**Medina Gabriel Andrés, Solano Yirlan Andrey, Urrutia Juan Pablo**

**No. de Equipo Trabajo: X**

# [[1]](#footnote-1) INTRODUCCIÓN

Según datos del Instituto Nacional para Sordos (INSOR), en

Colombia para el año 2018 había 543.000 personas sordas en

el territorio Nacional, de los cuales cerca de 64.482 personas

estaban en Bogotá, esto basado en los datos proyectados de la

Secretaría de Planeación Distrital para el mismo año [1].

La sordera representa un reto en la comunicación de las personas que la padecen, pues al no contar con una competencia lingüística tienen que recurrir a factores extralingüísticos, como mímicas o gestos, para darse a entender de una manera elemental. Es por esto que, en 1996, el Estado Colombiano reconoce La Lengua de Señas Colombiana (LSC) como lengua propia de la población sorda, para que junto con otras competencias como la escritura se llegue a un nivel más elaborado de comunicación

# DESCRIPCIÓN DEL PROBLEMA A RESOLVER

La gran mayoría de las personas sin ninguna discapacidad auditiva desconocen el lenguaje de señas y su importancia, de hecho, un gran porcentaje de la misma población sorda no sabe LSC y son analfabetas. La educación es uno de los grandes restos que tienen las personas sordas, y cada año se ve más excluidos según datos de del Sistema Integrado de Matricula (SIMAT) para el 2013 habían cerca de 10464 estudiantes sordos matriculados en establecimientos educativos, sin embargo, para el año 2018 esta cifra disminuyó a 6160 estudiantes.[2]

Es por lo **nuestro objetivo es crear** un aplicativo móvil, que consta de un diccionario de Lenguaje de Señas Colombiano que permita a las personas no sordas tener una herramienta que dado el caso les permita tener una conversación con una persona con discapacidad auditiva. Asimismo, se plantea que dicha aplicación cuente con una herramienta practica en donde se fortalezca el aprendizaje de las palabras más comunes, utilizando la técnica de la repetición espaciada. Y la posibilidad de que usuario agregue o elimine las palabras que tenga la necesidad o intensión de aprender en la herramienta de práctica.

# usuarios DEL PRODUCTO DE SOFTWARE

Con el fin de promover la aplicación en distintos ámbitos educativos y que esta sirva como una herramienta tanto para estudiantes autodidactas, como para las instituciones o profesores, se establecieron tres roles dentro de la aplicación.

* Visitante: Usuario que puede usar la aplicación sin la necesidad de registrarse, pero no hacer un seguimiento de su avance ni estadísticas.
* Usuario autentificado: Luego de hacer el registro dentro de una de las pantallas de la aplicación, esta puede acceder a estadísticas y graficas que le den retroalimentación sobre el proceso de aprendizaje espaciado.
* Administrador: Los usuarios con permisos de administrador pueden añadir, editar y eliminar palabras del diccionario. Esto con el fin de que si un profesor quiere que sus estudiantes aprendan palabras específicas del LSC y que no estén añadidas puedan incluirlas.

# REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

Los requerimientos funcionales de un sistema son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

Entre los posibles requerimientos funcionales de un sistema, se incluyen:

* Descripciones de los datos a ser ingresados en el sistema.
* Descripciones de las operaciones a ser realizadas en cada pantalla que se presenta.
* Descripción de los flujos de trabajo realizados por el sistema.
* Descripción de los reportes del sistema y otras salidas.
* Definición de quiénes pueden ingresar datos en el sistema.

Diccionario LSC- Aprende Lenguaje de Señas

De esta manera, se deben describir las interacciones que tendrán los usuarios con el software.

Cada funcionalidad se debe especificar así:

## Nombre de la funcionalidad

En el título de la funcionalidad, se recomienda utilizar nombres muy descriptivos para cada funcionalidad. No limitarse a nombrarlas “Funcionalidad 1”, en cambio usar por ejemplo: “Autorización de pedido de compra”.

## Descripción:



## Descripción breve de la funcionalidad.

* *Acciones iniciadoras y comportamiento esperado*:

Secuencia de acciones del usuario y respuestas esperadas del programa para esta funcionalidad.

*Requerimientos funcionales*:

Lista detallada de los requerimientos funcionales asociados a esta funcionalidad.

Para cada requerimiento funcional se establece cómo debe mostrarse el software y cuáles comportamientos debe desempeñar para que el usuario pueda realizar la función que necesita.

Es recomendable prever y describir cómo debe responder el software ante condiciones de error y entradas de datos inválidas.

Las funcionalidades mínimas sobre los datos que se manejen deben prever operaciones de:

* Creación
* Actualización
* Eliminación
* Consulta total de los datos
* Búsqueda parcial de datos
* Ordenamiento
* Almacenamiento

Aunque en otros cursos se estudian estrategias de organización y almacenamiento, en este curso el almacenamiento se requiere principalmente para facilitar las pruebas del prototipo de software. También, para facilitar su implementación, se deja abierta la opción a que se apoyen en el uso de sistemas manejadores de bases de datos, o se haga almacenamiento por archivo, de objetos u otra estrategia que les convenga, siempre que se garantice la implementación y uso de las estructuras de datos vistas en clase en la memoria en tiempo de ejecución.

IMPORTANTE: En cada una de las entregas para reportar el avance en el desarrollo del proyecto, se especificarán las funcionalidades mínimas y las estructuras de datos mínimas requeridas que se deben implementar. También, se debe presentar un análisis (comparativo y asintótico) breve de la eficiencia de las estructuras de datos usadas.

NOTA: En el siguiente enlace web (URL) puede encontrar una explicación de cómo diferenciar Requisitos Funcionales de los No Funcionales  
<https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>

# DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

Se plantean varias pantallas básicas para la aplicación.

Primero La pantalla de inicio de sesión, donde se definen los 3 roles del usuario. Le botón de registro lo lleva a la pantalla para registrar un usuario y agregarlo a la base de datos.

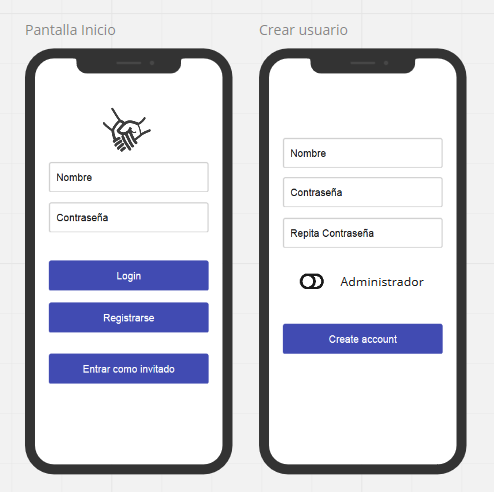


Ilustración 1Pantalla Inicio- Pantalla Crear Usuario

Ya con el usuario registrado y habiendo iniciado sesión se accede a las siguientes pantallas conectadas entre las tres por medio de la barra inferior.

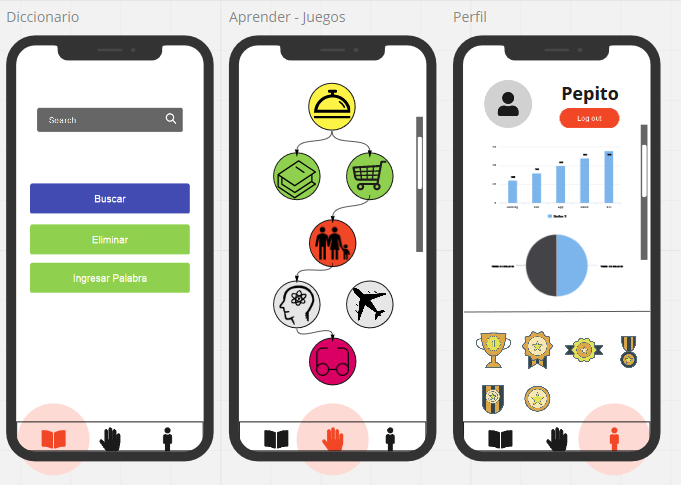


Ilustración 2Pantallas Diccionario, Aprender y Perfil

En la primera pantalla, la de la izquierda, es el diccionario. Acá podremos buscar las palabras que están en la base de datos, los botones verdes sólo serán **visibles** para usuarios con rol de administrador. Por otro lado, el botón azul será visible para los 3 roles.

En la pantalla del medio “Aprender-Jugar”, encontraremos las actividades basadas en aprendizaje gamificado y aprendizaje por medio de repetición espaciada. Dado el objetivo académico de este proyecto, esta pantalla será una de las ultimas de implementar.

En la pestaña de Usuario encontraremos graficas que muestran el progreso del perfil, un botón para salir de la sesión y una lista de trofeos que se obtendrán de la pantalla de aprendizaje.

# Entornos de desarrollo y de operación

La aplicación se desarrollará en la última versión Android Studio, en su configuración con Java, el manejo de las versiones se hará en Git, en preferencia se usará Github Deskstop o SourceTree.

El software operará en dispositivos Android a partir de la versión 5.0 “Lollipop”, en cuanto a requerimientos de hardware, no se hará un uso exigente de los mismos por tanto no existe ningún requerimiento mínimo más allá de la versión de Android.

# PROTOTIPO DE SOFTWARE INICIAL

Como fue mencionado anteriormente, el control de versiones del proyecto se hace en Git, y el repositorio del proyecto es:

<https://github.com/andreysolano/Diccionario-LSC>



# 

# IMPLEMENTACIÓN Y aplicación de las estructuras de datos

En este prototipo se tuvo 2 grandes retos: manejar los datos para el ingreso de usuarios, tales como nombre de usuario, contraseña y demás atributos que pueda tener un potencial usuario; y almacenar los datos de las palabras, como el contenido, el identificador, entre otras.

Esto se logró a través de la creación de las clases pertinentes (Usuario/Palabra) para guardar internamente ese tipo de información. A su vez, se implementaron estructuras de datos para almacenar las instancias de estas clases.

Las estructuras de datos que se emplearon fueron:

* Listas ordenadas doblemente enlazadas para almacenar las palabras.
* Listas simplemente enlazadas para almacenar usuarios.

Ambas estructuras fueron dotadas de métodos de inserción, eliminación, búsqueda y otras específicas de cada implementación.

Para la lista ordenada se implementó un mecanismo interno con el objetivo de que cada dato insertado fuese organizado en orden lexicográfico, dependiendo de la palabra que es ingresada a la lista, de esta forma se puede presentar la información de manera más clara al consultar la lista de palabras disponibles.

Para la lista simplemente enlazada, se dotó de los métodos comunes de una lista de este tipo, como lo son el *addAfter*, *addBefore*, *pushFront, pushBack, peekFront, peekBack, popFront, popBack, find,* entre otros.

# PRUEBAS DEL PROTOTIPO Y ANÁLISIS COMPARATIVO

Se deben realizar y documentar las pruebas del prototipo para algunos ejemplos (casos) de prueba para las funcionalidades que tomen más tiempo y realizar un análisis comparativo así:

* Escoger entre tres y cinco funcionalidades que sean las de mayor costo computacional en tiempo;
* Para cada funcionalidad se deben realizar pruebas para varios tamaños de datos de prueba (n), por lo menos para los siguientes valores:
  + 10 mil datos,
  + 100 mil datos,
  + 1 millón de datos,
  + 10 millones de datos, y
  + 100 millones de datos,
* Hacer una tabla comparativa de los tiempos que toma realizar las funcionalidades consideradas para los diferentes tamaños de los datos de prueba.
* Determine y grafique el correspondiente análisis asintótico comparativo entre las estructuras implementadas y su respectiva complejidad, de acuerdo con las pruebas realizadas. Para esto debe usar, por lo menos, la notación O grande (Big O).

# roles y actividades

Se debe describir de manera concisa los roles asignados y las respectivas actividades realizadas por cada uno los integrantes del equipo durante el desarrollo de esta entrega. Para esto, tenga en cuenta los siguientes roles positivos, que puede aplicar durante el desarrollo del proyecto; los mismos deberán ser ‘rotados’ entre todos los integrantes del equipo en las diferentes entregas.

|  |  |
| --- | --- |
| **ROL** | **Actividades fundamentales** |
| Líder/esa | Consultar a los otros miembros del equipo, atento que la información sea constante para todos. Aportar con la organización y plan de trabajo. |
| Coordinador/a: | Mantener el contacto entre todos,  Programar y agendar y reuniones, Ser facilitador con el acceso a los recursos. |
| Experto: | Líder técnico que propende por coordinar las funciones y actividades operativas. |
| Investigador | Consultar otras fuentes. Propender por resolver inquietudes comunes para todo el equipo. |
| Observador | Siempre está atento en el desarrollo del proyecto y aporta en el momento apropiado cuando se requiera apoyo adicional por parte del equipo. |
| Animador/a | Energía positiva, motivador en el grupo. |
| Secretario | Se convierte en un facilitador de la comunicación en el grupo. Documenta (actas) de los acuerdos/compromisos realizados en las reuniones del equipo. |
| Técnico | Aporta técnicamente en el desarrollo del proyecto. |

Puede utilizar la siguiente tabla para definir los integrantes del grupo, los roles asignados y el listado de actividades durante el desarrollo de la entrega:

|  |  |  |
| --- | --- | --- |
| **INTEGRANTE** | **ROL(ES)** | **ACTIVIDADES REALIZADAS (Listado)** |
| Gabriel Medina |  |  |
|  |
|  |  |
|  |
| Andrey Solano |  |  |
|  |
|  |  |
|  |
| Juan Urrutia |  |  |
|  |
|  |  |
|  |

# DIFICULTADES Y LECCIONES APRENDIDAS

Ya que se está desarrollando una aplicación móvil, se tuvo que aprender a usar las nuevas herramientas como lo son Android Studio, Gradle, entre otros. Además, dado que las pruebas de rendimiento requieren de grandes cantidades de datos, fue necesario aprender sobre el manejo e integración de bases de datos, lo que resultó un poco problemático en un comienzo.

# REFERENCIAS BIBLIOGRÁFICAS

[1] Insor.gov.co. 2020. Preguntas Frecuentes – INSOR – INSORInstituto Nacional Para Sordos. [online] Available at:<http://www.insor.gov.co/home/servicio-alciudadano/preguntas-frecuentes/> [Accessed 8 September 2020].

[2]2019. Plan Institucional 2019-2022. online] Available at: <http://www.insor.gov.co/home/descargar/plan\_estrategico\_NSOR\_2019\_2022V1.pdf [Accessed 8 September 2020].

1. [↑](#footnote-ref-1)