comunicación *RFCOMM,* el cual es un protocolo de comunicación que permite solo un cliente, por lo que llama al método *close(),* para no aceptar nuevas solicitudes de conexión [41].

C. Conexión como Cliente

El dispositivo que sea cliente, utilizará un *BluetoothDevice* que identifica al servidor, para obtener el *BluetoothSocker* con el UUID único, lo cual llama al método *connet()* para iniciar la conexión. Previo a este llamado, se recomienda detener la búsqueda automática de nuevos dispositivos, pues puede generar fallos en establecer la conexión [41].

D. Manejo de la conexión

Cuando ambos dispositivos se encuentren con un *BluetoothSocket* activo, se podrá recibir datos mediante *InputStream* y transferir por *OutputStream*. Estos datos que se pueden obtener mediante los métodos *getInputStream()* y *getOutputStream()* (Fig. 2.23) [41].

```
public ConnectedThread(BluetoothSocket socket) {
    mmSocket = socket;
    InputStream tmpIn = null;
    OutputStream tmpOut = null;

    // Get the input and output streams, using temp objects because
    // member streams are final
    try {
        tmpIn = socket.getInputStream();
        tmpOut = socket.getOutputStream();
    } catch (IOException e) { }

    mmInStream = tmpIn;
    mmOutStream = tmpOut;
}
```

Fig. 2.23. Obtención de datos de InputStream y OutputStream
Fuente: Basado en Documentación Oficial de Android Developers [41].

```
public void run() {
    byte[] buffer = new byte[1024];  // buffer store for the stream
    int bytes; // bytes returned from read()

    // Keep listening to the InputStream until an exception occurs
    while (true) {
        try {
            // Read from the InputStream
            bytes = mmInStream.read(buffer);
            // Send the obtained bytes to the UI activity
            mHandler.obtainMessage(MESSAGE_READ, bytes, -1, buffer)
                    .sendToTarget();
        } catch (IOException e) {
            break;
        }
    }
}
```

(a)

```
/* Call this from the main activity to send data to the remote device */
public void write(byte[] bytes) {
    try {
        mmOutStream.write(bytes);
    } catch (IOException e) { }
}
```

(b)

```
/* Call this from the main activity to shutdown the connection */
public void cancel() {
    try {
        mmSocket.close();
    } catch (IOException e) { }
```

(c)

Fig. 2.24. Configuración inicial para habilitar funciones de Bluetooth
(a) Lectura de datos desde stream. **(b)** Escritura en stream. **(c)** Finaliza conexión.
Fuente: Documentación Oficial de Android Developers [41].

La escritura en el flujo de salida es realizada por el método *write(byte[])* y la lectura mediante *read(byte[]).* En el ejemplo de la Fig. 2.24.a los datos leídos son enviados a la Actividad de interfaz de usuario a través de un *Handler*, encargado de responder a diferentes estados de la comunicación Bluetooth, en esta caso “MESSAGE_READ”. Se define una función *write* (Fig. 2.24.b) que permite obtener la información a enviar desde otra actividad.

2.5. Aplicación Principal SIPLYC

El proyecto SIPLYC (actualmente en desarrollo) cuenta con un prototipo funcional de la aplicación de juego o estimulación, sobre la cual se deben implementar las características de control y base de datos. Esta aplicación es desarrollada por un ingeniero contratado por el proyecto para generar el sistema final, por lo que se procuró que las funcionalidades y aplicación secundaria implementadas en el presente trabajo, se acoplen correctamente sin afectar su lógica de funcionamiento.

2.5.1 Descripción General

Esta aplicación es la que cumple el rol de software de estimulación. Permite intercambiar entre tres módulos de juego, que se describirán a continuación.

A. Módulo de Palabras

Denominado también "*Módulo de Lenguaje Comprensivo*", esta actividad se enfoca en enseñar al niño palabras de uso infrecuente o de tipo conceptual. La actividad se encuentra compuesta por tres elementos, en un *LinearLayout* horizontal. Un esquema del funcionamiento se encuentra presente en la Fig. 2.25, donde se observa que el objeto central corresponde a un video de una educadora, que guía y evalúa la actividad de forma automática en una actividad de entrenamiento de palabra infrecuente. En el caso de enseñar conceptos ("*encima/debajo", "cerca/lejos"*), se mostrará imágenes que representen estos conceptos ("pelota *encima/debajo* de la silla"), para evaluarlos bajo la misma dinámica. La actividad presentará palabras de forma aleatoria, y debe generar recreos de forma automática o recibirlos desde la aplicación de Tutor [15].

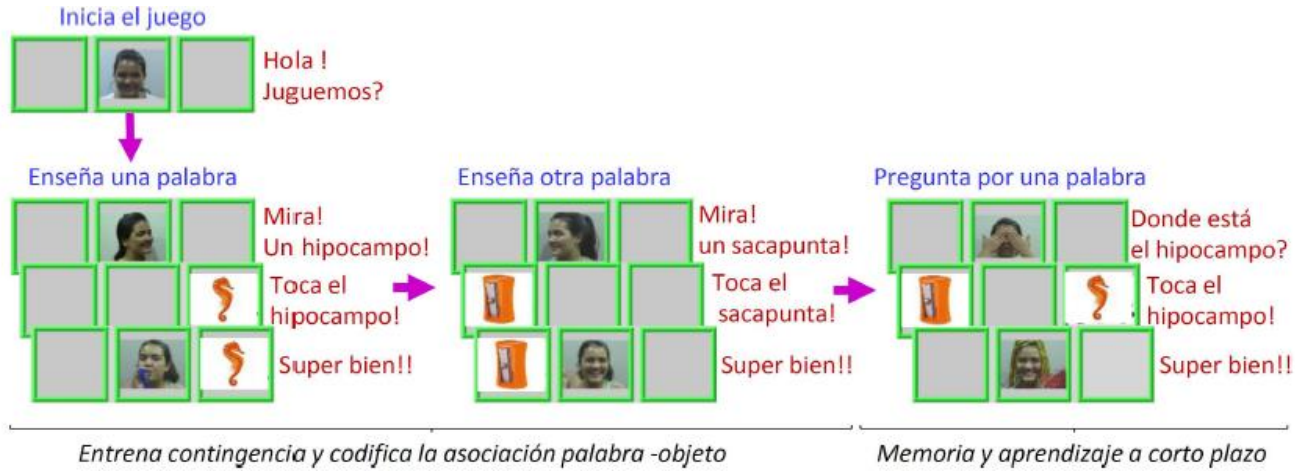


Fig. 2.25. Esquema de funcionamiento en Módulo de Palabras.
Fuente. Datos entregados en Formulación Proyecto SIPLYC [15].

B. Módulo del Habla

Esta actividad tendrá un comportamiento similar al módulo anterior, pero, en este caso se solicitará "llamar" al objeto a través del habla. Cuando se presentó originalmente en el proyecto, se esperaba realizar un procesamiento del audio para evaluar esta actividad de forma automática. Sin embargo, esto se descartó para dar preferencia a una evaluación de parte del tutor, quien podrá evaluar correctamente (o pedir una repetición desde su aplicación) y entregar retroalimentación directa al niño. Este cambio, en palabras de los investigadores, busca promover la estimulación del habla del niño, el cuál recibirá una respuesta positiva por intentar decir las diferentes palabras, pese a que un sistema de reconocimiento automático podría reconocer como objetivamente incorrectas.

C. Módulo de Letras

Denominado también "*Módulo de Conciencia Fonológica*", con una estructura similar al módulo de palabras, solicitará indicar al niño sonido de letras en sílabas o palabras cortas. Por ejemplo, indicar el sonido "*mm*" en la sílaba "*MA*" [15].

2.5.2 Implementación de Aplicación Principal

Al momento de comenzar este trabajo, se contaba con un prototipo funcional de la aplicación principal. Ésta se probó en un dispositivo Tablet *Samsung Galaxy Tab (SM-T350)* de pantalla de 10,1" (pulgadas), con resolución de 1280x800 píxeles y sistema operativo Android 5.0.2.

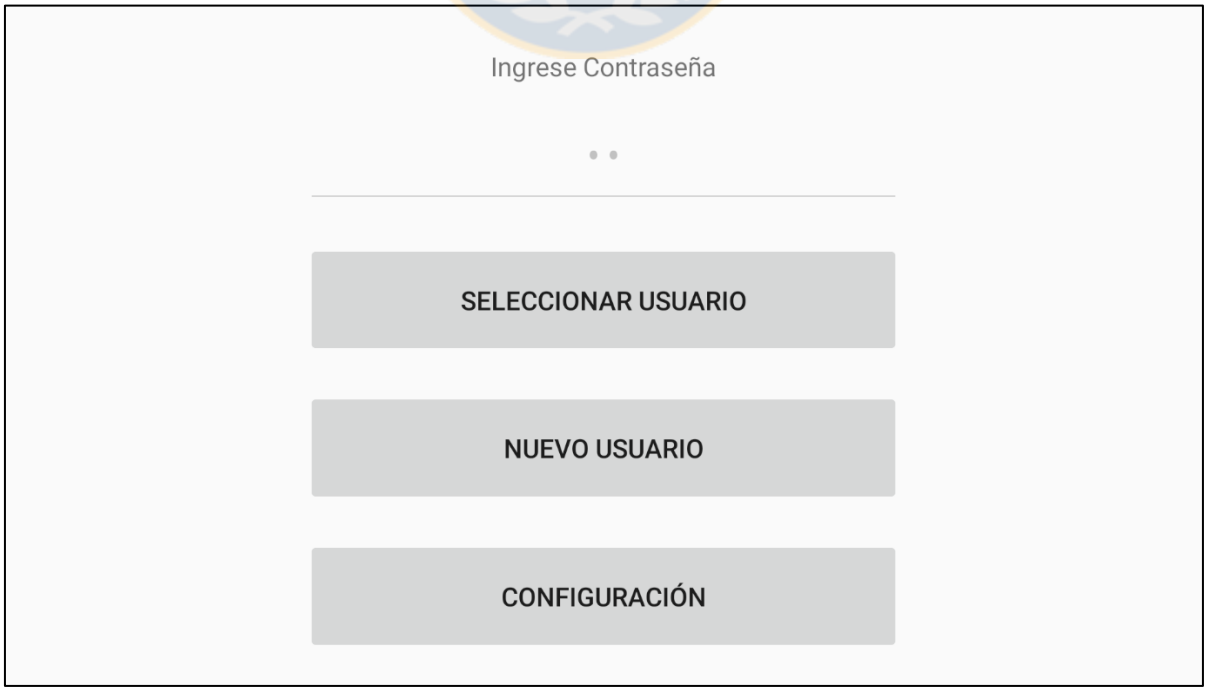


Fig. 2.26. Captura de *MainActivity* de Aplicación de Juego desde Tablet.

Esta versión cuenta con las características necesarias para la conexión inalámbrica mediante un ejemplo tradicional de ChatBluetooth, mediante el uso de un Handler (descrito previamente).

A continuación, se describirán las diferentes actividades de la aplicación.

A. Actividad de Inicio o MainActivity

Su Actividad de Inicio o *MainActivity* (Fig. 2.26) corresponde a una pantalla con tres botones (Button View) que se encuentran bloqueados por una contraseña por defecto.

(i) **Seleccionar Usuario:** Permite seleccionar un usuario existente.

(ii) **Nuevo Usuario:** Permite crear un nuevo usuario.

(iii) **Configuración:** Permite modificar directorio de datos almacenados, activar Bluetooth y permitir la conexión con el dispositivo con la app secundaria.

B. Nuevo Usuario

En esta actividad se crea un nuevo usuario con datos de Nombre, Apellido, ID (rut), sexo y fecha de nacimiento. De forma local se crea un directorio en el dispositivo con el ID del usuario, en el cual se guardarán grabaciones de las evaluaciones del Habla, además de interacciones con la aplicación que no son relevantes para las estadísticas actuales, pero pueden ser de relevancia para los investigadores en el futuro. Además, se habilitará el directorio para guardar de forma local videos personalizados de felicitación y recreos por parte de los padres del usuario. Se puede iniciar la *Actividad de Ejercicio (o Juego)* directamente luego de ingresar al nuevo usuario.

Fig. 2.27. Captura de Actividad Ingreso de Nuevo Usuario

C. Seleccionar Usuario

Esta actividad permite la selección de un usuario a partir de una lista de directorios. Estos se encuentran identificados por su ID. Como se observa en la Fig. 2.28, permite ingresar a una actividad de entrenamiento, que muestra al usuario la dinámica de juego o comenzar directamente con el ejercicio (Fig. 2.29). Además, se encuentra la opción de "activar la vocalización" que inicia la evaluación de "Módulo del Habla". Como esta evaluación requiere de la interacción con el dispositivo del Tutor, si no se identifica la conexión Bluetooth entre ambos, no permite realizar este ejercicio.

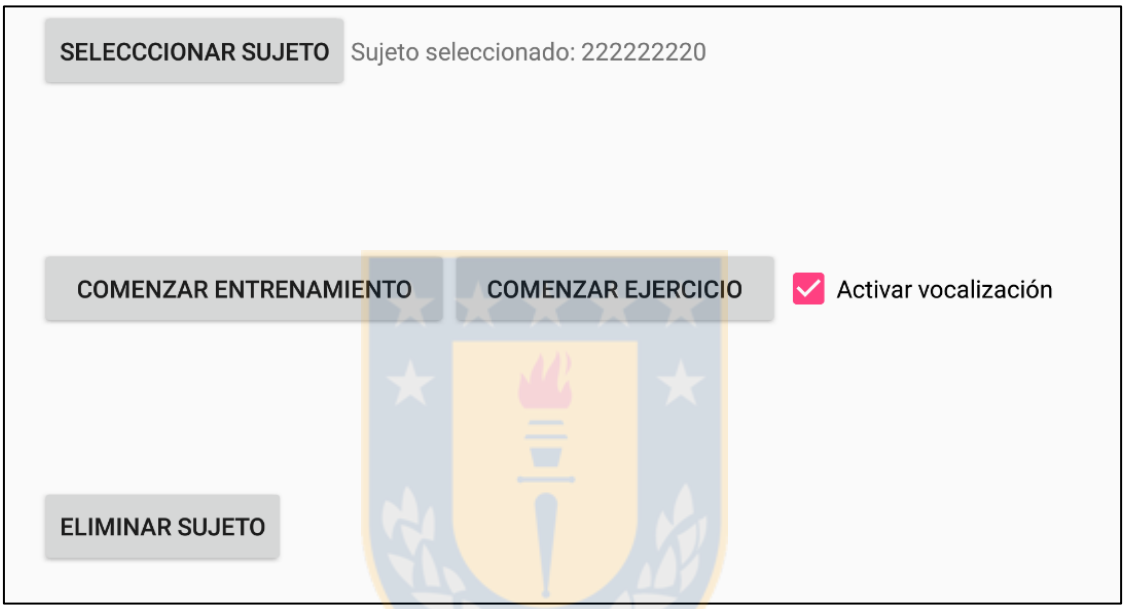


Fig. 2.28. Actividad de Selección de Usuario.

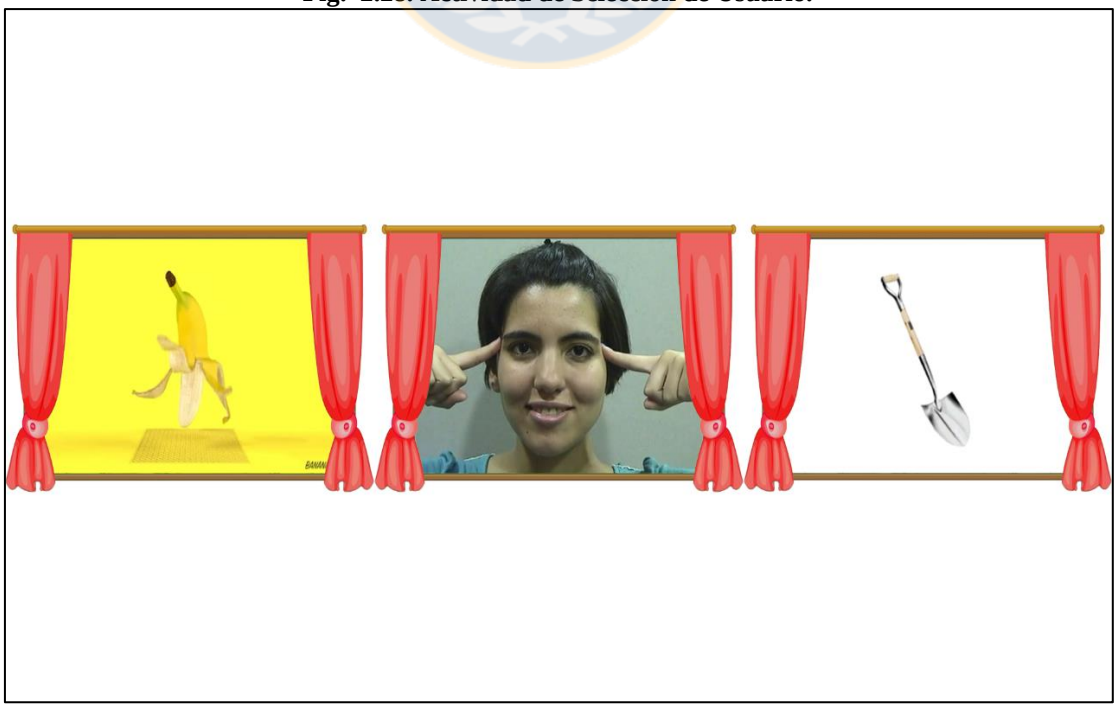


Fig. 2.29. Actividad de Juego.

Fig. 2.30. Actividad de Configuración.

D. Configuración y Conexión Bluetooth

La actividad de configuración, permite elegir el lugar de almacenamiento en el que se generan los directorios de los usuarios. Además, será la encargada de realizar la conexión inalámbrica vía Bluetooth con la aplicación de Tutor. Como se observa en la Fig. 2.30, permite activar la visibilidad del dispositivo e iniciar conexión con equipos en el alcance o previamente emparejados.

Es posible modificar el comportamiento de los ejercicios, variando la frecuencia de recreos automáticos (o deshabilitarlos) y modificar el tiempo de espera de respuesta para evaluar una palabra como omitida. En el caso de Fig. 2.30, se observan los valores por defecto de recreos cada tres pares de palabras y tiempo de espera de dos segundos.

En esta actividad se incluirán las opciones de personalización para el ingreso de videos de recreo y felicitación por parte de los padres, pero hasta este momento, la función no se encuentra habilitada.

E. Recreos e interrupciones de Juego

Esta actividad es iniciada de forma automática de acuerdo a la frecuencia definida en la *Actividad de Configuración*. De igual forma, puede ser iniciada en respuesta a una solicitud de recreo enviada por la aplicación del Tutor a través de la conexión Bluetooth.

Un ejemplo de recreo se puede observar en la Fig. 2.31.Los recreos se reproducen en forma aleatoria y corresponden a videos de niños y bebés. De esta forma, los usuarios evaluados pueden sentirse identificados.

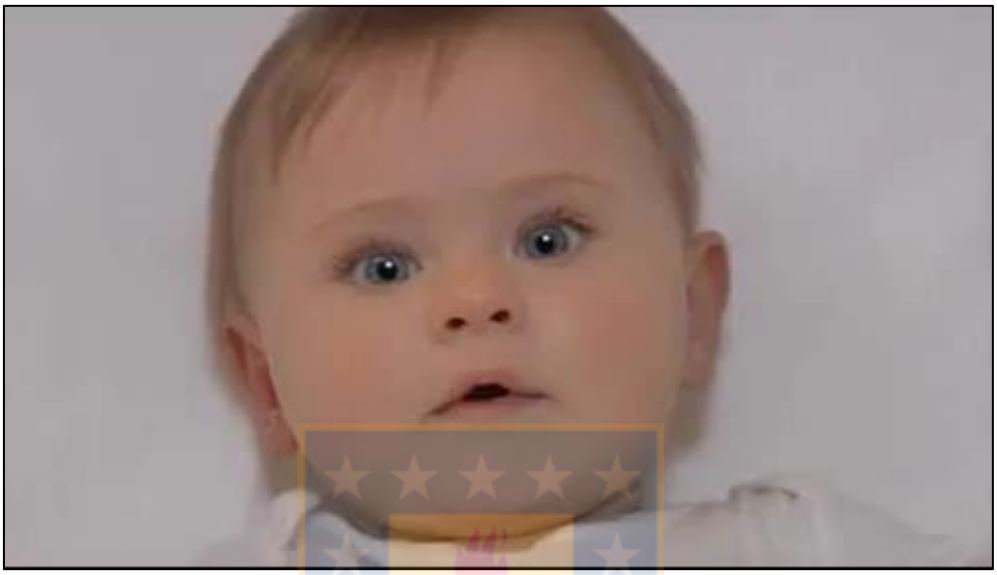


Fig. 2.31. Captura de pantalla de Recreo.

F. Actividad de Ejercicio o Juego

Corresponde a la actividad principal de la aplicación y que debe responder a la **lógica de juego** propuesta por los investigadores, para esto la evaluación de un par de objetos se divide en cuatro etapas: "Learning Objeto 1", "Learning Objeto 2", "Testing de Objeto X" y "Testing de Vocalización", de acuerdo al diagrama de flujo presentado en la Fig. 2.32.

Cada sujeto será evaluado entre 8 a 10 sesiones. Cada sesión cuenta con pares de palabras (Módulo de Vocabulario) y letras (Módulo de Conciencia Fonológica) definidas, que se irán alternando automáticamente entre ambas evaluaciones durante la sesión.

Etapas de la Lógica de Evaluación:

(i) **Learning Objeto 1:** En esta etapa se presenta el primer objeto y se solicita que lo seleccione. Se determina el intervalo de tiempo entre la presentación del objeto y el momento en que el usuario lo toca, mediante un *Timer* y un *OnTouchListener()* asociado a la *view*. Este evento táctil sólo es atendido posterior a la finalización del video que solicita al objeto, de esta forma se entrena al usuario para una "*respuesta contingente*".

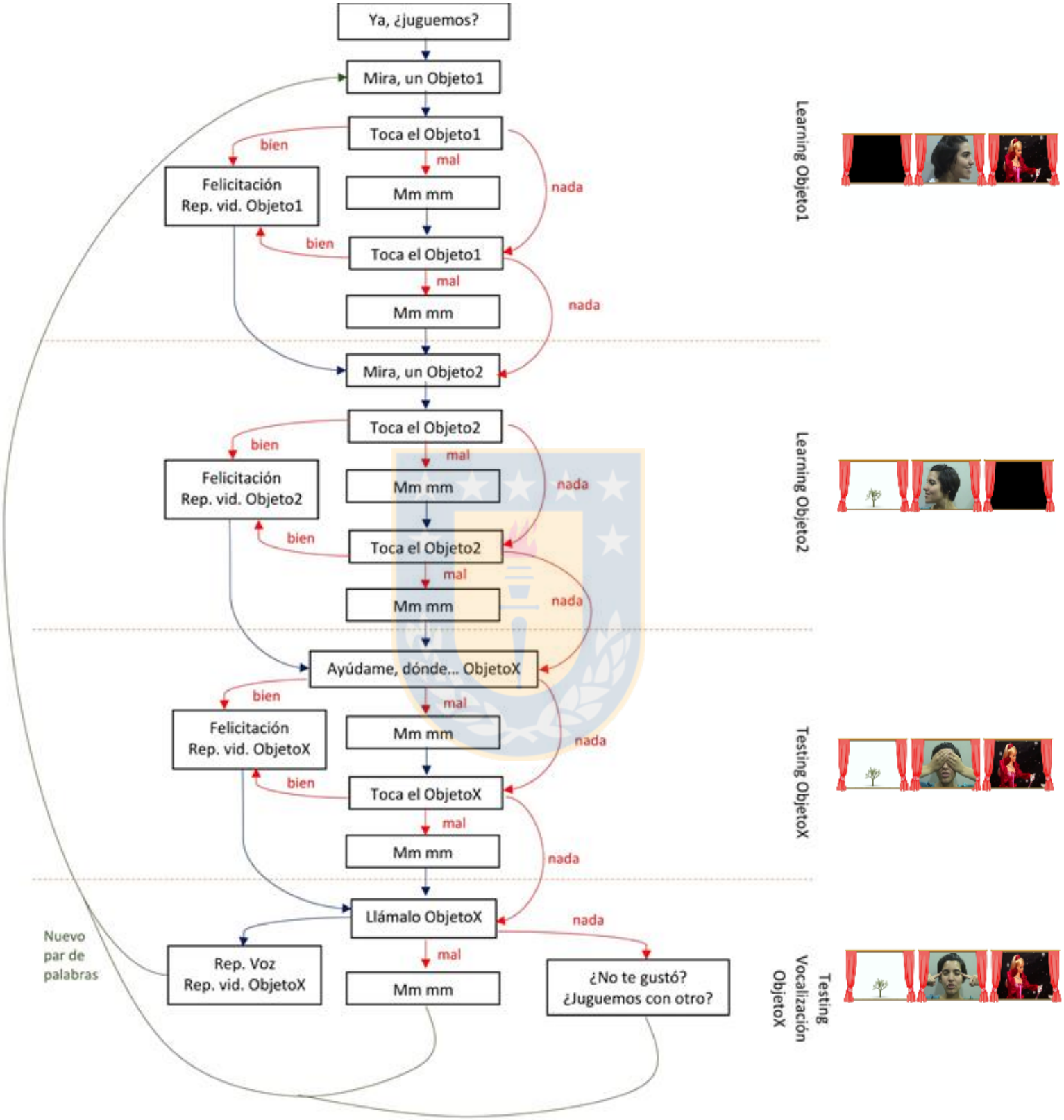


Fig. 2.32. Lógica de evaluación para vocalización.

OBS: Flujo en rojo indica evaluaciones que generan un efecto en la actividad, ya sea *correcta* (mediante un video de felicitaciones), *incorrecta* (indica error, solicita nuevamente una vez y la segunda vez pasa a la siguiente etapa) u *omitida* cuando no hay respuesta del usuario durante el tiempo de espera (solicita de nuevo una vez, y la segunda vez pasa a la siguiente etapa).

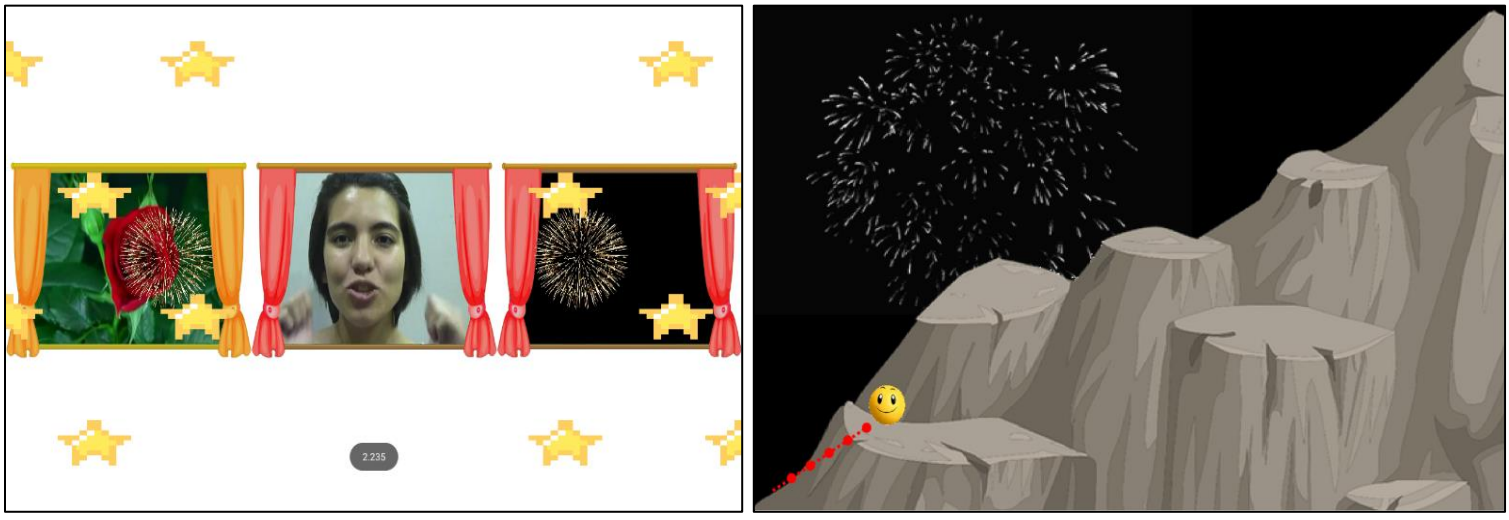
(ii) **Learning Objeto 2:** Se presenta el segundo objeto, en lógica análoga a (i).

(iii) **Testing Objeto X:** Se selecciona uno de los dos objetos y solicita al usuario que indique el correcto. Si la vocalización está desactivada, se regresa a (i) con el siguiente par de objetos a evaluar.

(iv) **Testing de Vocalización:** Sólo se realiza si la vocalización está activada. En esta etapa, se le solicita al usuario "llamar" al objeto evaluado en (iii). El audio con la respuesta del usuario es grabada y guardada de forma local automáticamente. El usuario recibirá una respuesta de acuerdo a la evaluación recibida desde la aplicación del tutor, que al igual que en las anteriores, puede ser evaluada como "correcta", "incorrecta" u "omitida".

Como se observó anteriormente, las etapas (i), (ii) y (iii) son realizadas tanto para la evaluación de palabras como de letras, mientras que la etapa (iv) solo se realiza si la vocalización fue activada. Esta última estará disponible solo para algunos usuarios, seleccionados por el equipo de investigación de acuerdo a sus edades. El objeto a evaluar y el orden en el que se presentarán los pares de objetos estará definido por sesión.

Una evaluación positiva le muestra al usuario un video de felicitaciones (Fig. 2.33.a), y si este posee al menos una etapa (de las cuatro anteriormente descritas) evaluada correctamente, le mostrará una actividad donde un avatar irá escalando una montaña (Fig. 2.33.b), para representar un nivel superado. Posterior a esto, se presentará el siguiente par de objetos, repitiendo la lógica descrita hasta terminar la sesión.



(a) (b)

Fig. 2.33. Finalización de una evaluación
(a) Felicitaciones por evaluación correcta. **(b)** Actividad de superación.

2.6. Discusión y Conclusiones

Pese a que el presente trabajo se enfoca en un grupo etario entre los 2 y 3 años y medio, el proceso de desarrollo del lenguaje es un fenómeno complejo que abarca diferentes sub-procesos y donde su principal forma de apoyo es a través de la estimulación e interacción con otras personas.

El estudio que se realizará con la SIPLYC se aplicará a niños en estados de vulnerabilidad, por lo que, el diseño debe evitar las distracciones y ofrecer un ambiente agradable para que esta herramienta logre su fin pedagógico.

La experiencia previa de los investigadores con la implementación del SIEL les permitió identificar nuevas necesidades para el desarrollo de la herramienta de aprendizaje a utilizar en el actual proyecto. Se requiere de una aplicación que permita la evaluación del Habla, que cuente con una versión secundaria que le permita a un tutor interactuar en tiempo real para evaluar y enviar recreos de forma contingente. Como se espera contar con múltiples tabletas para realización del estudio, se requerirá de una Base de datos para almacenar los datos de todos los sujetos evaluados, para la generación de estadísticas y su posterior análisis.

Para la implementación de la base de datos, la documentación investigada sobre *SyncAdapter* determinó que, pese a utilizar sistemas bien documentados como *MySQL* y *SQlite*, su implementación requiere de un servidor web, con manejo de consultas por *PHP*. Este sistema presentaba la ventaja del manejo automático de sincronizaciones de forma transparente al usuario, pero como desventaja, presentaba problemas para sincronizar múltiples tablas de contenido. Al considerar que el sistema debe almacenar tablas con datos de usuarios, evaluaciones, progreso y estadísticas, la arquitectura no se ajustaba a los requerimientos. En respuesta a lo anterior, se decidió implementar una base de datos mediante *Firebase Realtime Database*, debido a sus características que facilitan el desarrollo con el manejo de múltiples clientes, sincronización en línea y el funcionamiento en modo offline (sin conexión a internet) mediante la capacidad de persistencia de datos.

Se describió el diseño general y desarrollo actual de la Aplicación Principal de Estimulación (o de Juego), desarrollada por el ingeniero parte del proyecto. Se revisaron sus características en forma general y se analizó la lógica de juego implementada. Ésta implementa una conexión inalámbrica a través del ejemplo tradicional de ChatBluetooth, que debe responder a los mensajes desde la Aplicación Secundaria a desarrollar. Por tratarse de un software en desarrollo, es importante que las funcionalidades implementadas se adapten correctamente y no modifiquen su lógica.

Capítulo 3. Diseño de la Aplicación Secundaria

3.1. Introducción

En esta sección se presentan las consideraciones de diseño de la aplicación secundaria desarrollada. Se describirán módulos principales y funcionalidades.

3.2. Aplicación Secundaria o de Tutor

3.2.1 Descripción General

Esta aplicación proporcionará herramientas para el control de las actividades que requieren evaluación del tutor, además de dar la opción de generar recreos. Debe tener acceso a la base de datos para generar estadísticas y contar con al menos dos módulos especializados. Esta aplicación es generada en su totalidad en el presente trabajo, y debe ser capaz de integrarse a la aplicación de juego.

A. Módulo de Evaluación del Habla

Este módulo debe funcionar en paralelo a la actividad del niño. Se espera que el tutor sea capaz de evaluar positivamente la palabra (generando mensaje de felicitaciones), evaluar negativamente (pasando a la siguiente palabra) o evaluar como omitida.

Se espera que pueda detener la actividad tanto momentáneamente (recreo) o totalmente (terminando con la actividad de juego).

B. Módulo de Estadísticas

Este módulo debe acceder a los valores almacenados de cada usuario y permitir su visualización. Se observa una captura de las estadísticas obtenidas del software *SIEL* (ver Fig. 3.1). Se solicitó obtener estadísticas a partir de estimaciones de los aciertos y tiempos de respuesta promedio [15]. A partir del análisis de otras experiencias, se considera interesante incluir un *Perfil* que almacene las palabras específicas de cada sesión, como lo hace el app "*Articulation Station Español*" [21], que guarda las palabras correctas, aproximadas e incorrectas, y permite repetir estas últimas o el módulo completo.

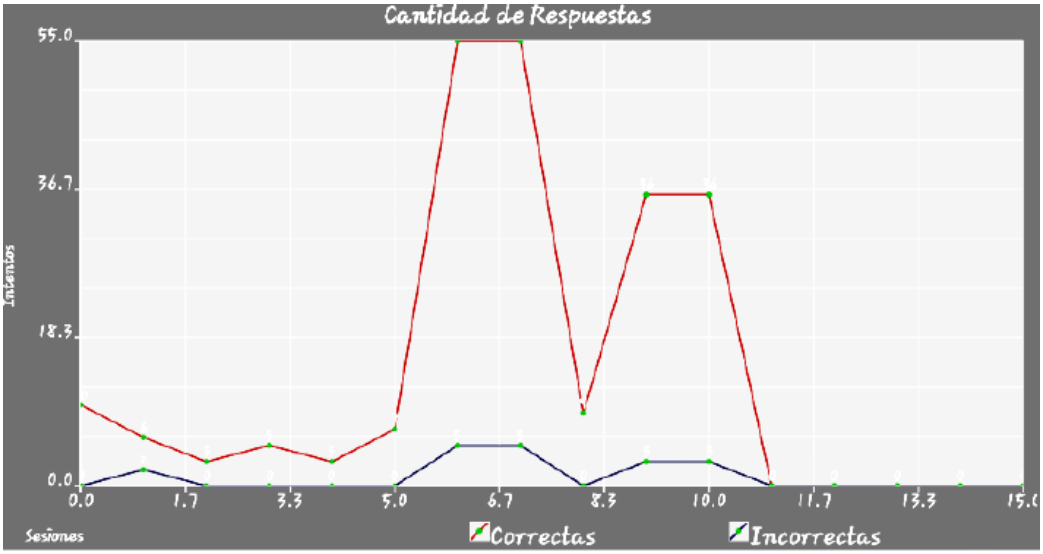


Fig. 3.1. Captura de pantalla de estadísticas en SIEL
Fuente. Cid R. Informe de Memoria de Título [26]

Fig. 3.2. Captura de pantalla de Perfil de estudiantes en *Articutation Station Español*.
Fuente. Sitio web oficial de la aplicación [21].

3.2.2 Interacción entre Aplicaciones

La aplicación de juego debe responder a las evaluaciones y recreos enviados desde la aplicación del tutor. Debe, además, guardar en la base de datos los eventos de importancia para futuras

estadísticas, tanto evaluaciones y recreos generados de forma automática por la interacción del niño, como los eventos provocados desde la aplicación secundaria.

La comunicación entre ambas aplicaciones se realizará a través de Bluetooth y se debe adaptar a la conexión implementada en la Aplicación Principal.

Debido a las características descritas en el capítulo anterior, se descartó la implementación de una base de datos con arquitectura basada en SyncAdapter, y se utilizará Firebase.

3.3. Diseño de Aplicación del Tutor

Previo a la implementación, se diseñó un prototipo no funcional mediante *MockingBot* [42], una interfaz Web que permite el desarrollo de un bosquejo de la futura aplicación con el fin de distribuir elementos en pantalla y organizar el flujo de actividades. La aplicación funciona como un boceto, que permite analizar el comportamiento sin la necesidad de programar, por lo que sirve tanto de guía para la posterior implementación, como para realizar pruebas de concepto de forma rápida.

Se propone el flujo de trabajo, de acuerdo a la Fig. 3.3. Se inicia con una actividad de Login (1), que entregue protección a los datos mediante el ingreso con el uso de usuario y contraseña. Una actividad de menú principal (2) que permita establecer la conexión Bluetooth (3), ingresar a la actividad de evaluación (5) y acceder al módulo de estadísticas, que entregaría una lista de los niños evaluados (6), una estadística general de cada uno (7) y finalmente un detalle de las palabras evaluadas por sesión (8).

Fig. 3.3. Diseño de flujo de trabajo de Aplicación de Tutor.
Fuente. Sitio web oficial de la aplicación [42]

Fig. 3.4. Diseño de Actividad de Evaluación
Fuente. Prototipo diseñado en Interfaz Web *MockingBot* [42].

Para la actividad de evaluación (Ver Fig. 3.4), se diseñó a modo de pantalla completa con visión de la palabra evaluada y con botones para evaluar positiva o negativamente, además de solicitar una repetición. Así lo hace automáticamente la aplicación de juego en el Módulo de Vocabulario, donde la lógica implementada solicita una repetición de una palabra inicialmente evaluada negativamente. En el caso de la visualización de las estadísticas, se diseñó una actividad que entrega la información general del rendimiento del niño a lo largo de las sesiones (Fig. 3.4.a), variando entre los tres módulos (Vocabulario, Habla y Letras), con sus respectivas estadísticas (Palabras Aprendidas por Sesión, Palabras Correctas/Incorrectas y Tiempo de Respuesta promedio). Se incluyó una actividad para analizar el detalle por sesión, con información de palabras, su evaluación y su tiempo de respuesta (Fig. 3.4.b). Ambas actividades serían independientes para el perfil de cada niño.

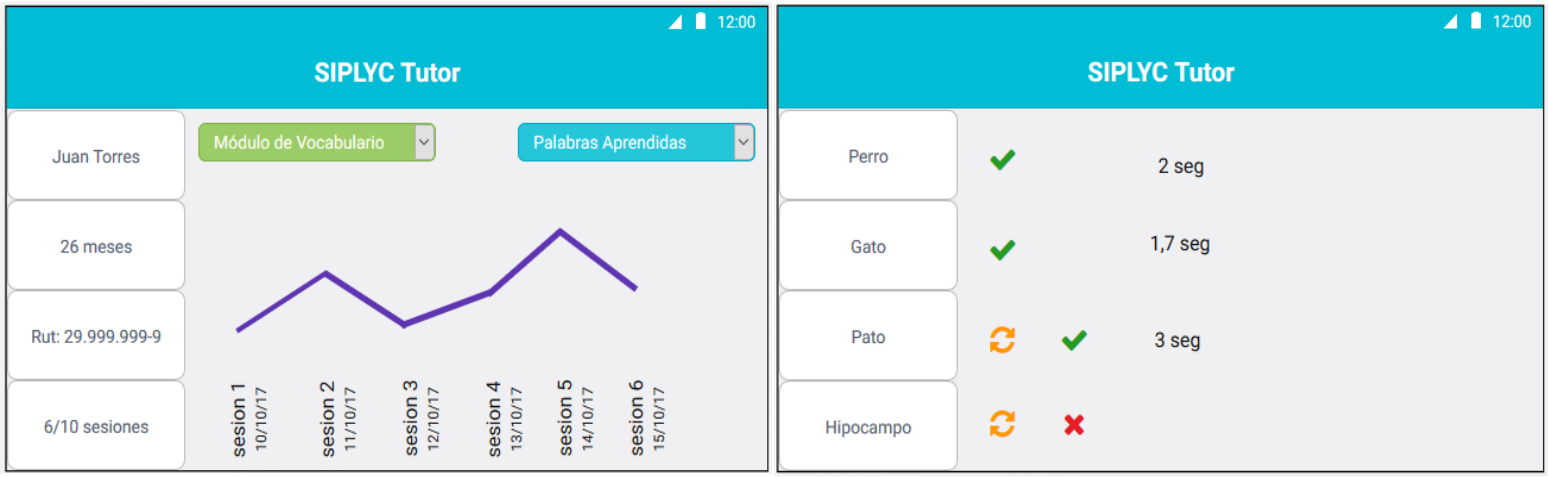


Fig. 3.5. Diseño de Actividad de Estadísticas
(a) Estadísticas generales del usuario **(b)** Evaluaciones por sesión.
Fuente. Prototipo diseñado en Interfaz Web *MockingBot* [42].

3.4. Discusión y Conclusiones

En el presente capítulo se definen las funcionalidades de los diferentes elementos de la aplicación, tanto de juego como del tutor. Es importante incluir en el desarrollo la información analizada en capítulos anteriores, como las indicaciones al software *SIEL*, las necesidades planteadas por los investigadores que utilizarán el *SIPLYC*, además de los aportes de parte de la literatura y experiencia previa de softwares actualmente disponibles en el mercado. Dos aspectos son claves: la aplicación de juego debe mantener la atención del niño evaluado y registrar toda la información necesaria para generar estadísticas de aprendizaje. Por su parte, la aplicación del tutor debe permitir un fácil y correcto control de las actividades de la aplicación del niño, además de la visualización de los datos estadísticos.

Capítulo 4. Desarrollo e Implementación

4.1. Introducción

En el presente capítulo se describe el desarrollo de la aplicación de tutor a través de las diferentes actividades que permiten la lectura, generación y visualización de estadísticas. Se describe, a su vez, las funcionalidades implementadas en la aplicación principal.

Se utilizarán los siguientes conceptos o abstracciones: se llamará "*Usuario*" al sujeto evaluado y "*Tutor*" al evaluador. Se llamará "*Actividad*" a cada "pantalla" dentro de la aplicación y "View" a los elementos dentro de la misma (para mayor detalle revisar *2.4.Consideraciones de Software*).

4.2. Desarrollo de la Aplicación

4.2.1 Implementación y Comportamiento de Comunicación Bluetooth

Existen diferentes demostraciones funcionales para la implementación de las Bluetooth API. Para el trabajo actual, se utilizó un ejemplo tradicional de *Chat de Bluetooth*, que envía mensajes entre las aplicaciones, recogidas por un *Handler* y generando cambios en la interfaz de usuario.

Se definieron los siguientes mensajes enviados por la aplicación de tutor (Ver Fig. 4.1):

(i) **"start":** Inicia un recreo, si la actividad de juego está activa.

(ii) **"finish":** Finaliza el recreo, si hay uno activo.

(iii) **"correct":** Evalúa correctamente una evaluación del Habla.

(iv) **"incorrect":** Evalúa incorrectamente una evaluación del Habla.

(v) **"pause":** Permite pausar la actividad de Juego de una sesión.

(vi) **"unpause":** Permite retomar la actividad de Juego, en el par de palabras que se pausó.

De forma análoga, se implementó un *Handler* para responder a mensajes enviados desde la aplicación de Juego a la del Tutor, con los siguientes mensajes:

(i) **"palabra: X":** Se revisa si el mensaje contiene la expresión "*palabra:*", y recibe de esta forma la palabra que se está evaluando y puede ser visualizada por el Tutor.

(ii) **"patient: X":** Se revisa si el mensaje contiene la expresión "*patient:*", y recibe el nombre del evaluado, el cual se visualiza en la interfaz del tutor.

```
// The Handler that gets information back from the BluetoothChatService
private final Handler mHandler = handleMessage(msg) → {
        switch (msg.what) {
            case MESSAGE_WRITE:
                byte[] writeBuf = (byte[]) msg.obj;
                // construct a string from the buffer
                String writeMessage = new String(writeBuf);
                break;
            case MESSAGE_READ:
                byte[] readBuf = (byte[]) msg.obj;
                // construct a string from the valid bytes in the buffer
                String readMessage = new String(readBuf, 0, msg.arg1);
                if (readMessage.equals("start")& !BreakActivity.getState() & ExcersiceActivity.getState()) {
                    Intent intent = new Intent(getApplicationContext(), BreakActivity.class);
                    startActivity(intent);
                }else if(readMessage.equals("finish")& BreakActivity.getState()){
                    BreakActivity.BrkAct.finish();
                }else if(readMessage.equals("correct") & ExcersiceActivity.getRecordingState()){
                    ExcersiceActivity.VocalizingRight();
                }else if(readMessage.equals("incorrect") & ExcersiceActivity.getRecordingState()){
                    ExcersiceActivity.VocalizingWrong();
                }else if(readMessage.equals("skip") & ExcersiceActivity.getRecordingState()){
                    ExcersiceActivity.VocalizingSkip();
                }
```

Fig. 4.1. Handler implementado para responder a lectura/escritura de datos a través de Bluetooth.

4.2.2 Interacción con la Base de datos

En el capítulo *"2.5.2F. Actividad de Ejercicio o Juego*", se describe el proceso de evaluación. Esta lógica no debe alterarse, por lo que la interacción con la base de datos debe realizarse en el segundo plano de forma imperceptible.

Como se describió previamente, *Firebase Realtime Database* funciona en base a un árbol de datos, con nodos padres e hijos. Para la escritura y lectura, se implementaron *Clases* de JAVA. Mediante el constructor de la *clase* es posible generar una instancia para escribir y guardar un objeto, modificar sus variables mediante el método *set* y leerlas con el método *get*.

Se describirá de forma general las Clases implementadas para guardar los datos almacenados en *Firebase* y su posterior uso para la generación y visualización de estadísticas.

A. Patient

La clase *Patient*, es creada para guardar los datos del usuario evaluado. Ésta es inicializada en la actividad de "Nuevo Usuario" de la aplicación principal. Se guardan:

(i) **Variables de texto (String):** *Nombre, Apellido, sexo* e *ID*, que corresponde al rut.

(ii) **Variables enteras (Integer):** *dd* (día), *mm* (mes), *yy* (año) de nacimiento, *n_sessions* (indica el número de sesiones realizadas).

(iii) **Variables booleanas (Boolean):** *retired* y *complete*, son variables lógicas para indicar si el usuario se retiró de la investigación o si completó todas las sesiones.

(iv) **Variables var:** *date_entered* y *date_last_time*, que corresponden a valores ingresados por el método "*ServerValue.TIMESTAMP*", que permite guardar una fecha de acuerdo al servidor, y permitirá guardar el día de ingreso de paciente a la Base de datos, y la última instancia de sesión.

En la Fig. 4.2, se observa un ejemplo de un objeto de esta Clase guardado en el nodo "*patients*" de la base de datos implementada. Se puede observar que las variables de clase definidas como *null* (sin valor), no son guardadas en el nodo creado. El objeto *Patient* creado, es agregado a la Base de datos, previo a una *Query* para identificar si el usuario ya existe en la misma.

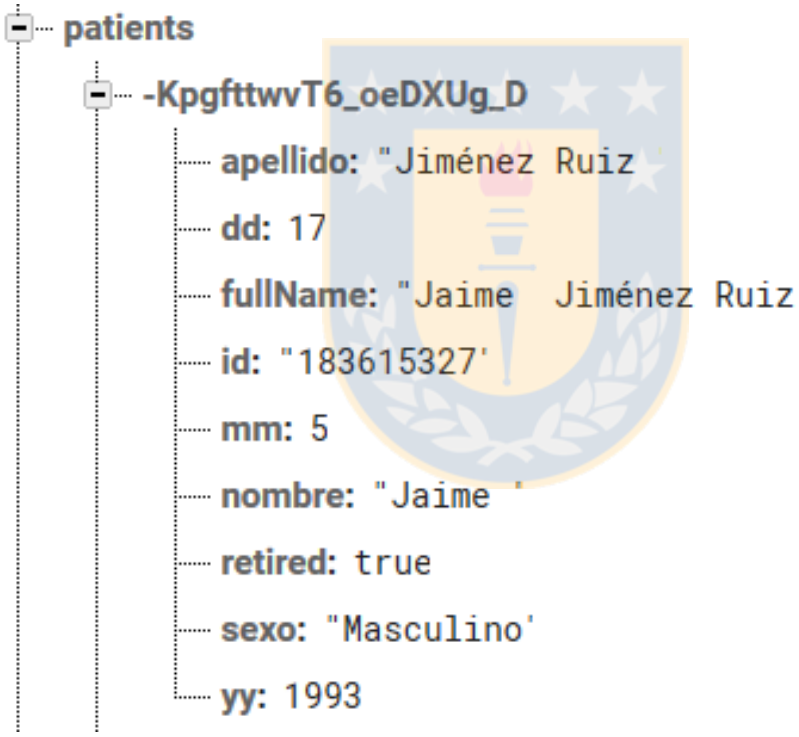


Fig. 4.2. Ejemplo de instancia a Clase Patient almacenada en Firebase.

B. Tutor

La Clase Tutor, permite guardar los datos del tutor. Estos son ingresados en la aplicación secundaria y le permiten identificarse.

(i) **Variables de texto (String):** *nombre, apellido, mail, rut, password.*

(ii) **Variables var:** *date_entered,* registra el día de ingreso del Tutor a la Base de datos.

C. Session

La Clase *Session*, permite almacenar los datos de cada sesión realizada en alguno de los dispositivos. Esta clase es creada en la actividad de juego, guardando datos de la fecha y usuario. Previo a su creación, se realiza una Query a Firebase, para identificar si ésta fue definida previamente. Esta clase es de importancia para identificar la cantidad de sesiones de cada usuario y para generar las futuras estadísticas.

(i) **Variables de texto (String):** *id_SESSION* (identifica a la sesión) *date_SESSION* (guarda string de la fecha), *id_PATIENT* (guarda rut del usuario evaluado), *id_TUTOR* (guarda rut del Tutor que evalua la sesión).

(ii) **Variables booleanas (Boolean):** *stats_OK*, esta variable se inicializa con valor *false* al crear una instancia a la clase, tomará un valor *true* cuando las estadísticas sean calculadas y guardadas correctamente en la base de datos.

D. Eventos

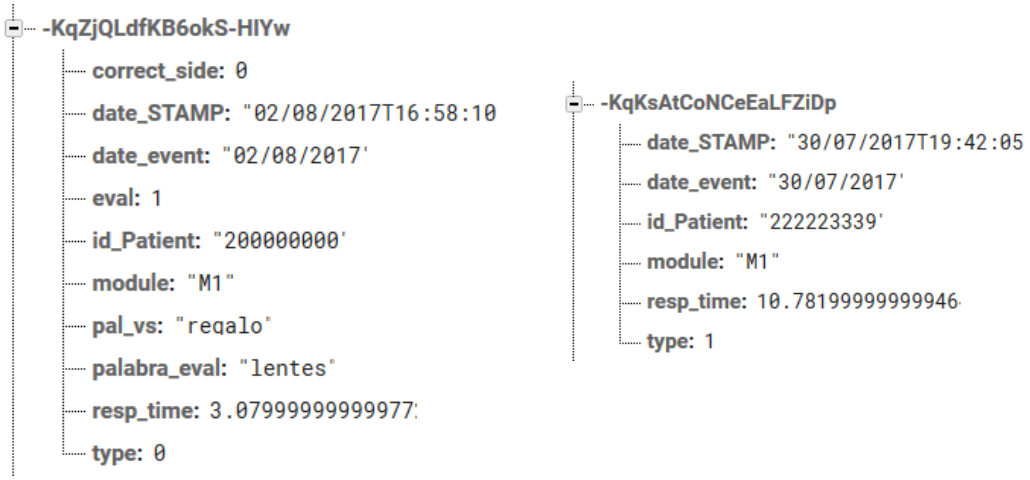
La clase *Eventos* es la responsable de guardar los eventos de importancia para la generación de estadísticas. Guarda evaluaciones o recreos y cuenta con las siguientes variables:

(i) **Variables enteras (Integer):** *Type* (identifica el tipo de evento como evaluación “0”, recreo “1” u otro como pausas “2”), *Eval* (identifica el *tipo de evaluación* como omitida “0”, correcta “1”, incorrecta “2” o correcta al segundo intento “3”, o el *tipo de recreo*, ya sea automático “0” o generado por el tutor “1”), *correct_side* (identifica el lado correcto: izquierdo “0” o derecho “1”).

(ii) **Variables de texto (String):** *module* (módulo evaluado, vocabulario “M1”, letras “M2” o habla “M3”), *id_Tutor*, *id_Patient* y *date_event* (estos valores permiten asociar los eventos a una sesión específica).

(iii) **Variable tipo double (Double):** *resp_time*, en esta variable se guardó el tiempo de respuesta de las evaluaciones o la duración de los recreos, como un número flotante de doble precisión.

Esta clase guarda en forma secuencial los eventos, los que posteriormente pueden ser consultados por la aplicación del tutor, para generar las estadísticas. La información de palabras evaluadas, el tiempo y el tipo de respuesta, son utilizados para visualizar una lista detallada de los eventos ocurridos en cada sesión.



(a) (b)

Fig. 4.3. Ejemplo de datos de Clase Evento.
(a) Evaluación de una palabra. **(b)** Recreo

En la Fig. 4.3.a, se muestra un ejemplo de una evaluación entre palabras. A través de la información almacenada es posible saber que se evaluó correctamente al primer intento la palabra "lentes", frente a la palabra "regalo", en una evaluación de vocabulario ("M1"). Se puede determinar que la palabra correcta estaba al lado izquierdo, el usuario evaluado era rut 20.000.000-0, fue evaluado a las 16:58 hrs del 2/08/2017 y demoró 3,08 segundos en responder.

En la Fig. 4.3.b, se observa un ejemplo de la clase al guardar una instancia de un recreo. La información obtenida permite identificar el momento exacto en el que se guardó el recreo en conjunto a su tiempo de duración en segundos.

E. Stats

La clase *Stats*, guarda las estadísticas de una sesión, una vez que esta ha concluido. Esta clase es generada para las sesiones que tengan su variable (*stats_OK == false*), y es creada cada vez que se solicita visualizar las estadísticas de un usuario. El valor se hace estático cuando se crea un día diferente al de la sesión. Con esta lógica se entrega la capacidad de visualizar estadísticas de sesiones en desarrollo, con los valores disponibles hasta el momento en la Base de datos.

(i) **Variables de texto (String):** *ID_Patient*, *ID_Tutor*, *DATE_EXAM* (día de sesión), *DATE_ENTERED* (día en que fueron ingresados los datos).

(ii) **Variables enteras (Integer):** *M1_PA*, *M1_PC*, *M1_PI*, *M1_PO*, *M2_PA*, *M2_PC*, *M2_PI*, *M2_PO*, *M3_PA*, *M3_PC*, *M3_PI*, *M3_PO*, *ALL_PA*, *ALL_PC*, *ALL_PI y ALL_PO*. En este caso, M1, M2 y M3, representan al módulo correspondiente, ALL se

refiere a todas las palabras evaluadas en la sesión, PA (Palabras Aprendidas), PC (Palabras Correctas), PI (Palabras Incorrectas) y PO (Palabras Omitidas).

(iii) **Variable tipo float (Float):** *M1_TR*, *M2_TR*, *M3_TR*, *ALL_TR*, en estas variables se guardan el tiempo promedio de respuesta de las evaluaciones de cada tipo de evaluación, como un número flotante, debido a que es el formato de entrada de los gráficos que se implementaron en la aplicación del Tutor.

Para las estadísticas se tomaron las siguientes consideraciones descritas en TABLA 4.1.

TABLA 4.1. ABREVIATURAS UTILIZADAS PARA DESCRIBIR ESTADÍSTICAS

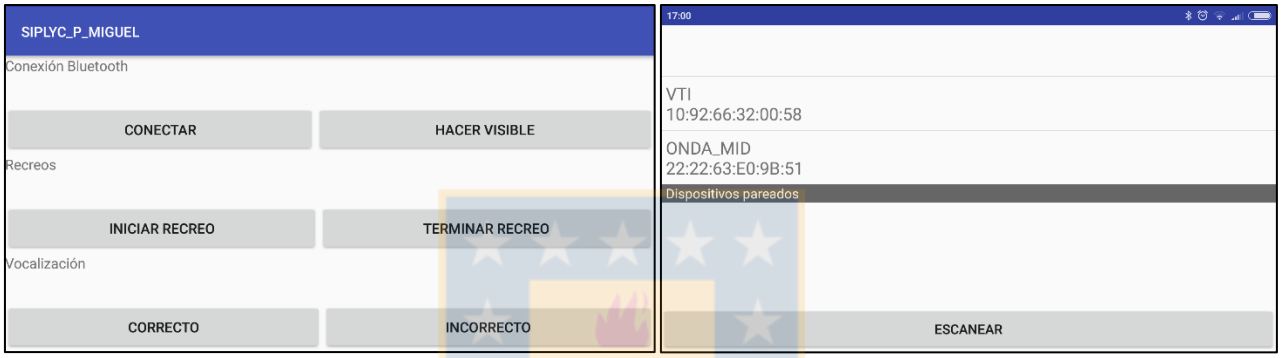
Abreviatura	Estadística	Consideración
M1	*Módulo de Vocabulario*	Se consideraron aquellas **palabras** evaluadas en estados *Testing Objeto X*.
M2	*Módulo de Letras*	Se consideraron aquellas **letras** evaluadas en estados de *Testing Objeto X*.
M3	*Módulo de Vocalización*	Considera **palabras y letras** evaluadas en estados de *Testing de Vocalización*
ALL	*Todos los objetos*	Considera a todos los objetos evaluados (palabras o letras) tanto en estados de *Learning* como de *Testing.*
PA	*Palabras Aprendidas*	Considera aquellos **objetos únicos** evaluados correctamente (es posible que se repitan algunos objetos dentro de una sesión)
PC	*Palabras Correctas*	Considera aquellos objetos evaluados como **correctos,** sin importar si es en primer o segundo intento.
PI	*Palabras Incorrectas*	Considera objetos evaluados como **incorrectos.**
PO	*Palabras Omitidas*	Considera aquellos objetos evaluados como **omitidos.**
TR	*Tiempo de Respuesta*	Se refiere al **tiempo promedio de respuesta** considerando exclusivamente las evaluaciones **correctas e incorrectas** (sin considerar las palabras omitidas).

OBS: Describe las abreviaturas utilizadas para identificar las estadísticas como variables de la Clase *Stats*.
Se consideran los procesos de evaluación mediante cuatro estados consecutivos descritos previamente como: *Learning Objeto 1, Learning Objeto 2, Testing Objeto X, Testing de Vocalización (Ver* Fig. 2.32, *pág. 37).*

4.3. Aplicación Secundaria o de Tutor

4.3.1 Prototipo Inicial

En primer lugar, el ingeniero a cargo de la actividad principal, desarrolló un prototipo para probar las funcionalidades de conexión por Bluetooth. Esta versión contaba con una actividad principal (Fig. 4.4.a) y una actividad de conexión que permite escanear en búsqueda de dispositivos disponibles (Fig. 4.4.b). La actividad principal permitía habilitar la conexión, iniciar/finalizar recreos y evaluar la vocalización como correcta/incorrecta.



(a) **(b)**

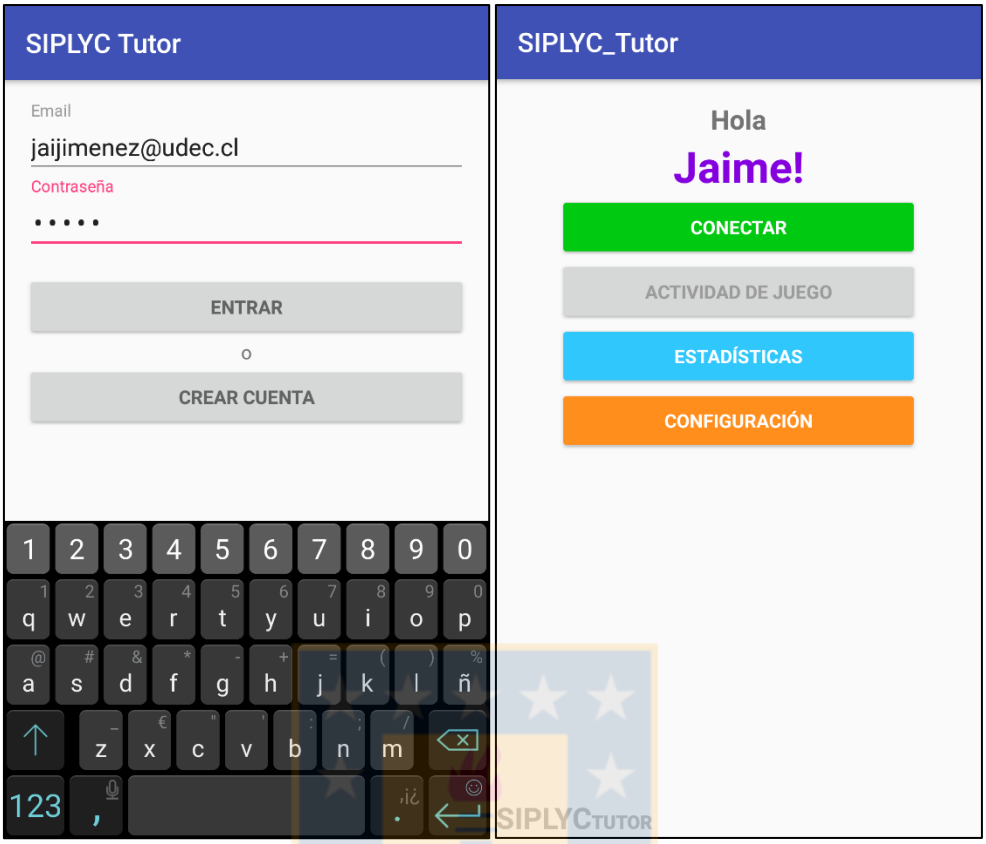
Fig. 4.4. Screenshot de versión inicial de Aplicación de Tutor.
(a) *MainActivity* de aplicación del tutor. **(b)** Actividad de conexión

4.3.2 Prototipo Final

Como se presentó en el capítulo anterior, se diseñó una aplicación con diferentes actividades. Para la implementación, se decidió cambiar la orientación por defecto de algunas actividades de horizontal a vertical (*Portrait*) para facilitar el uso de esta aplicación con una sola mano y desde un dispositivo celular (Ver Anexo A.). Esta aplicación fue probada en un dispositivo de 5.5”, con resolución de 1920 x 1080 píxeles, con sistema operativo Android 6.0.1. Su implementación se describirá a continuación

A. Actividad de Login

Como la aplicación tendrá acceso a la información de los sujetos evaluados, se decidió implementar una actividad de *Login* que bloquea la aplicación mediante los datos de la clase “Tutor”, a través de su correo y contraseña. Se utilizó la actividad de login por defecto incluida en Android Studio (Fig. 4.5.a), a la cual se le implementó la lógica para guardar a los usuarios en forma local como *SharedPreferences* (preferencias compartidas), en conjunto a la Base de datos (realizando una Query previa para no sobrescribir a un perfil existente).



(a) **(b)**

Fig. 4.5. Captura de pantalla de Aplicación de Tutor.
(a) Actividad de *Login* **(b)** *MainActivity*

B. Actividad de Registro

Una versión de la aplicación (Fig. 4.5.a) cuenta con un botón que permite ingresar a una actividad de registro ("*crear cuenta*") para crear una nueva cuenta de *Tutor*. Sin embargo, para versiones posteriores se eliminará para que solo los investigadores con cuentas existentes puedan ingresar.

C. MainActivity

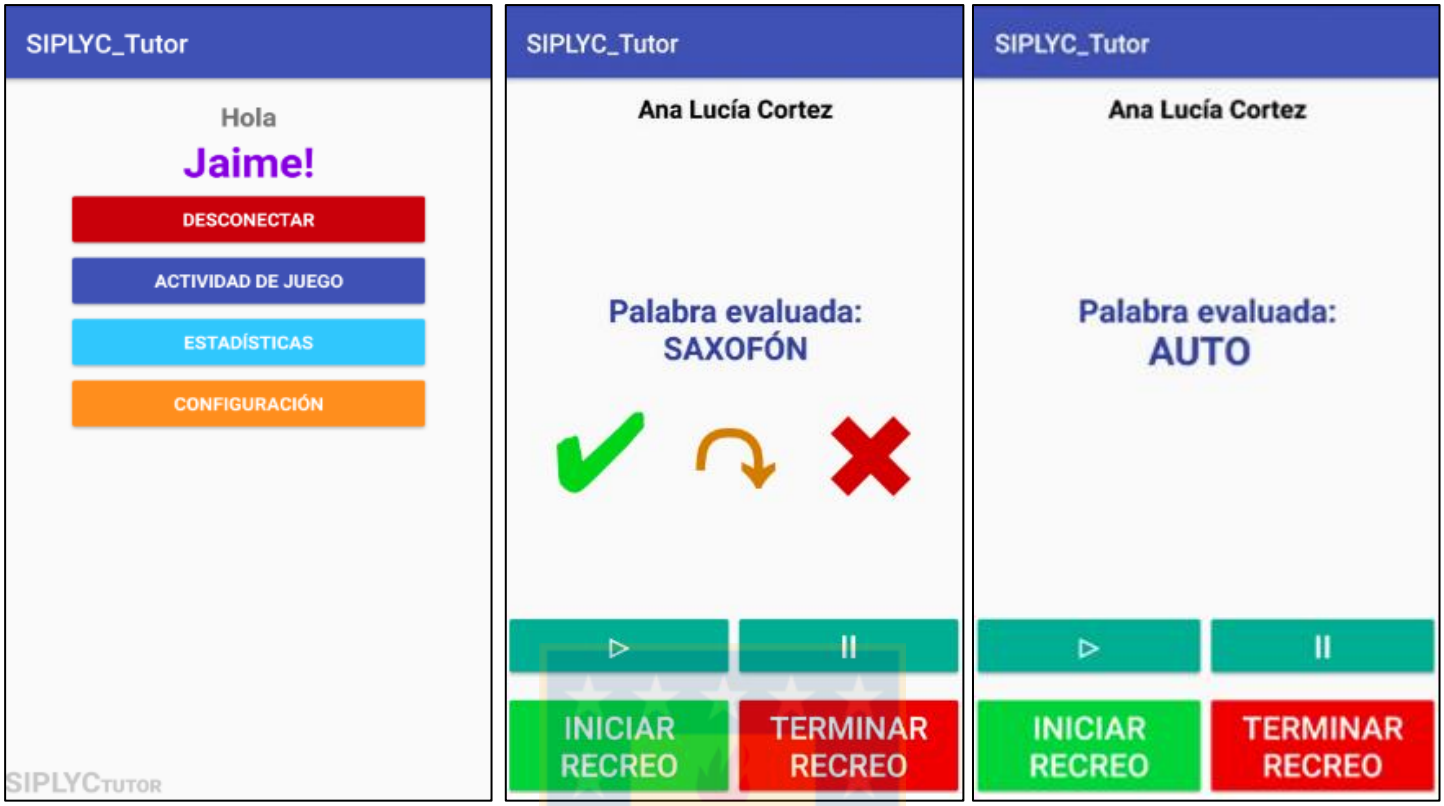
La actividad de *Menú Principal* de la aplicación (Fig. 4.5.a), cuenta con cuatro botones visibles:

(i) **Conectar:** **P**ermite realizar la conexión vía Bluetooth.

(ii) **Actividad de Juego:** Inicialmente bloqueado, solo permite su acceso cuando se encuentra correctamente conectado a la aplicación de juego.

(iii) **Estadísticas:** Permite ingresar a la actividad de Estadísticas.

(iv) **Configuración:** Permite realizar cambios de la contraseña personal e identificar a los usuarios que abandonaron la investigación en el transcurso de la misma.



(a) **(b)** **(c)**

Fig. 4.6. Actividad de Control y Evaluación.
(a) Actividad de Juego disponible **(b)** Actividad de Evaluación **(c)** Actividad de Supervisión

D. Actividad de Juego

Al conectarse correctamente a través de Bluetooth con la aplicación principal, se permite el ingreso a la *Actividad de Juego*, como se observa en la Fig. 4.6.a.

Esta actividad, compuesta por un *RelativeLayout* para la disposición de los elementos en pantalla, cuenta con una versión que dispone de botones para evaluar para cuando se realiza la actividad de vocalización (Fig. 4.6.b.) o una versión limpia para cuando se decide supervisar un ejercicio sin evaluación del Habla (Fig. 4.6.c.).

Se cuenta con íconos que representan las diferentes interacciones disponibles:

(i)  Permite evaluar correctamente.

(ii)  Permite evaluar como incorrecto.

(iii)  Permite omitir la evaluación.

(iv) 

Permite pausar una sesión.

(v) 

Permite reanudar una sesión pausada.

(vi) 

Inicia un recreo en la actividad principal.

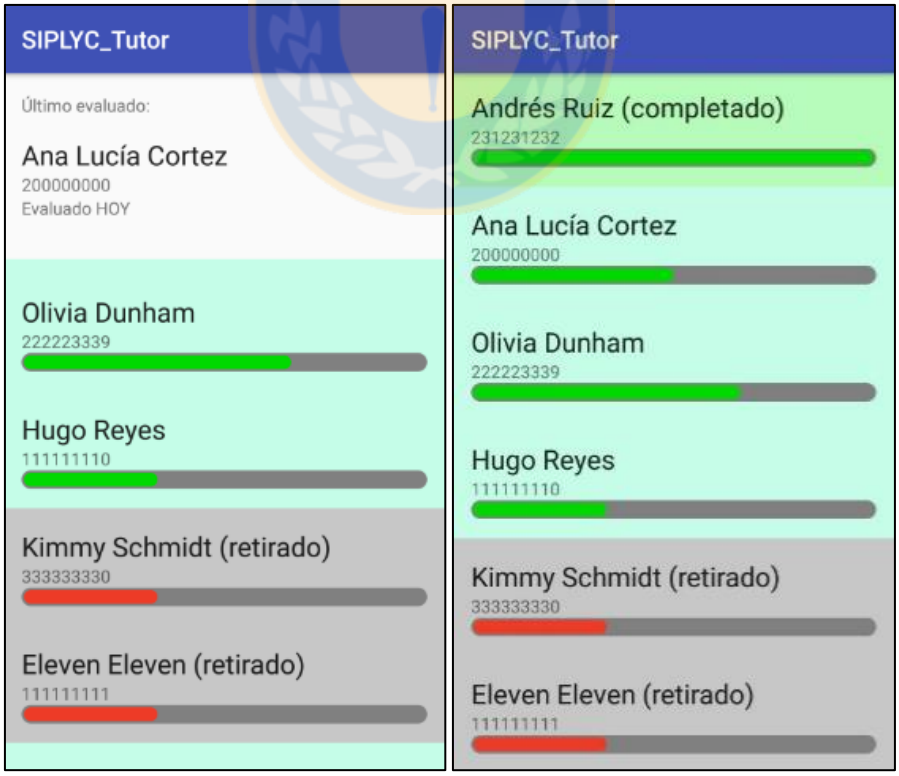
(vii) 

Detiene un recreo activo.

Como se explicó previamente, la acción de estos botones genera un mensaje que es enviado como un *stream* de Bytes que es identificado por la otra aplicación. Esto ocurre de forma inversa para obtener los datos de nombre del usuario que está evaluando y el objeto evaluado, los cuales se reciben a través de la comunicación Bluetooth, modificando los *TextView* de la interfaz del tutor.

Los eventos son registrados en la Base de datos desde la otra aplicación, por lo que en esta actividad no hay mayor interacción con *Firebase*.

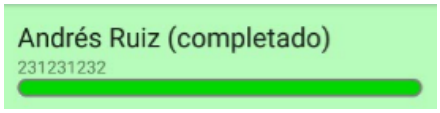
E. Módulo de Estadísticas: Lista de Usuarios



a) **(b)**

Fig. 4.7. ListView de usuarios evaluados.
(a) Último evaluado en prioridad. **(b)** Alumnos ordenados según estado.

Esta actividad permite la visualización de las estadísticas generadas a partir de las evaluaciones. La interfaz muestra un ListView, donde cada elemento está personalizado de acuerdo a su estado:

(i) 

Completó las 8 -10 sesiones según corresponda.

(ii) 

Usuario activo, y en progreso.

(iii) 

Usuario retirado (Sus estadísticas se conservan).

En el caso de haberse conectado con la aplicación secundaria para evaluación, se mostrará en primera opción al usuario evaluado.

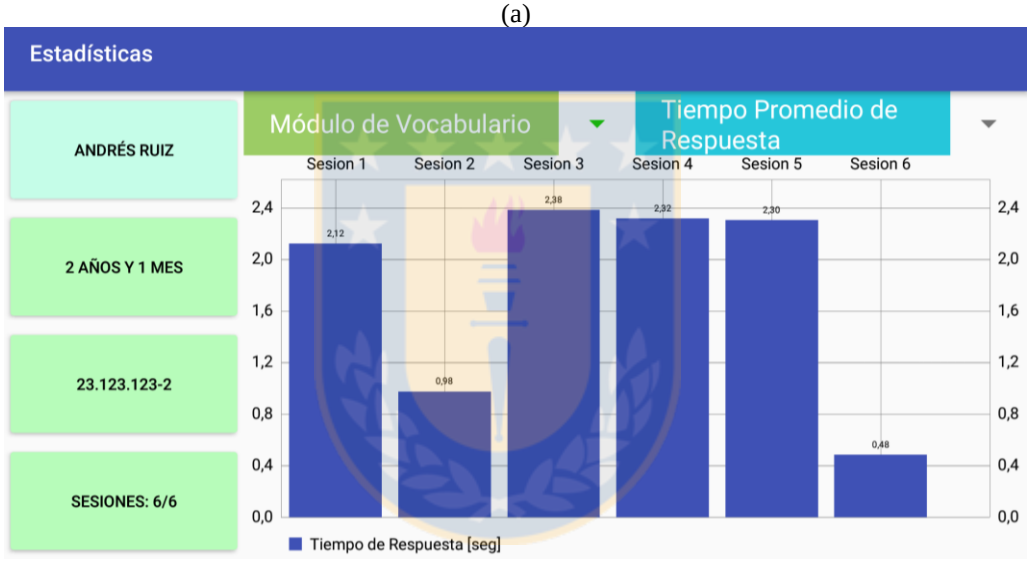
La lista es llenada a partir de una *Query* al nodo "*patients*" de *Firebase*. Se implementó un *Adapter* que recibe los objetos recuperados de la consulta, y a partir de su estado (retirado, completado, activo) los agregará a la *ListView* con sus respectivos colores. Se implementó un *OnItemTouchListener*, que responde ante un evento táctil sobre uno de los ítems de la lista, devolviendo la *view* de la posición. A través de esto, se obtiene el valor del ID de usuario y se realiza un *Intent* a la actividad que grafica las estadísticas.

F. Módulo de Estadísticas

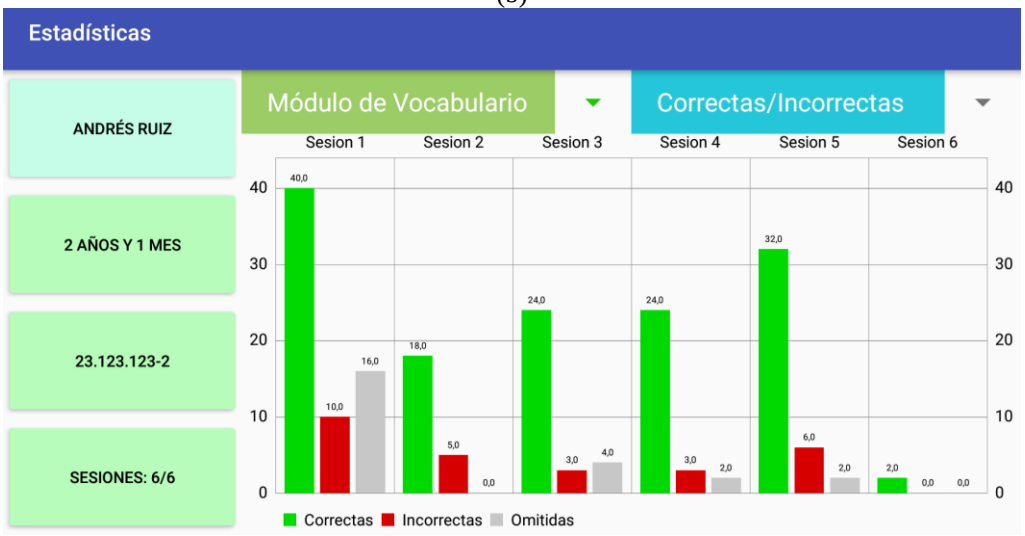
En esta actividad se grafican las estadísticas para los tres módulos evaluados. Para su implementación se utilizó la biblioteca *MPAndroidChart* [43], que cuenta con diferentes tipos de gráficas, permitiendo personalizar su diseño y respuesta a gestos. De forma análoga a otros tipos de *views*, el gráfico se posiciona en el *layout* en el archivo XML de la actividad, para luego generar una instancia de tipo *BarChart* en su clase de JAVA. Se utilizó un gráfico de barras individual para la estadística de Palabras Aprendidas (Fig. 4.8.a) y Tiempo de Respuesta (Fig. 4.8.b), mientras que información de correctas, incorrectas y omitidas fue incluida en un mismo gráfico (Fig. 4.8.c).

Se utilizaron dos *Spinners* (listas desplegables) para variar entre los diferentes módulos, y las diferentes estadísticas. Se incluyó además información del usuario seleccionado: Nombre, Edad, Rut y el número de sesiones. Toda información disponible desde la Base de datos.

(a)



(b)



(c)

Fig. 4.8. Captura de gráfico de estadísticas

(a) Palabras Aprendidas **(b)** Tiempo Promedio de Respuesta **(c)** Correctas/Incorrectas/Omitidas

En el método *onCreate()* de esta actividad, se realizan las *Querys* correspondientes para generar las estadísticas de las sesiones que aún no las hayan creado. De esta forma son temporales las estadísticas de sesiones del mismo día de la consulta, creando estadísticas estáticas para las de días anteriores. Posteriormente, se realiza una *Query* al nodo "*stats*" de la base de datos, para generar los gráficos.

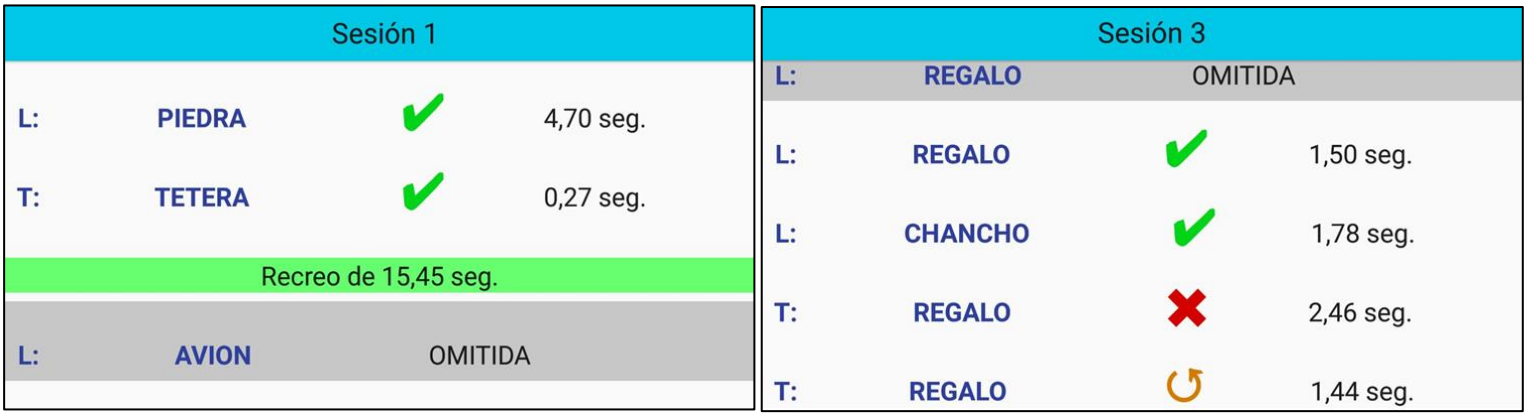
Se realizó un *@Override* al método *onChartDoubleTapped()*, dentro de los métodos de respuesta a gestos propios del *BarChart*. De esta forma, se configura el grafico para abrir la actividad de "*Detalle por Sesión*", al realizar el gesto sobre la sesión seleccionada.

G. Módulo de Estadísticas: Detalle por Sesión

Esta actividad permite visualizar los eventos de la sesión en orden cronológico. A través de un *ListView* de forma similar a la visualización de lista de usuarios, se realiza una *Query* al nodo "*events*" de *Firebase*, para obtener aquellas evaluaciones y recreos de una sesión específica.

Mediante un *Adapter* se modifican los elementos a graficar por cada ítem. En la Fig. 4.9.a. se observan dos palabras evaluadas correctamente, un recreo y una palabra omitida. Se identificó mediante "*L*" estados de *Learning*, y con "*T*", estados de *Testing*. De esta forma, es posible reconstruir en detalle una jornada de evaluación. Por ejemplo, en la Fig. 4.9.b., sabemos que en la etapa de *Learning* se le presentó la palabra "*regalo*", omitida en primera instancia y luego contestada correctamente en 1,5 seg. Luego se le presentó la palabra "*chancho*" y finalmente en etapa de *Testing* se evaluó "*regalo*", la cual se contestó correctamente al siguiente intento.

El detalle de estas estadísticas se refiere a palabras del módulo de vocabulario, de acuerdo a la lógica descrita anteriormente. Pese a que son datos generados mediante consultas a la Base de datos, estos son graficados en un tiempo de carga prácticamente indetectable para el usuario.



a) **(b)**

Fig. 4.9. Detalle de eventos por sesión.
(a) Sesión con recreo **(b)** Sesión con palabra omitida y correcta al segundo intento.

a) **(b)** **(c)** **(d)**

Fig. 4.10. Actividad de Configuración

(a) Menú de Configuración **(b)** Cambio de contraseña **(c)** Retirar sujetos **(b)** Sujeto retirado

H. Actividad de Configuración

En esta actividad se le permite al Tutor modificar su contraseña (Fig. 4.10.b). Este cambio provoca un efecto tanto en los valores guardados en forma local a través de *SharedPreferences* (preferencias compartidas) como en la Base de datos.

Se agregó la funcionalidad de "*retirar*" a *Usuarios* que abandonaron la investigación, esto hace que cambien de estado y de prioridad para su visualización, pero no elimina sus datos, permitiendo así guardar su avance en el caso de que se reintegren en el futuro.

4.4. Discusión y Conclusiones

El uso de *Firebase* permitió la correcta implementación de ambas aplicaciones y el desarrollo de las características requeridas por el proyecto. Los datos leídos desde la Base de datos, fueron recuperados sin generar tiempos de carga visibles para el usuario. La persistencia en el disco y la capacidad sin conexión, permite guardar correctamente los eventos de evaluación en la aplicación de juego, los cuales, una vez sincronizados, son visibles por todos los dispositivos que cuenten con la aplicación de *Tutor*.

Capítulo 5. Evaluación del Sistema

5.1. Introducción

En el presente capítulo se presentan las pruebas realizadas para evaluar las funcionalidades del sistema. Actualmente la aplicación principal se encuentra en fase de desarrollo, por lo que se evaluó la aplicación de tutor en diferentes dispositivos, con las funcionalidades disponibles.

5.2. Descripción de Pruebas

Para la realización de esta prueba se utilizó un prototipo de la aplicación principal en Tablet *Samsung Galaxy Tab (SM-T350)* de pantalla de 10,1" (pulgadas), con resolución de 1280x800 píxeles y sistema operativo Android 5.0.2.

Se instaló la Aplicación "*SIPLYC Tutor*" en diez dispositivos, con diferentes versiones de sistema Android y resolución de pantalla. Se evaluaron 5 aspectos principales:

(i) **Instalación:** Se instaló la aplicación, mediante su archivo *APK.* (formato en el cual se guarda el instalador de una aplicación en Android). Se verificó la correcta instalación.

(ii) **Ingreso de Usuario:** Se evaluó si se podía crear una nueva cuenta de tutor correctamente, además de cambiar la clave.

(iii) **Lectura y Visualización de Estadísticas:** Se evaluó que la aplicación cargara correctamente las estadísticas en modo online y que pudiera acceder a ellas en modo offline. Se verificó (en forma online) si era capaz de sincronizar en tiempo real para visualizar las estadísticas del niño mientras es evaluado.

(iv) **Conexión Bluetooth:** Se vincularon los dispositivos con el Tablet de evaluación. Se verificó que la app del Tutor pudiera visualizar correctamente el nombre del sujeto evaluado, en conjunto a la palabra. Se comprobó el correcto comportamiento de control con envío de Recreos, y la visualización de los botones de evaluación de forma exclusiva en el estado de *Testing de Vocalización.* No se evaluaron los botones de pausa, pues esa funcionalidad aún no está implementada.

En la TABLA 5.1, se presentan los resultados de la evaluación. Se entrega la información técnica de modelo, tamaño de pantalla, resolución y versión de Android, de los dispositivos utilizados. Se les solicitó a los usuarios de los dispositivos, realizar las actividades descritas y entregar comentarios, en el caso que encontraran errores o recomendaciones para mejorar las funcionalidades.

TABLA 5.1. RESULTADOS DE PRUEBAS DE EVALUACIÓN

Modelo	Instalación	Ingreso de Usuario	Lectura y visualización de Estadísticas	Conexión Bluetooth	Comentarios
Lenovo K6 5'' 1080x1920 px Android: 6.0.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Correcta **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Correcta
Samsung Galaxy J5 2016 5.2'' 1280x720 px Android: 6.0.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Incorrecta al reiniciar actividad. **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Recomienda visualizar lista de niños por nombre en app principal. Campo de texto de rut no permite ingresar letra "k" en app principal. Error en visualización de mes en campo de fecha de nacimiento. Identificó error en visualización de ListView, al retirar alumnos. Sugiere ventana emergente en vez de *long Click* para retirar sujetos.
XiaomiRedmi3s 5" 1280x720 px MIUI 8 (Android: 6.0.1)	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Correcta **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Correcta
XiaomiRedmi4x 5,5" 1080x1920 px MIUI 8 (Android: 6.0.1)	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Correcta **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Correcta
Sony Xperia E5 5" 1280x720 px Android 5.1.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Correcta **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	No le gustó que al salir de la aplicación se cerrara la sesión. Apareció la actividad de registro al volver atrás.

Lenovo Vibe K5 5” 1280x720 px Android 5.1.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Incorrecta al reiniciar actividad **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Considera incómodo ocupar una contraseña de más cinco caracteres, sugiere un pin de 4.
Sony Xperia M2 4.8” 540 x 960 px Android 5.1.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Incorrecta al reiniciar actividad **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	Sugiere que ítems en la ListView de los usuarios para acceder a las estadísticas son muy grande. Sugiere eliminar el código de colores (activo, retirado, completado) y dejar solo la barra de progreso con color.
Motorola Moto-G 4,5” 1280x720 px Android 5.1	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Incorrecta al reiniciar actividad **Palabra:** Correcta **Recreos:** Correcta **Botones:** Correctos	Sugiere más colores en la aplicación del niño. Sugiere que la actividad de entrenamiento le indique al niño que debe esperar las instrucciones para responder.
LG Prime II 4,5” 480 x 854 Android 5.0	Correcta	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Correcta **Palabra:** Correcta **Recreos:** Correcta **Botones:** Correctos	Conexión Bluetooth debe ser realizada dos veces para permitir conexión.
Huawei Y635-L02 5” 480 x 854 px Android 4.4.4	Instalación directa desde Android Studio	**Creación:** Correcta **Cambio de Contraseña:** Correcta	**Online:** Correcta **Offline:** Correcta **Sincronización**: Correcta	**Nombre:** Incorrecta al reiniciar actividad. **Palabra:** Correcta **Recreos:** Correcta **Botones:**Correctos	No se ven colores de los botones.

OBS: Se probó en un dispositivo celular Own S3010 Android 4.2.2, que no permitió la instalación de la aplicación, debido a tratarse una versión menor a la mínima requerida (4.4.4).

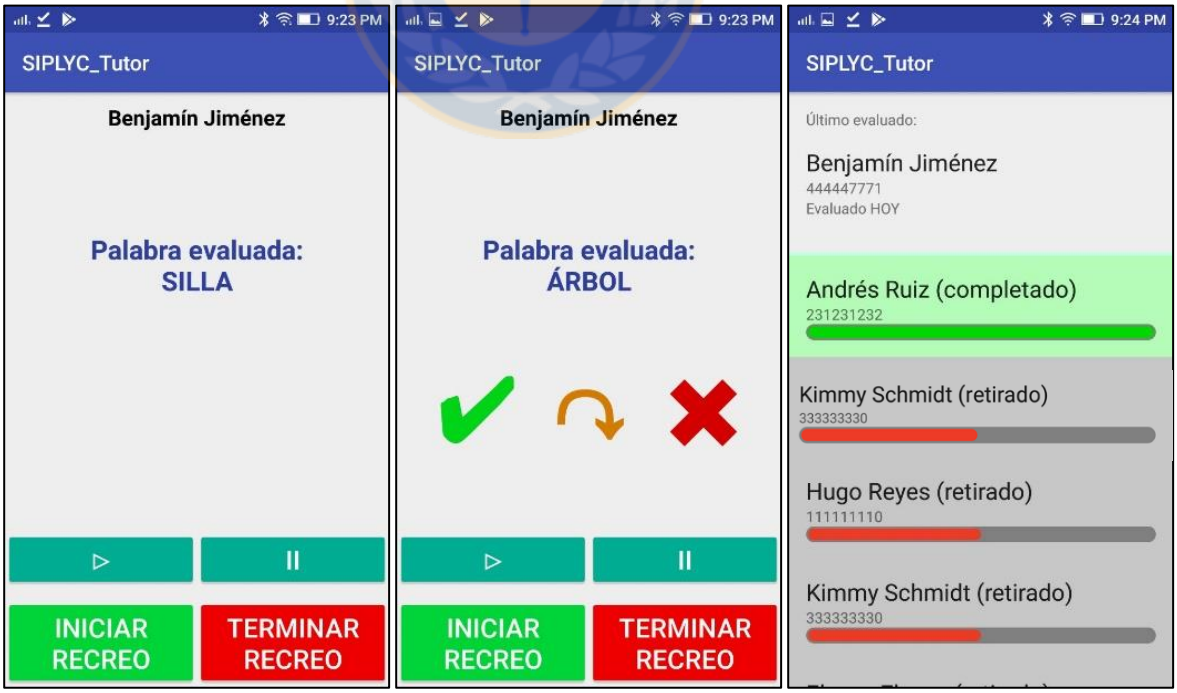
5.2.1 Resultados de Evaluación

La aplicación se instaló correctamente en los diez dispositivos que cumplían con las características mínimas de sistema operativo Android (4.4.4). Sin embargo, el dispositivo con Android 4.4.4, no aceptó al archivo APK, y se realizó la instalación a través de Android Studio.

La creación de nuevos usuarios fue correcta, pudieron crear una cuenta y modificar posteriormente su contraseña. La sincronización de los datos y posterior visualización, fue correcta tanto en modo online como offline.

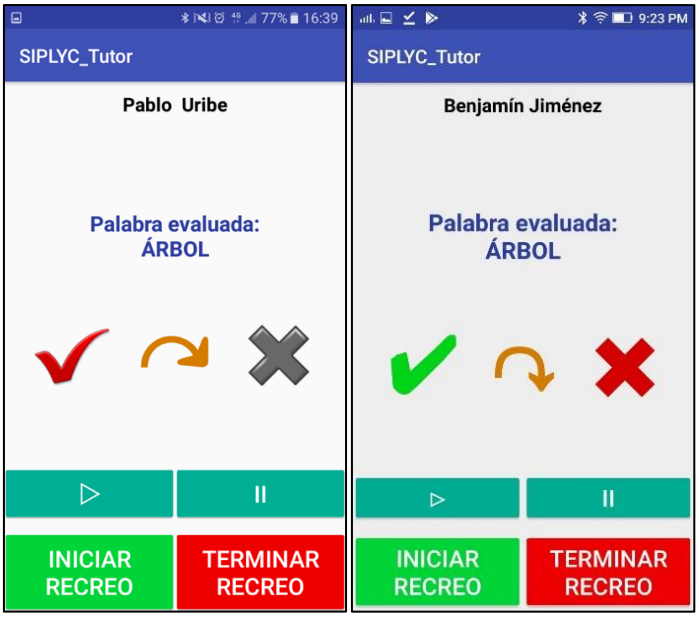
La conexión a través de Bluetooth fue correcta en los dispositivos evaluados, y la aplicación principal respondía a las instrucciones de control, generándose correctamente recreos, durante la actividad de evaluación, así como la aplicación de Tutor, fue capaz de reconocer las palabras evaluadas (Fig. 5.1.a -b), el sujeto evaluado y presentarlo con prioridad (Fig. 5.1.c).

La distribución de los elementos en pantalla fue correcta en todos los dispositivos, sin "deformaciones", pese a los diferentes tamaños de pantalla y de resolución. Se identificaron algunas variaciones entre diferentes marcas, donde los elementos ASCII (carácter de texto) se presentaron en formato diferente al programado. Por ejemplo, en la Fig. 5.2.a, los caracteres de "correcto" e "incorrecto", se presentaron en otros colores predeterminados, en comparación a Fig. 5.2.b, que presenta los íconos de acuerdo a como se programó.



(a) **(b)** **(c)**

Fig. 5.1. Capturas de pantalla en modelo LG Prime II
(a) Menú de Configuración **(b)** Cambio de contraseña **(c)** Retirar sujetos

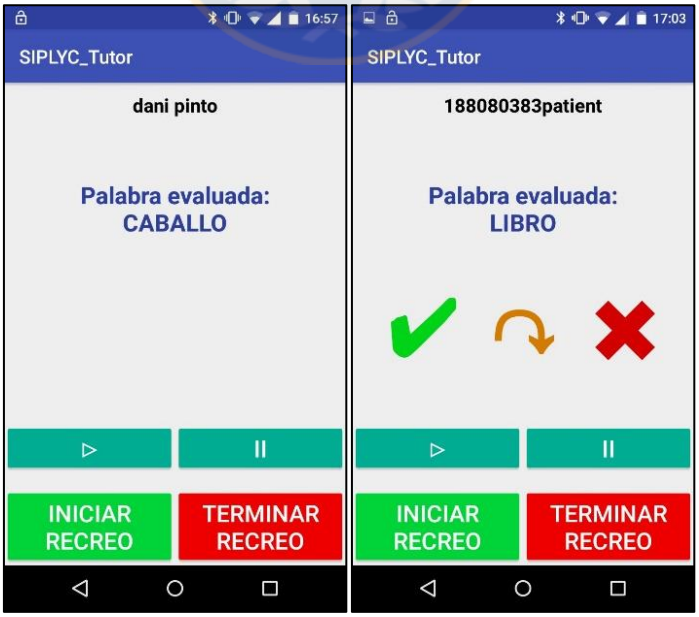


(a) **(b)**

Fig. 5.2. Capturas de pantalla con íconos cambiados.
(a) Galaxy J5 **(b)** LG Prime II

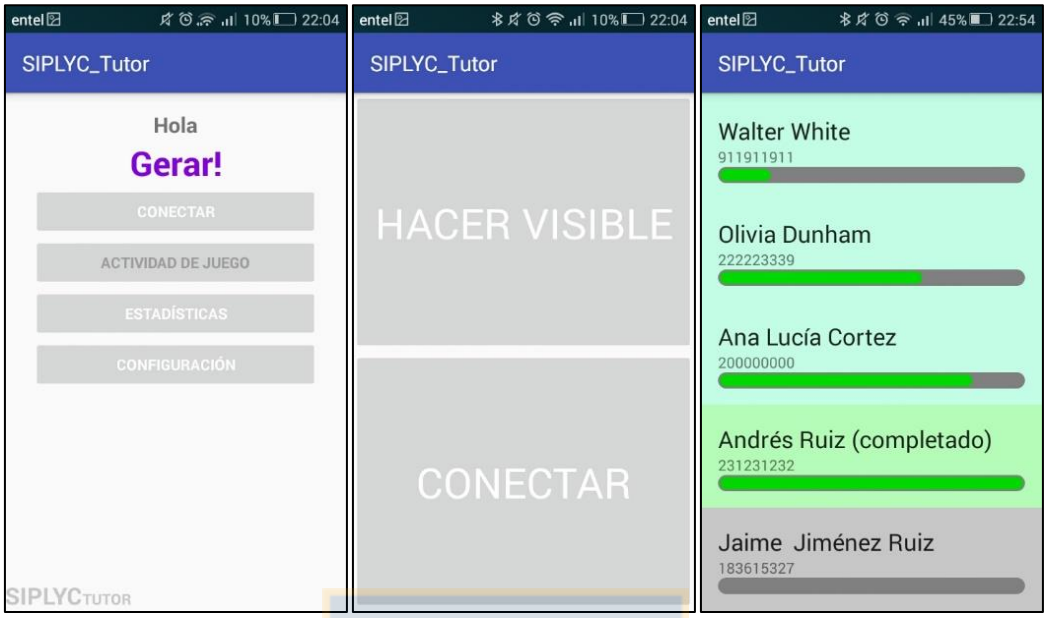
5.2.2 Errores Identificados

Se identificó un error al reiniciar la actividad de juego desde la aplicación principal. Esto enviaba el ID del sujeto evaluado a través de Bluetooth, y es utilizado por la UI del Tutor para identificar al sujeto por su nombre en pantalla (Fig. 5.3). Se identificó que era un problema debido al envío de la palabra, lo cual se corrigió en la aplicación de juego.



(a) **(b)**

Fig. 5.3. Capturas de pantalla con error en nombre recibido.
(a) Pantalla con nombre correcto. **(b)** Pantalla con nombre incorrecto.



(a) **(b)** **(c)**

Fig. 5.4. Capturas de pantalla en modelo Huawei con Android 4.4.4
(a) MainActivity **(b)** Actividad de Conexión **(c)** Lista de usuarios evaluados.

Se identificó un error de visualización de colores de los botones en el dispositivo evaluado con Android 4.4.4 (Fig. 5.4.a-b). En esta versión de Android no acepta colores de *background* para *views* de botones, opción que se encuentra disponible a partir de la versión posterior. Se propone como solución el uso de un *layout* diferente, que modifique el color de las letras de los botones, para facilitar una mejor visualización de la interfaz. Se observa que los colores de los ítems de la lista de usuarios evaluados no presenta problemas (Fig. 5.4.c).

Otro error identificado fue en la *Actividad de Retiro*. Esta actividad permite marcar como "retirado" al usuario que abandonó la investigación sin terminar el proceso completo. Mediante un *longClick* (Tocar durante un tiempo), el usuario cambiaba de estado. Sin embargo, al realizar el *scroll* (deslizamiento) de la lista, el ítem volvía a su estado original, pese a que programáticamente la variable había sido editada, debido a un problema de actualización de elementos visibles del objeto ListView. Para responder a esta situación, se modificó el efecto para forzar una actualización de la actividad al realizar un cambio en lista.

Cuando uno de los usuarios intentó cerrar la aplicación mediante el botón de regreso, se abrió nuevamente la *Actividad de Registro*, aún con los datos ingresados. Esta situación se puede corregir forzando el cierre de la actividad al realizar un correcto registro de los datos.

En la aplicación principal, en la Actividad de registro de *Nuevo Usuario*, se identificó una diferencia en el mes de la fecha de nacimiento ingresada y la que se muestra en el campo de texto. Este es un error producido por la indexación de fechas en JAVA, que entrega números a los meses entre 0 y 1. Si bien la fecha es guardada correctamente, el usuario ve números equivocados, por lo que se debe corregir. En esta actividad se identificó que el campo no permite ingreso de la letra “k”, lógica que puede ser corregida solicitando ingresar al usuario un cero, o incluyendo esta letra los caracteres permitidos.

5.2.3 Funciones Sin Conexión

Se revisaron diferentes escenarios que podrían producirse debido a las funcionalidades sin conexión de la Base de datos.

Se identificó un error al generar una estadística incompleta debido a un dispositivo que se desconectó a la mitad de una sesión. Sin embargo, posterior a la sincronización de los datos, fue posible generar nuevamente las estadísticas, borrando los valores previos directamente desde la consola de Firebase en la interfaz web del administrador.

Si bien es posible la duplicación de usuarios y sesiones, al realizar la creación de estos datos de forma offline, esto no afecta en el almacenamiento de las estadísticas, ni sobrescribe la información. Esto, debido a la lógica utilizada y a la propiedad de Firebase de generar ID únicas para cada valor agregado a la Base de datos.

5.2.4 Comentarios

Se realizó registro de las observaciones de los usuarios durante las pruebas. Se describirá las consideradas relevantes.

A. Sugerencias en Aspectos Visuales

Se sugirió, incluir el nombre junto al rut, en la lista de sujetos de la aplicación principal. El prototipo actual muestra los directorios generados en la memoria del dispositivo y son identificados mediante el ID entregado al usuario (en este caso, su rut). Se recomendó, además, incluir más colores en la interfaz de usuario de la aplicación principal.

B. Sugerencias en Funcionalidades

Algunos usuarios consideraron incómodo el uso de una contraseña de mínimo cinco caracteres, recomendaron el uso de un pin de cuatro números, que suele ser más fácil de recordar.

Se sugiere el uso de una ventana emergente para confirmar el retiro de un sujeto, en vez del uso de un *LongClick*. Esto facilitaría el uso de la función, le daría más seguridad y podría ayudar a realizar la recarga de la ListView sin que lo note el usuario.

Se sugirió que la actividad de entrenamiento le indique al niño que debe esperar las instrucciones para responder.

Se sugirió un cambio en la forma de entregar el detalle de evaluaciones por sesión, realizando un resumen de los eventos por palabra, sin considerar su presentación en estado de *Learning* (de acuerdo a la Fig. 5.5). En esta propuesta es mucho más clara la evaluación individual de cada objeto. Por lo que podría implementarse en una futura versión.

Fig. 5.5. Propuesta de visualización de evaluación.

5.3. Discusión y Conclusiones

En el presente capítulo se realizó la evaluación de las funcionalidades de la aplicación de Tutor desarrollada, y las componentes integradas a la aplicación de estimulación del lenguaje para permitir su comunicación inalámbrica y sincronización con la base de datos implementada.

En general, los módulos tuvieron un correcto funcionamiento en los dispositivos con sistema operativo Android 5.0 o superior, por lo que se definirá éste, como el mínimo requerido. Se identificaron algunos funcionamientos errados que pudieron ser corregidos.

La aplicación principal se comportó de acuerdo a la lógica esperada. Se registraron los comentarios de los usuarios que probaron el sistema *SIPLYC*, sobre aspectos visuales y funcionales que pueden ser utilizados para la mejora del actual software en su versión final a utilizar en la investigación futura.

Capítulo 6. Discusión y Conclusiones

6.1. Discusión

En el presente trabajo, se desarrolló parte de un software de estimulación lingüística para niños entre 2 y 3 años y medio, para el proyecto denominado Sistema Interactivo de Promoción del Lenguaje y de la Comunicación (*SIPLYC*).

Se presentó una revisión bibliográfica referente a los procesos cognitivos y fisio-anatómicos asociados al desarrollo del lenguaje en esta edad, además de realizar un análisis de algunos de los softwares de estimulación existentes en el mercado. Se observó que la mayoría de las apps existentes fueron creadas por profesionales o por padres, pero sin contar con algún seguimiento, o mecanismos de evaluación que permitan comprobar su efectividad. Se revisó el trabajo anterior del grupo investigador, denominado *SIEL* y se analizaron los requerimientos para este nuevo sistema. De acuerdo a la bibliografía, la aplicación final de ejercicios debe considerar un diseño que no presente distracciones que saquen al niño del juego. Esto puede ser contrario con el efecto esperado en el grupo de mayor vulnerabilidad, esto es importante a considerar en los recreos a incluir.

Se presentó el diseño e implementación del *SIPLYC*, que cuenta con dos aplicaciones móviles desarrolladas para Android, una enfocada para la evaluación automática y estimulación del lenguaje (desarrollada por un ingeniero que es parte del proyecto), y otra para el control y supervisión por parte de los encargados de la investigación (desarrollada en el presente trabajo).

Se utilizó comunicación inalámbrica mediante Bluetooth para la interacción entre ambas aplicaciones, y se implementó una base de datos de *Firebase* para guardar los datos relevantes para la generación de estadísticas desde diferentes dispositivos de forma sincronizada. Se incluyeron ambas en la aplicación principal o de juego. Se consideró que esta implementación no afectase a la lógica de ejercicios ni las otras funcionalidades disponibles.

Se desarrolló la aplicación secundaria o de evaluación, que permitió la lectura de la base de datos para la generación y visualización de estadísticas. Se implementó un módulo de conexión Bluetooth y una actividad que permite la interacción con la aplicación principal, recibiendo datos desde ésta (usuario en evaluación y palabras evaluadas) y con una interfaz que permite “pausar/retomar ejercicios” o “iniciar/terminar recreos”. Se agregó una actividad de Login, que bloquea la aplicación mediante contraseña, para permitir la autenticación del Tutor que utilice la aplicación y proteger los datos.

El uso de *Firebase Realtime Database*, permitió prescindir de la configuración de un servidor propio y de una arquitectura más compleja para la sincronización de los datos. La base de datos implementada permite responder al requerimiento de múltiples clientes con información sincronizada, pero al mismo tiempo contar con respaldo y permitir su funcionamiento en estados sin conexión. Firebase cuenta con una consola de administración en la interfaz web, que permite revisar directamente la base de datos y corregir de forma sencilla posibles errores identificados en el ingreso de datos. Esta característica es de utilidad para modificar variables globales y generar efecto en todos los dispositivos sincronizados.

El uso de comunicación Bluetooth entre ambas aplicaciones, para el envío de evaluaciones, recreos y otras instrucciones en tiempo real, le permite funcionar correctamente en instancias de evaluación donde no se cuente con internet. De esta forma, se cumplen con los requerimientos del equipo de investigación, para el envío de recreos y evaluación del Habla.

Se evaluó el sistema *SYPLYC*, instalando la aplicación de Tutor correctamente en diez dispositivos móviles con sistema operativo superior a Android 5.0, y con limitaciones visuales en un dispositivo Android 4.4.4. Por lo que se definió como sistema mínimo requerido Android 5.0. Se probaron las funcionalidades añadidas en la aplicación de ejercicios, donde se observó un correcto comportamiento de la comunicación Bluetooth para la generación de eventos (evaluación y recreos) y una sincronización exitosa de los datos generados de forma tanto online como offline. La actividad secundaria, además de realizar las funciones por comunicación inalámbrica, permitió un correcto login/registro de tutores y una exitosa visualización de las estadísticas, donde se observaron tiempos de carga y lectura a la base de datos transparentes para el usuario. Esta evaluación permitió identificar y corregir errores menores de visualización, además de recibir comentarios sobre la interfaz de usuario y el comportamiento del sistema.

La generación de estadísticas y su visualización fue implementada con éxito, permitiendo diferentes niveles de detalle. Además de almacenar datos relacionados al contexto de presentación de la palabra, como su posición en pantalla en estados de *Learning* y *Testing*, o la palabra *versus* con la que fue evaluada. Lo que, se espera, permitirá una mejor evaluación de las curvas de aprendizaje e historial de cada usuario, siendo un aporte para el futuro estudio que hará uso del sistema *SIPLYC*.

6.2. Conclusiones

Se estudiaron versiones previas del software y la aplicación de ejercicios desarrolladas para el proyecto del sistema *SIPLYC*.

Se diseñó e implementó la aplicación del Tutor, con funciones de evaluación del habla y control de la aplicación principal, a través de comunicación inalámbrica mediante Bluetooth. Se implementaron tres tipos de evaluación: "*correcto*", "*incorrecto*" y "*omitida*"; y cuatro instrucciones de control: "*inicio/finalización de recreos*" y "*pausa/reanudar actividad*".

Se logró implementar correctamente una base de datos con datos persistentes en el disco, con funcionalidades sin conexión y con sincronización online de la información almacenada en múltiples dispositivos. El uso de Firebase para esta implementación, permitió tiempos de lectura/escritura imperceptibles para el usuario.

Se integraron a la aplicación principal las características desarrolladas para comunicación inalámbrica y uso de la base de datos. La sincronización de los datos almacenados se realizó correctamente, permitiendo, en la aplicación secundaria, la generación y visualización de estadísticas del progreso del niño evaluado y su historial de evaluación con diferentes niveles de detalle (por sesión, por módulo, por palabra).

Se evaluó el sistema *SIPLYC* instalando la aplicación del Tutor en diez dispositivos Android y la aplicación principal (con las características desarrolladas) en un Tablet. Las funcionalidades de conexión inalámbrica y base de datos tuvieron un correcto funcionamiento en ambas aplicaciones. Es importante destacar que esta evaluación permitió encontrar detalles desde el punto de vista de la interfaz de usuario que fueron corregidos. La aplicación de tutor se instaló correctamente para dispositivos con Android 5.0 o superior, por lo que se recomendará esta versión como la mínima requerida.

6.3. Trabajo Futuro

Como continuación directa de este trabajo se espera desarrollar un Manual de Usuario del sistema desarrollado considerando ambas aplicaciones, para los profesionales que harán uso de ésta. Se espera que este manual considere la información sobre los requerimientos de instalación de las aplicaciones, procedimiento para realizar la conexión Bluetooth, interpretación y lectura de las estadísticas, entre otras funcionalidades.

Se espera evaluar en terreno el sistema SIPLYC, durante el piloto de la investigación a realizar los meses de septiembre - octubre del presente año, en el cual se recogerá retroalimentación de los propios investigadores para mejorar el sistema.

A partir de la retroalimentación entregada en la evaluación, se propone mejorar la visualización de las evaluaciones, implementando una presentación de los resultados por palabra en un solo ítem, en vez de un ítem por cada evaluación independiente.

En el actual sistema se visualizan las estadísticas de evaluaciones de *Testing*, filtradas de acuerdo a los tres módulos (letras, palabras y vocalización), pero también son calculadas las estadísticas totales considerando estados de *Learning*, por lo que se espera implementar la visualización de estas últimas.

Se espera implementar una exportación directa de las estadísticas generadas por la aplicación del tutor a un documento .PDF, en forma similar a la funcionalidad del software *SIEL*.

Firebase permite un sencillo desarrollo de nuevas estadísticas, mediante conversión de la base de datos en JSON, a archivos CSV (valores separados por coma). Esto permite revisar detalles del contexto de las evaluaciones, mediante datos que nos son visualizados en la aplicación principal, como el "lado" en el que se muestra la palabra o el par "versus" el que fue evaluado. Se espera implementar un sistema para facilitar la obtención de estos archivos sin necesidad de ingresar a la consola de la base de datos. Se propone el desarrollo de una interfaz web, que permita acceder a la información almacenada desde el navegador.

Bibliografía

[1] Casales, J. C. (1989). “Psicología Social. Contribución a su estudio”. La Habana: Editorial Ciencias Sociales. [Disponible Online: Junio 2017] [Consultado en: http://www.psicologia-online.com/articulos/2005/comunicacion.shtml]

[2] Real Academia Española. (s.f.). Diccionario de la lengua española (22.a ed.). [Disponible Online: Junio 2017] [Consultado en: http://www.rae.es/]

[3] Beorlegui. C. (2006) Universidad de Deusto La Capacidad Lingüística del Ser Humano: Una Diferencia Cualitativa. Thémata. Revista de Filosofía. Núm. 37. 206. [Disponible Online: Junio 2017] [Consultado en: http://institucional.us.es/revistas/themata/37/11Beorlegui.pdf]

[4] Carreto. A. (1995) Lenguaje y Comunicación. Editorial Grijalbo. I Edición venezolana.

[5] Neneka P. (2002) “Lenguaje y comunicación: conceptos básicos, aspectos teóricos generales, características, estructura, naturaleza y funciones del lenguaje y la comunicación” N°2. Colección Minerva. Editor El Nacional, 2001.[Disponible Online: Junio 2017] [Consultado en: https://books.google.cl/books?id=5rqRZJjSZQsC]

[6] Boeree C. G. (2007) “Los Orígenes del Lenguaje” Traducción al Español: Fuenzalida C. Disponible en sitio web Shippensburg University of Pennsylvania. [Disponible Online: Junio 2017] [Consultado en: http://webspace.ship.edu/cgboer/origenesesp.html]

[7] “Estudio VTR Internet Segura: Un 40% de los niños de 5 años tiene un celular propio con acceso a Internet” (2015) T13 Artículo Online. Mayo 2015. [Disponible Online: Junio 2017] [Consultado en: http://www.t13.cl/noticia/nacional/estudio-indica-que-40-de-los-ninos-de-5-anos-tiene-un-celular-propio-con-acceso-a-internet]

[8] Sepúlveda P. (2015) “Dos de cada tres niños menores de 12 años del país es dueño de un celular”, Diario La Tercera. Octubre 2016. [Disponible Online: Junio 2017] [Consultado en: http://www.latercera.com/noticia/dos-tres-ninos-menores-12-anos-del-pais-dueno-celular/]

[9] GfK Adimark (2017), CHILE3D KIDS, Mayo 2017, [Disponible Online: Junio 2017] [Consultado en: http://www.chile3d.cl/kids3d.html]

[10] Park M. (2009) “Study: Want a smart baby? TV's not going to help” CNN, Reportaje, [Disponible Online: Junio 2017] [Consultado en: http://edition.cnn.com/2009/HEALTH/03/

03/babies.watch.TV/index.html?]

[11] Solon O. (2016) “Does spending too much time on smartphones and tablets damage kids’ development?” The Independent, Diario Online, Junio 2016. [Disponible Online: Junio 2017] [Consultado en: http://www.independent.co.uk/life-style/health-and-families/does-spending-too-much-time-on-smartphones-and-tablets-damage-kids-development-a7067261.html]

[12] Seattle Children’s Hospital, Research Foundation. “Studying how media use impacts brain development” Ramirez Lab, Sitio Web. [Disponible Online: Junio 2017] [Consultado en: http://www.seattlechildrens.org/research/integrative-brain-research/our-labs/ramirez-lab/]

[13] Cheung C., Bedford R., Saez De Urabain I., Karmiloff-Smith A., Smith T. (2017) “Daily touchscreen use in infants and toddlers is associated with reduced sleep and delayed sleep onset”. Cientific Reports, Abril 2017/04/13, Centre for Brain and Cognitive Development, Birkbeck, University of London, London, UK [Disponible Online: Junio 2017] [Consultado en: https://www.nature.com/articles/srep46104]

[14] Darling-Hammond L., Zielezinski M., Goldman S. (2014) “Using Technology to Support At-Risk Students' Learning” Septiembre 2014, Stanford University. [Consultado en: https://edpolicy.stanford.edu/sites/default/files/scope-pub-using-technology-report.pdf]

[15] Peña M., Guevara P., (2016) “Sistema Interactivo de Promoción del Lenguaje y la Comunicación” Formulario de Presentación Proyecto.

[16] Department of Education and Skills. (2011) “Literacy and Numeracy: For Learning and life, The National Strategy to Improve Literacy and Numeracy among Children and Young People” [Disponible Online: Junio 2017] [Consultado en: https://www.education.ie/en/Publications/Policy-Reports/lit_num_strategy_full.pdf]

[17] US NIH NIDCD “Etapas del desarrollo del habla y el lenguaje” (s.f.) [Disponible Online: Junio 2017] [Consultado en: https://www.nidcd.nih.gov/es/espanol/etapas-del-desarrollo-del-habla-y-el-lenguaje]

[18] Pinel. J., “Biopsicología”. (2007) Editorial Pearson Addison Wesley. 6ta. Edición. Madrid, España.

[19] Castañeda P. F. (1999) “El Lenguaje Verbal del Niño: ¿Cómo estimular, corregir y ayudar para que aprenda a hablar bien?” Universidad Nacional Mayor de San Marcos Fondo Editorial.

Sitio Web Comunidad Andina. Biblioteca Digital Andina. [Disponible Online: Junio 2017] [Consultado en: http://www.comunidadandina.org/BDA/docs/PE-EDU-0003.pdf]

[20] Children's Speech Therapy NI “Fun Speech Therapy Apps 4 Kids” (s.f.) [Disponible Online: Junio 2017] [Consultado en: http://www.childrensspeechtherapyni.com/pages/index.asp?title=Fun_Apps_4_Kids]

[21] Little Bee Speech: Apps for Speech & Language. (s.f.) “Articulation Station Español” Sitio web. [Disponible Online: Junio 2017] [Consultado en: http://littlebeespeech.com/articulation_station_spanish.php]

[22] STAYTOONED (2011) "iTouchiLearn Words for Preschool Kids App" Canal de YouTube oficial del desarrollador. [Disponible Online: Junio 2017] [Consultado en: https://www.youtube.com/watch?v=986gH4YsnQg]

[23] King A., Brady K., Voreis G. (2017) “It’s a blessing and a curse: Perspectives on tablet use n children with autism spectrum disorder”. Enero de 2017. SOIU: Southern Illinois University Edwardsville Research. Department of Special Education and Communication Disorders at Southern Illinois University. [Disponible Online: Junio 2017] [Consultado en: http://journals.sagepub.com/doi/full/10.1177/2396941516683183]

[24] GraphoGame “Research/Recent Publications 2007-2015”. University of Jyväskylä [Disponible Online: Junio de 2017] [Consultado en: http://info.graphogame.com/research/]

[25] Bus A., Takacs Z., Swart E. (2015) “Benefits and Pitfalls of Multimedia and Interactive Features in Technology-Enhanced Storybooks: A Meta-Analysis” Review of Educational Research. National Center for Biotechnology Information, U.S. National Library of Medicine. [Disponible Online: Junio 2017] [Consultado en: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4647204/]

[26] Cid R. (2015), “Sistema interactivo de Estimulación del lactante para plataformas Android” Informe de Memoria de Título para optar al título de Ingeniero Civil Biomédico. Universidad de Concepción, Chile. Marzo de 2015.

[27] Android Developers, Documentación Oficial. “Android Studio” [Disponible Online: Junio 2017] [Consultado en: https://developer.android.com/studio/index.html]

[28] Miller P. “Google is adding Kotlin as an official programming language for Android development” The Verge, 17 de mayo de 2017. [Disponible Online: Junio 2017] [Consultado

en: https://www.theverge.com/2017/5/17/15654988/google-jet-brains-kotlin-programming-language-android-development-io-2017]

[29] Android Developers. Documentación Oficial. “Actividades” [Disponible Online: Junio 2017] [Consultado en: https://developer.android.com/guide/components/activities.html]

[30] Android Developers. Documentación Oficial. “Diseños” [Disponible Online: Junio 2017] [Consultado en: https://developer.android.com/guide/topics/ui/declaring-layout.html]

[31] Android Developers. Documentación Oficial. “Gestores de Eventos” [Disponible Online: Junio 2017] [Consultado en: https://developer.android.com/guide/topics/ui/ui-events.html

[32] Revelo J. (2015). ¿Cómo Sincronizar Sqlite Con Mysql En Android? Blog de desarrollos en Android: Hermosa Programación [Disponible Online: Junio 2017] [Consultado en: www.hermosaprogramacion.com/2015/07/como-sincronizar-sqlite-con-mysql-en-android/]

[33] MySQL, Sitio Web Oficial. [Disponible Online: Junio 2017] [Consultado en: https://www.mysql.com/products/community/]

[34] Firebase Realtime Database, Sitio Web Oficial, Documentación. [Disponible Online: Junio 2017] [Consultado en: https://firebase.google.com/docs/database/]

[35] Google Cloud Platform. “Mobile App Backend Services. – Firebase” [Disponible Online: Junio 2017] [Consultado en: https://cloud.google.com/solutions/mobile/mobile-app-backend-services#firebase-appengine-standard]

[36] “The base difference between SQL and NoSQL databases. MS SQL Server vs. MongoDB”. [Disponible Online: Junio 2017] [Consultado en: http://sql-vs-nosql.blogspot.cl/2013/10/the-base-difference-between-sql-and.html]

[37] Firebase Realtime Database, Sitio Web Oficial, Documentación, Android Setup [Disponible Online: Junio 2017] [Consultado en: https://firebase.google.com/docs/android/setup/]

[38] Firebase Blog: “The 2^120 Ways to Ensure Unique Identifiers” [Disponible Online: Junio 2017] [Consultado en: https://firebase.googleblog.com/2015/02/the-2120-ways-to-ensure-unique_68.html]

[39] Firebase Realtime Database, Documentación, Retrieve Data. [Disponible Online: Junio 2017] [Consultado en: https://firebase.google.com/docs/database/admin/retrieve-data]

[40] Firebase Realtime Database, Documentación, Offline Capabilities. [Disponible Online: Junio 2017] [Consultado en: https://firebase.google.com/docs/database/android/offline-capabilities]

[41] Android Developers. Documentación. Bluetooth API [Disponible Online: Junio 2017] [Consultado en: https://developer.android.com/guide/topics/connectivity/bluetooth.html]

[42] MockingBot (Interfaz web para desarrollo de prototipos de apps) [Disponible Online: Junio 2017] [Consultado en: https://mockingbot.com/]

[43] Jahoda P. MPAndroidChar. Documentacion. [Disponible Online: Junio 2017] [Consultado en: GitHubhttps://github.com/PhilJay/MPAndroidChart]

Anexo A. Prototipo v/s App Final

A ..1 Actividad Secundaria

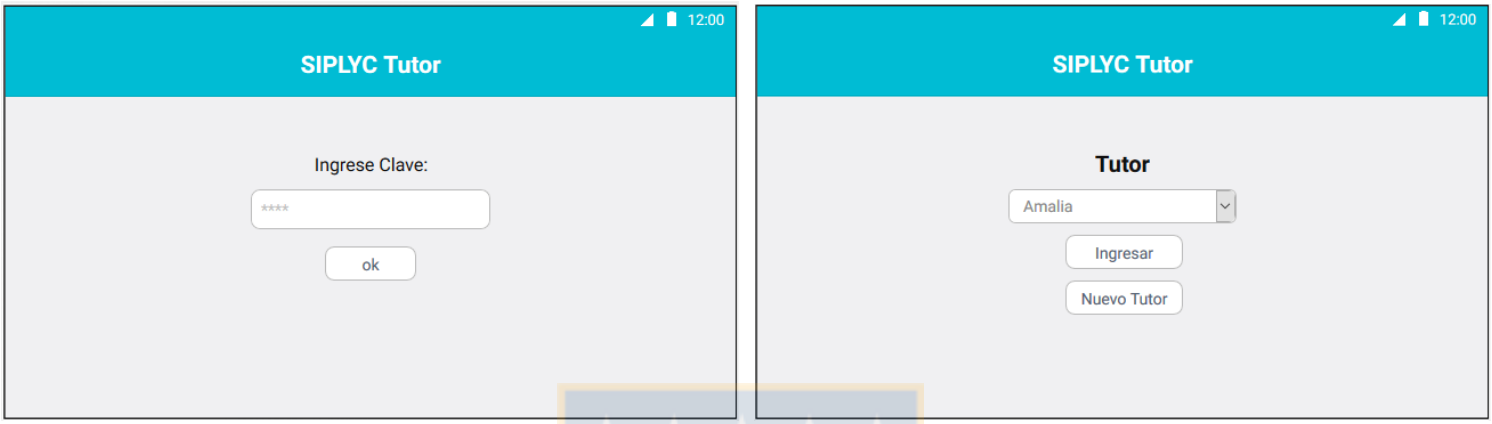


Fig. A.1. Actividad de Login de Prototipo en MockingBot

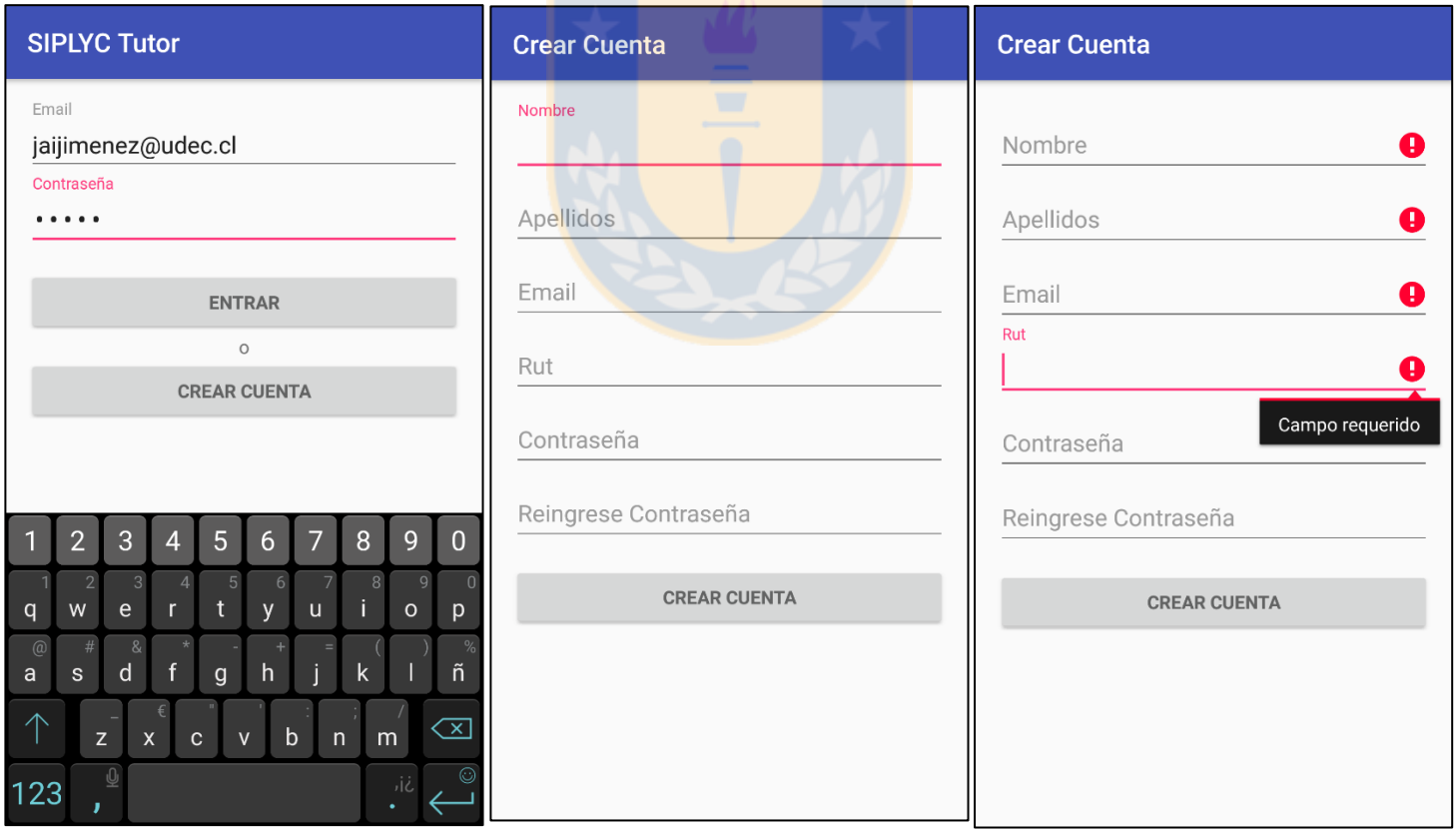


Fig. A.2. Actividad de Login y de registro implementada.

Como se observa en la Fig. A.2, se implementó una actividad de registro. Los campos pueden identificar errores bajo ciertas condiciones. Se solicita un email que al menos contenga el carácter "@", una contraseña de al menos 5 caracteres y todos los campos son requeridos.

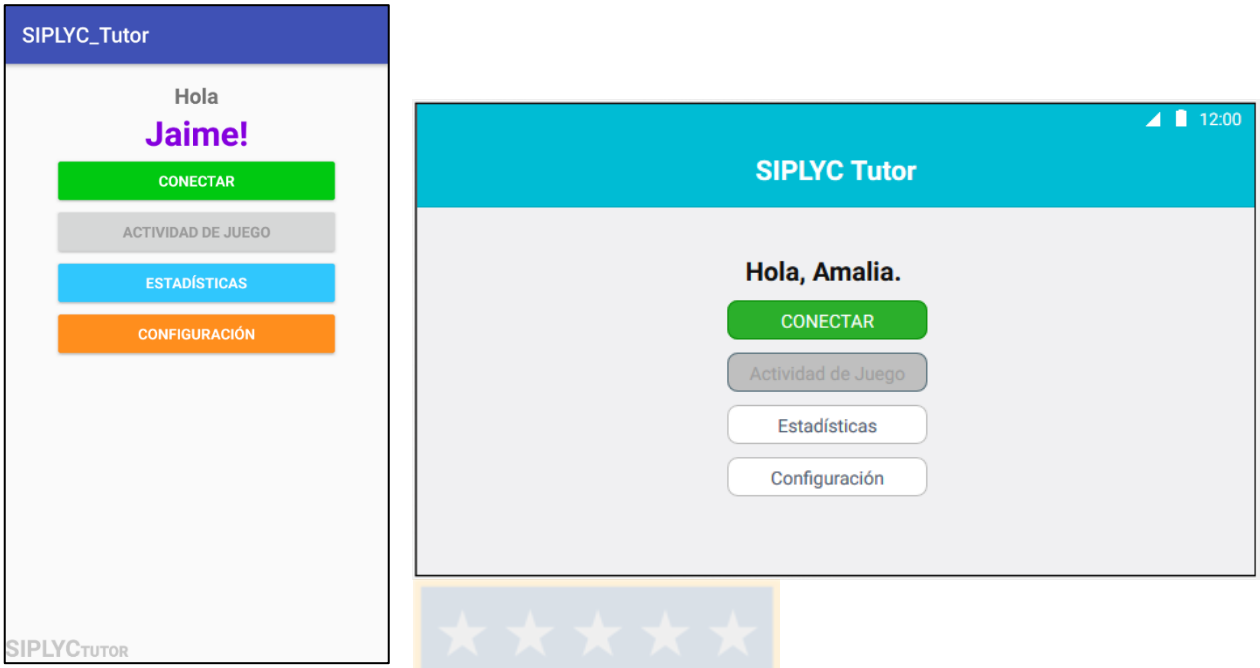


Fig. A.3. Main Activity, previo a conexión Bluetooth

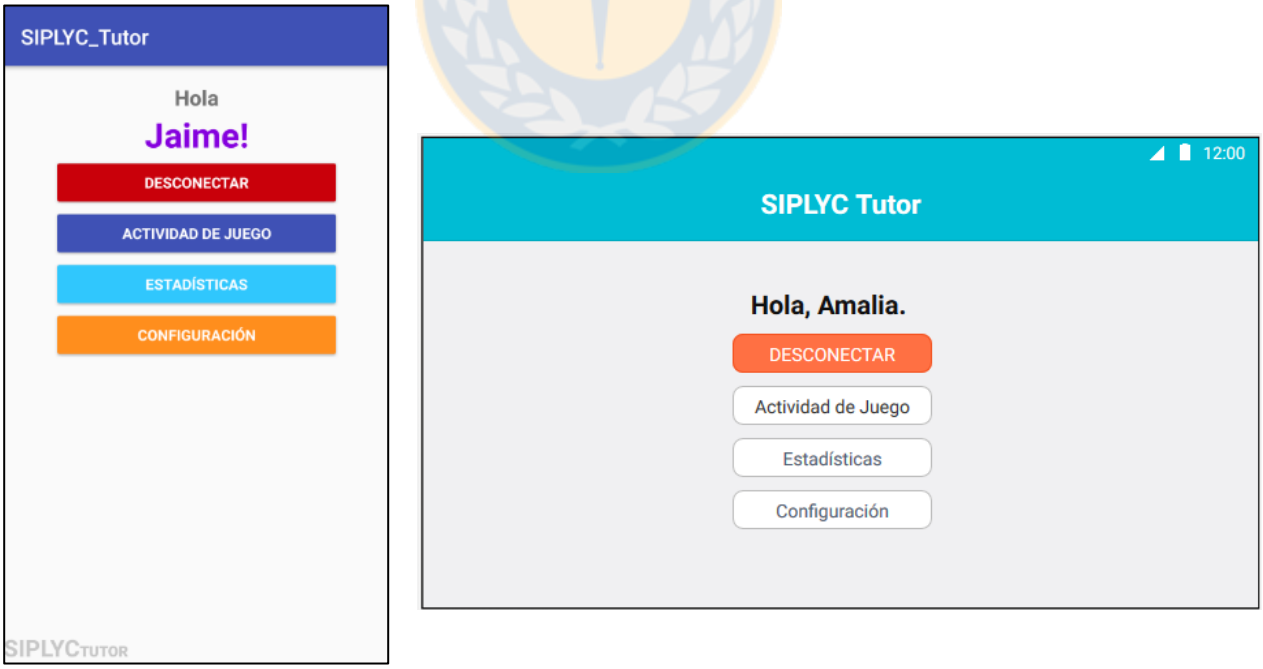


Fig. A.4. Main Activity, posterior a conexión Bluetooth

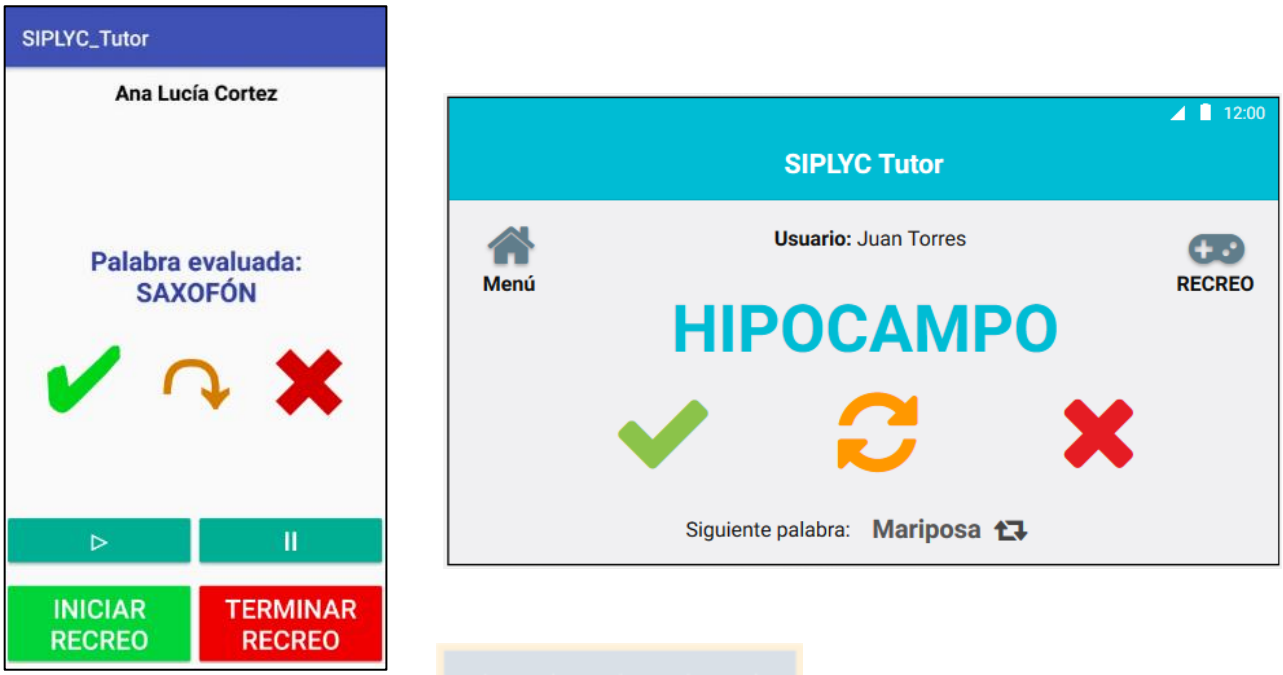


Fig. A.5. Actividad de Juego.

Fig. A.6. Prototipo no funcional de visualización de estadísticas.

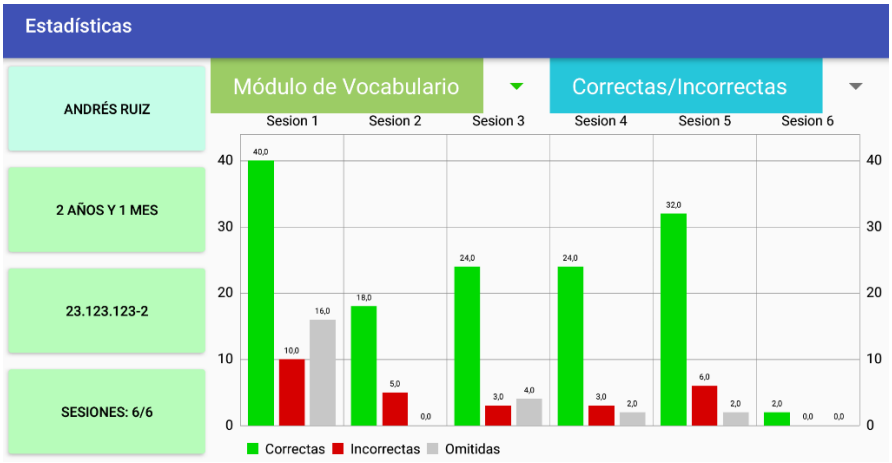


Fig. A.7. Visualización de estadísticas implementada.

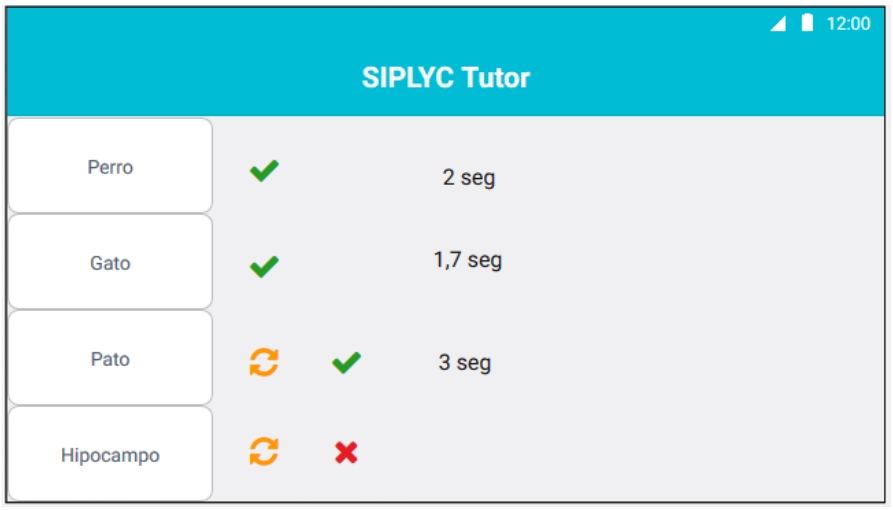


Fig. A.8. Detalle por sesión en prototipo.

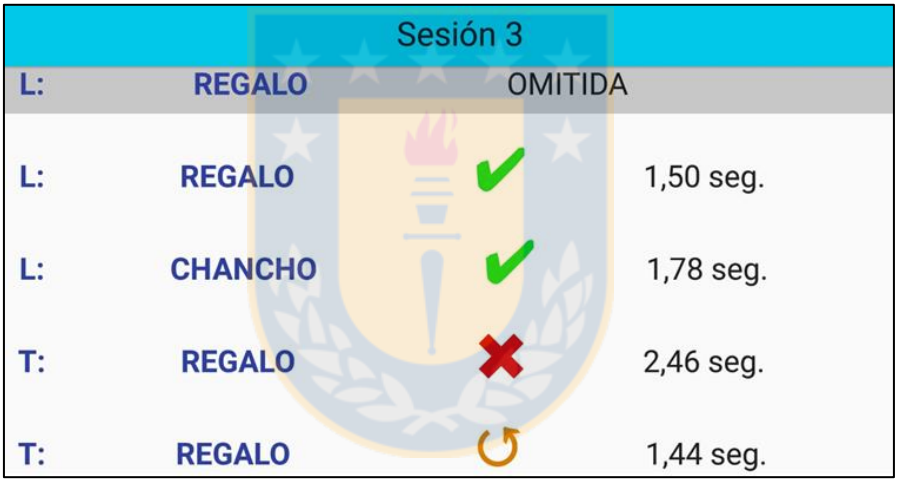


Fig. A.9. Detalle por sesión en aplicación implementada.