Neiva, 16 de enero de 2024

Señores

CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD SURCOLOMBIANA

Ciudad

El (Los) suscrito(s):

Karen Emilia Meneses Orjuela, con C.C. No.1003952941,

Alejandro Palacios Figueroa, con C.C. No.1003801943,

_____, con C.C. No._____,

_____, con C.C. No._____,

Autor(es) de la tesis y/o trabajo de grado o _____

titulado DISEÑO DE UNA APLICACIÓN MÓVIL PARA INDICAR EL VALOR NUTRICIONAL DE LOS ALIMENTOS Y MOSTRAR ADVERTENCIA SEGÚN EL PLAN ALIMENTICIO

_____

_____

presentado y aprobado en el año 2023 como requisito para optar al título de Ingeniero Electrónico;

Autorizo (amos) al CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN de la Universidad Surcolombiana para que, con fines académicos, muestre al país y el exterior la producción intelectual de la Universidad Surcolombiana, a través de la visibilidad de su contenido de la siguiente manera:

- Los usuarios puedan consultar el contenido de este trabajo de grado en los sitios web que administra la Universidad, en bases de datos, repositorio digital, catálogos y en otros sitios web, redes y sistemas de información nacionales e internacionales “open access” y en las redes de información con las cuales tenga convenio la Institución.

- Permita la consulta, la reproducción y préstamo a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato Cd-Rom o digital desde internet, intranet, etc., y en general para cualquier formato conocido o por conocer, dentro de los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia.

- Continúo conservando los correspondientes derechos sin modificación o restricción alguna; puesto que, de acuerdo con la legislación colombiana aplicable, el presente es un acuerdo jurídico que en ningún caso conlleva la enajenación del derecho de autor y sus conexos.

De conformidad con lo establecido en el artículo 30 de la Ley 23 de 1982 y el artículo 11 de la Decisión Andina 351 de 1993, “Los derechos morales sobre el trabajo son propiedad de los autores” , los cuales son irrenunciables, imprescriptibles, inembargables e inalienables.

EL AUTOR/ESTUDIANTE:

Karen Emilia Meneses Orjuela

Firma:_____

EL AUTOR/ESTUDIANTE:

Alejandro Palacios Figueroa

Firma: Alejandro Palacios F.

TÍTULO COMPLETO DEL TRABAJO: DISEÑO DE UNA APLICACIÓN MÓVIL PARA INDICAR EL VALOR NUTRICIONAL DE LOS ALIMENTOS Y MOSTRAR ADVERTENCIA SEGÚN EL PLAN ALIMENTICIO

AUTOR O AUTORES:

Primero y Segundo Apellido	Primero y Segundo Nombre
Meneses Orjuela Palacios Figueroa	Karen Emilia Alejandro

DIRECTOR Y CODIRECTOR TESIS:

Primero y Segundo Apellido	Primero y Segundo Nombre
Quintero Polanco	Jesús David

ASESOR (ES):

Primero y Segundo Apellido	Primero y Segundo Nombre

PARA OPTAR AL TÍTULO DE: Ingeniero electrónico

FACULTAD: **Ingeniería**

PROGRAMA O POSGRADO: Electrónica

CIUDAD: Neiva **AÑO DE PRESENTACIÓN:2023** **NÚMERO DE PÁGINAS: 71**

TIPO DE ILUSTRACIONES (Marcar con una **X**)**:**

Diagramas **X**__Fotografías____Grabaciones en discos____Ilustraciones en general **X**__Grabados____ Láminas____Litografías____Mapas__ Música impresa___ Planos____Retratos____ Sin ilustraciones____Tablas o Cuadros **X**_____

SOFTWARE requerido y/o especializado para la lectura del documento: Android Studio

MATERIAL ANEXO: Aplicación móvil QUIMO

PREMIO O DISTINCIÓN *(En caso de ser LAUREADAS o Meritoria)*:

PALABRAS CLAVES EN ESPAÑOL E INGLÉS:

Español	Inglés	Español	Inglés
1. Tabla de composición de alimentos colombianos (TCAC)	"Tabla de composición de alimentos colombianos" (TCAC)	6. Android Studio	Android Studio
2. JAVA	JAVA	7.	
3. Firebase	Firebase	8.	
4. Google	Google	9.	
5. App script	App script	10.	

RESUMEN DEL CONTENIDO: (Máximo 250 palabras)

La aplicación diseñada representa una herramienta fundamental para que los usuarios mejoren su dieta y adopten hábitos alimenticios conscientes. Permite el almacenamiento independiente de información sobre las tres comidas diarias, ofreciendo una visión detallada de los alimentos consumidos y su composición nutricional. Además, brinda un resumen diario con advertencias basadas en estándares de una dieta saludable, fomentando así un control nutricional preciso. La proporción de nutrientes ingeridos se calcula en base a la cantidad específica ingresada por el usuario, utilizando la tabla de composición de alimentos colombianos como referencia.

Esta efectiva gestión de la dieta es posible gracias a la comunicación establecida entre el dispositivo del usuario y el servidor, facilitada por los servicios de Firebase. La sincronización de datos en tiempo real garantiza que la información esté actualizada constantemente, permitiendo la toma de decisiones informada y alineada con los objetivos nutricionales. Para asegurar un funcionamiento óptimo, cada funcionalidad de la aplicación fue sometida a rigurosas pruebas en diversas condiciones. Este proceso de validación identificó posibles fallos y puntos de mejora, asegurando que cada característica opere de manera correcta y coherente, proporcionando una experiencia de usuario fluida y satisfactoria.

ABSTRACT: (Máximo 250 palabras)

The designed application represents a fundamental tool for users to improve their diet and adopt conscious eating habits. It allows the independent storage of information on the three daily meals, providing a detailed overview of the food consumed and its nutritional composition. In addition, it provides a daily summary with warnings based on healthy dietary standards, thus promoting accurate nutritional control. The proportion of nutrients ingested is calculated based on the specific amount entered by the user, using the Colombian food composition table as a reference.

This effective diet management is possible thanks to the communication established between the user's device and the server, facilitated by Firebase services. Real-time data synchronization ensures that information is constantly updated, enabling informed decision making aligned with nutritional objectives. To ensure optimal performance, each functionality of the application was subjected to rigorous testing under various conditions. This validation process identified potential bugs and points for improvement, ensuring that each feature operates correctly and consistently, providing a smooth and satisfying user experience.

APROBACION DE LA TESIS

Nombre Jurado: Johan Julián Molina Mosquera

Firma 

Nombre Jurado: Martin Diomedes Bravo Obando

Firma: 

DISEÑO DE UNA APLICACIÓN MÓVIL PARA INDICAR EL VALOR NUTRICIONAL DE LOS ALIMENTOS Y MOSTRAR ADVERTENCIA SEGÚN EL PLAN ALIMENTICIO

Karen Emilia Meneses Orjuela
20182172957

Alejandro Palacios Figueroa
20182172738

UNIVERSIDAD SURCOLOMBIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA ELECTRÓNICA
NEIVA – HUILA
2023

DISEÑO DE UNA APLICACIÓN MÓVIL PARA INDICAR EL VALOR NUTRICIONAL DE LOS ALIMENTOS Y MOSTRAR ADVERTENCIA SEGÚN EL PLAN ALIMENTICIO

Línea de la propuesta: DESARROLLO DE SOFTWARE PARA APLICACIÓN MOVIL

Director y proponente de la idea:
Msc. Jesús David Quintero Polanco

UNIVERSIDAD SURCOLOMBIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA ELECTRÓNICA
NEIVA – HUILA
2023

TABLA DE CONTENIDO

Notas de aceptación

_____
_____
_____
_____
_____
_____
_____

_____
Firma del director de Tesis

_____
Firma del Jurado

_____
Firma del Jurado

Neiva, octubre de 2023

A mi madre Marcela, mi polo a tierra, la mujer que con su esfuerzo, amor y bondad de la mano de Dios y en equilibrio con el universo formaron la persona que soy hoy en día, la mujer que nunca tuvo ni una pisca de duda en mí, gracias por ser mi incondicional.

A mi amigo fiel Hades, mi compañero en cada una de mis madrugadas de estudio.
A mi hermano Mauricio y mi sobrina Evelyn, quienes me consideran excepcional y me ayudan a sentir que lo soy.
A mi compañero de tesis y amigo Alejandro, mi apoyo en cada uno de los días buenos y no tan buenos por permitirme vivir todo este proceso universitario con él.
A César, el refugio que encontré en una persona por brindarme una sonrisa y un abrazo calido cada que lo necesite.
A las personas maravillosas que conocí por la danza gracias a la Universidad, que me brindaron durante muchos semestres el equilibrio para vivir motivada.
Por último y no menos importante, a Miguel, Juan José, Jhan, Daniel, Camilo, Andrés y cada una de los amigos que conocí gracias a la ingeniería, por aportar felicidad y conocimiento en este proceso y en cada una de las vivencias compartidas.

Emilia

A mis madre por esforzarse durante tanto tiempo con la meta de educar a un gran ser humano que tenga un gran futuro por delante.
A mi padre por siempre apoyarme y siempre tener esa confianza depositiada en mi y en cada uno de mis actos.
A mi hermano por ser un apoyo en este proceso y darme tiempos de libertad de mis actividades.

A Emilia que mas que mi compañera de tesis una gran amiga que me ha ayudado a ser la persona que soy hoy de la que me siento muy orgulloso.
A Alejandra y Daniel, dos amigos que me deja la universidad llenandome de experiencias y recuerdos que se quedaran en mi toda la vida.

A mis amigos Ana, Angel, Camilo, Diego, Jhan y Jorge que hicieron de este proceso mucho mas ameno y por siempre tener los mejores deseos.
Y a cada una de las personas que fueron parte de este proceso llamado universidad.

Alejandro

AGRADECIMIENTOS

Como primera instancia agradecer a Dios por permitirnos llegar hasta este punto del recorrido como estudiantes de ingeniería electrónica, y guiar nuestro camino para lograr superar y disfrutar cada una de las etapas vividas en este proceso.

Agradecemos a nuestros padres por el apoyo durante este recorrido, por guiarnos hasta este punto. Agradecemos a nuestros amigos por hacer parte fundamental del día a día durante los últimos años, por cada una de las experiencias vividas juntos.

Por último, estamos muy agradecidos con cada uno de los docentes que nos impartieron su conocimiento y aportaron su intelecto para crecer como futuros profesionales

Contenido

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

La aplicación diseñada representa una herramienta fundamental para que los usuarios mejoren su dieta y adopten hábitos alimenticios conscientes. Permite el almacenamiento independiente de información sobre las tres comidas diarias, ofreciendo una visión detallada de los alimentos consumidos y su composición nutricional. Además, brinda un resumen diario con advertencias basadas en estándares de una dieta saludable, fomentando así un control nutricional preciso. La proporción de nutrientes ingeridos se calcula en base a la cantidad específica ingresada por el usuario, utilizando la tabla de composición de alimentos colombianos como referencia.

Esta efectiva gestión de la dieta es posible gracias a la comunicación establecida entre el dispositivo del usuario y el servidor, facilitada por los servicios de Firebase. La sincronización de datos en tiempo real garantiza que la información esté actualizada constantemente, permitiendo la toma de decisiones informada y alineada con los objetivos nutricionales. Para asegurar un funcionamiento óptimo, cada funcionalidad de la aplicación fue sometida a rigurosas pruebas en diversas condiciones. Este proceso de validación identificó posibles fallos y puntos de mejora, asegurando que cada característica opere de manera correcta y coherente, proporcionando una experiencia de usuario fluida y satisfactoria.

ABSTRACT

The designed application represents a fundamental tool for users to improve their diet and adopt conscious eating habits. It allows the independent storage of information on the three daily meals, providing a detailed overview of the food consumed and its nutritional composition. In addition, it provides a daily summary with warnings based on healthy dietary standards, thus promoting accurate nutritional control. The proportion of nutrients ingested is calculated based on the specific amount entered by the user, using the Colombian food composition table as a reference.

This effective diet management is possible thanks to the communication established between the user's device and the server, facilitated by Firebase services. Real-time data synchronization ensures that information is constantly updated, enabling informed decision making aligned with nutritional objectives. To ensure optimal performance, each functionality of the application was subjected to rigorous testing under various conditions. This validation process identified potential bugs and points for improvement, ensuring that each feature operates correctly and consistently, providing a smooth and satisfying user experience.

1. INTRODUCCION

1.1 Antecedentes

La obesidad y el sobre peso son dos términos diferentes con un concepto muy importante en común como lo es el sobre pasar el límite de peso saludable en consideración de la estatura del individuo, donde la obsedida es el sobrepasar este, pero por los diferentes componentes del cuerpo, como lo pueden ser el agua, los huesos o el mismo musculo, por otro lado la obesidad es el superar el límite de peso saludable pero en grasa, el segundo llega a ser el más complejo, ya que además del bajo bienestar de vida puede conllevar problemas de salud en el individuo que la padezca como lo son las enfermedades no transmisibles en las que se encuentran la diabetes, alto colesterol, problemas arteriales, entre otras las cuales se pueden presentar en un grupo de población que se encuentre en cualquier rango de edad, ya sean niños, adultos o adultos mayores y en mayor caso en sus dos extremos. Según la UNICEF en 2019 advierte que una gran cantidad de niños sufre las malas consecuencias de una mala alimentación [1]. Además, en el comunicado de prensa titulado El Estado Mundial de la Infancia 2019: Niños, alimentos y nutrición se dice que uno de cada tres niños menores de 5 años esta desnutrido o en sobrepeso [2].

El Índice de Masa Corporal (IMC), es un método matemático, donde mediante la relación del peso y la altura de una persona se dividen en categorías en las que las personas tienen una idea de cómo esta relación, puede afectar su salud.

$$IMC = \frac{PESO\ (Kg)}{ESTATURA^2\ (m)}$$

Ecuación 1. Índice de masa corporal

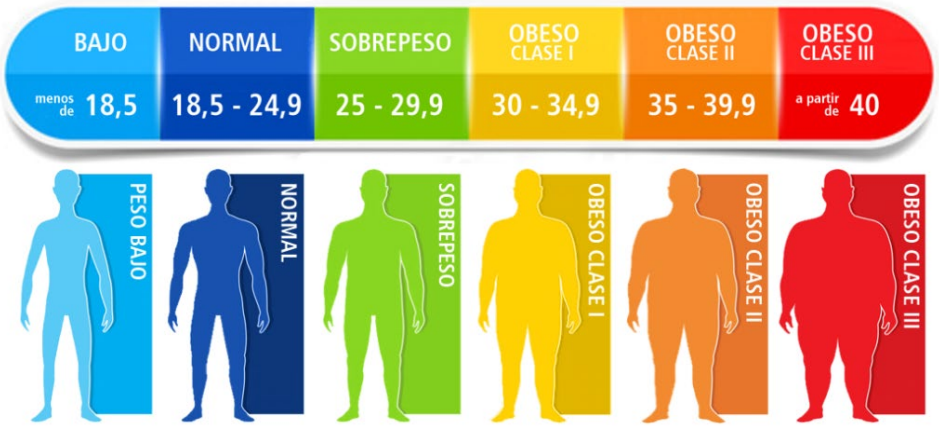


Figura 1 Relación de índice de masa corporal con respecto a la obesidad [6]

La tecnología móvil se ha vuelto cada vez más popular y accesible para las personas, gran parte de la población se encuentra conectada directamente a estas tecnologías que además de ayudar con la prevención, seguimiento y control de diversas enfermedades a personas del común sin importar su nivel de educación o preferencias, conectando a estas mismas personas con sus prestadores de servicios médicos, brinda información relevante sobre temas desconocidos de la salud, que a pesar de estar normalmente a disposición no está filtrada y organizada como lo permite estar la tecnología móvil.

Los aplicativos móviles son herramientas que pueden ayudar a las personas a mejorar su día a día facilitando actividades y tareas o tan solo recordando y dando información que es necesaria tener a la mano, según intereses, como lo pueden ser controlar la alimentación y el mantenimiento de una vida más saludable. Dentro de todas las categorías en las que se pueden dividir las aplicaciones, se tiene las que permiten llevar un registro de las calorías consumidas, las rutinas de ejercicio y los hábitos alimenticios.

Para la realización del proyecto se tendrán en cuenta los aplicativos móviles ya existentes, sus funcionalidades, sus limitaciones y alcances, para con esto realizar un aplicativo capaz de brindar soluciones y ventajas ya existentes, solventar las falencias que haya y mejorar la funcionalidad de este tipo de aplicaciones.

Las aplicaciones móviles para el control nutricional son cada vez más populares y hay una amplia variedad de opciones en el mercado. Algunas de las aplicaciones más populares incluyen:

- Contador de Calorías MyFitnessPal (Android, IOS): Permite realizar un seguimiento de las calorías que consumimos diariamente. Tiene una gran biblioteca de alimentos y la opción de agregar alguno nuevo. Lleva un registro de los principales nutrientes: grasas, colesterol, sodio, potasio, carbohidratos, proteínas, etc. Puede sincronizarse con distintos gadgets [4].
- Contador de Calorías Fat Secret (Android, IOS, web): Otra app que nos permite conocer la información nutricional de los alimentos que consumimos. Permite la búsqueda de alimentos a través de un código de barras y llevar un registro diario de las comidas. Dispone de una amplia base de datos y está orientada al país del usuario. Se caracteriza por su diseño ordenado y por presentar las cantidades totales de nutrientes al detalle [4].
- My Plate Calorie Tracker (Android): A través de esta app es posible realizar un seguimiento no solo de las comidas, sino también de la cantidad de agua que consumimos, algo importante para quienes olvidan beber los dos litros diarios que recomiendan los expertos. Su base de datos es constantemente actualizada, pues también permite introducir manualmente los datos de cada alimento. El usuario puede ingresar sus propias recetas y la aplicación calculará los valores nutricionales exactos de dicha receta [4].
- Lose It! (Android): Es una app ideal para quienes desean perder peso. Dispone de herramientas a través de las cuales podemos crear un plan personalizado con ejercicios, una dieta saludable, tiempo de trabajo, etc. Es necesario añadir cada alimento consumido para que la app calcule el número de calorías que se ha ingerido. A través de una fotografía del código de barras se obtiene el valor nutricional preciso [4].

Teniendo en cuentas las utilidades de estas aplicaciones se va a diseñar la interfaz de usuario que permita relacionar la cantidad de alimento ingerida con el producto específico para conocer el aporte nutricional de esa comida.

La información será obtenida de la tabla de composición de los alimentos colombianos (TCAC) de 2018 realizada por el instituto colombiano de bienestar familiar (ICBF) que teniendo en cuenta los lineamientos internacionales que

presenta 773 alimentos naturales, crudos, cocidos y procesados indicando el contenido de nutrientes y aporte de energía de los alimentos seleccionados.

1.2 Planteamiento del problema

La malnutrición en general definida según la RAE como una condición causada por una dieta inadecuada o insuficiente, o por un defecto en el metabolismo de alimentos se presenta de 3 formas diferentes: desnutrición, malnutrición relacionada con micronutrientes y sobrepeso y obesidad [5]. En todos los casos presentando consecuencias asociadas a la mala alimentación afectando el organismo como se enumeran a continuación:

- Enfermedades asociadas: Padecimiento de enfermedades crónicas como el cáncer, diabetes, osteoporosis, enfermedades dentales y obesidad.
- Deterioro de la memoria: Dando más posibilidades de sufrir de Alzheimer.
- Bajo rendimiento deportivo: Presentando un cambio en el desempeño usual.
- Problemas digestivos: La incorrecta asimilación de los nutrientes que se observa en desajustes intestinales y estomacales.
- Insomnio: Ocasionando dificultad para conciliar el sueño.

Con la finalidad de evitar estas dolencias que afectan a toda la población en general, sin discriminar edades, es necesario crear conciencia de la procedencia de las enfermedades acompañado de una estrategia que permita tener control de las causas de la malnutrición. Para lograr una alimentación saludable se propone una aplicación que pueda manejar la información pública de los alimentos que se producen en Colombia para luego presentar un resumen detallado del aporte nutricional de la ración ingerida.

Por este motivo se va a hacer uso de la base de datos presentada por el ICBF en su tabla de información nutricional dividida en 16 grupos de alimentos como se muestra a continuación:

- Cereales y derivados
- Verduras, hortalizas y derivados
- Frutas y derivados
- Grasas y aceites

- Pescados y mariscos
- Carnes y derivados
- Leche y derivados
- Bebidas (alcohólicas y no alcohólicas)
- Huevos y derivados
- Productos azucarados
- Misceláneos
- Alimentos para regímenes especiales
- Alimentos nativos
- Alimentos manufacturados
- Alimentos preparados
- Leguminosas y derivados

En los que cada uno muestra los alimentos correspondientes indicando análisis proximal, vitaminas, minerales, ácidos grasos y colesterol. Con la intención de manejar de manera eficiente la información se va a diseñar la aplicación cumpliendo los siguientes estándares:

1. Seguridad: ya que la información que ingresan los usuarios es personal y reflejan los hábitos alimenticios de la población.
2. Compatibilidad: permitiendo que la aplicación este en diferentes dispositivos móviles y sistemas operativos, adaptándose a las diferentes resoluciones de pantallas.
3. Diseño intuitivo: brindando al usuario una aplicación con diseño atractivo y que no presente vacilaciones al momento de usarla.
4. Rendimiento: realizando consultas en el menor tiempo posible para que la información sea presentada en pantalla momento de ser solicitada.

1.3 Justificación

Según la Organización Mundial de la Salud (OMS), la obesidad en niños y adolescentes aumento casi cinco veces durante el último decalustro. Según la directora general en la 47ª reunión de la Academia Nacional de Medicina, en tan solo unas décadas el mundo ha pasado de un perfil nutricional en el que la

prevalencia de la insuficiencia ponderal superaba en más del doble a la de la obesidad, a la situación actual, en la que hay en el mundo más personas obesas que personas con insuficiencia ponderal [8]. Por esta razón ha surgido una creciente preocupación por la salud y el bienestar de las personas a nivel global que ha conllevado a recurrir a medidas de prevención y tratamiento de la obesidad por medio de campañas y a su vez por medio de la tecnología.

La alimentación es un factor fundamental para mantener un estilo de vida saludable, y cada vez más personas se interesan por llevar una dieta equilibrada y sana. Sin embargo, al empezar con el proceso de mejoramiento en la alimentación, se puede llegar a cometer muchos errores ya que existe abundante información y ayuda profesional sobre cómo llegar a cierto objetivo, pero no sobre los alimentos como tal, información relevante que ayudarían a llevar un control preciso de cada alimento; como lo es su valor nutricional, porcentaje de grasa, entre otros, que facilitarían el proceso de adaptación a un estilo de vida saludable a personas del común, especialmente a personas que tienen un estilo de vida ajetreado o no disponen de la información necesaria. Un aplicativo móvil para el control nutricional de las comidas puede ser una herramienta muy útil para ayudar a las personas a llevar un registro de lo que comen, conocer el valor nutricional de los alimentos y planificar sus comidas de manera más saludable.

Además, un aplicativo móvil puede ser especialmente beneficioso no solo para personas interesadas en mejorar sus hábitos alimenticios, sino también para aquellos que deben cambiar su estilo de vida por padecimiento de enfermedades no transmisibles, crónicas o trastornos alimentarios, ya que les permite llevar un control más preciso de su dieta y adaptarla a sus necesidades específicas.

En resumen, un aplicativo móvil para el control nutricional de los alimentos que pueden contribuir significativamente a mejorar la salud y el bienestar de las personas, facilitándoles la tarea de llevar una dieta equilibrada y adaptada a sus necesidades individuales.

1.4 Objetivos

1.4.1 **Objetivo general**

- Diseñar un aplicativo móvil que permita llevar el conteo del consumo de nutrientes y calorías durante el día para tener control de las cantidades requeridas en una dieta saludable.

1.4.2 **Objetivos específicos**

- Diseñar un aplicativo móvil que permita seleccionar los alimentos y según la porción que se ingrese calcule la información nutricional.
- Diseñar un aplicativo cliente servidor del aplicativo de software.
- Validar el funcionamiento de la aplicación de software.

1.5 Metodología

Para llevar a cabo el proyecto se empleará la metodología de la siguiente manera:

1.5.1 **Fase 1: Construcción Base de datos de lectura.**

En esta etapa mediante la información brindada por el Instituto Colombiano de Bienestar Familiar (ICBF) en su tabla de composición de alimentos colombianos del 2018 (TCAC) se realizará una base de datos con los 773 alimentos que tiene con su información de contenido de ácidos grasos y aminoácidos tomando solo los necesarios para guardar información relevante.

1.5.2 **Fase 2: Construcción Base de datos de escritura**

Siguiendo un esquema para almacenar la información de manera organizada y utilizando Firebase como plataforma para guardar la información de los usuarios se identificarán los ítems necesarios, se crearán protocolos de seguridad al momento de realizar consultas y restricciones para garantizar la integridad de los datos.

1.5.3 **Fase 3: Diseño aplicativo móvil**

Con ayuda del entorno de desarrollo Android Studio se va a diseñar una interfaz amigable con el usuario que de manera intuitiva permita seleccionar el alimento y

su cantidad para que de esta manera la aplicación pueda calcular el aporte nutricional de cada una de las comidas.

1.5.4 **Fase 4: Realización de pruebas**

La aplicación desarrollada de someterá a una serie de pruebas que permitan corroborar su veracidad en la información que brinda, buscando corresponder con otras que hay en la tienda de aplicaciones.

1.5.5 **Fase 5: Elaboración documento del proyecto**

Diligenciar el documento final del proyecto en donde se incluirán aspectos técnicos durante el desarrollo del aplicativo y a su vez lo resultados de las pruebas.

2. MARCO TEORICO

2.1 Android

Android es un sistema operativo móvil desarrollado por Google ampliamente utilizado en teléfonos inteligentes, tablets, relojes inteligentes, televisores y automóviles. Este sistema operativo es conocido por su flexibilidad y personalización, lo que permite a los fabricantes y desarrolladores de dispositivos crear experiencias únicas.

Las características clave de Android incluyen un ecosistema de aplicaciones grande y diverso a través de Google Play Store, opciones de personalización para adaptar el sistema a sus preferencias personales, seguridad y privacidad, e integración con servicios de Google como Google Assistant y Google Maps.

2.2 Firebase, realtime database y autenticación

Firebase es una plataforma en la nube para el desarrollo de aplicaciones web y móviles disponible en todas las plataformas. Firebase tiene productos y soluciones para todas las etapas de desarrollo de la aplicación empezando por la compilación, lanzamiento y supervisión y el análisis de las estadísticas de interacción del usuario. Firebase ofrece una función de base de datos en tiempo real que almacena datos en la nube en formato JSON, lo que proporciona la ventaja de ser una base de datos NoSQL. Esta característica permite almacenar y acceder a datos en tiempo real, garantizando que la información esté siempre actualizada y disponible.

Además, firebase proporciona un sistema de autenticación que permite a los usuarios registrarse con correo electrónico y contraseña o utilizar perfiles de otras plataformas como Facebook o Google. Esto simplifica el acceso a las aplicaciones y garantiza la seguridad de los datos al almacenar de forma segura la información de inicio de sesión en la nube, evitando que los usuarios tengan que identificarse repetidamente al abrir la aplicación.

2.3 Java

JAVA es un lenguaje de programación creado por Sun Microsystems en 1995. Este lenguaje se caracteriza por ser multiplataforma, orientado a objetos y centrado en la red que se puede utilizar como una plataforma lo que lo hace un lenguaje de programación rapido, confiable y seguro para codificar aplicaciones móviles y software empresarial.

Dentro de sus usos principales estas el desarrollo de videojuegos, computación en la nube, manejo de conjuntos de datos complejos, inteligencia artificial e internet de las cosas (IoT).

2.4 GitHub

GitHub es una plataforma integral que ofrece servicios en la nube esenciales para los desarrolladores. Permite almacenar, gestionar y colaborar en proyectos de código, facilitando un control preciso de versiones y un seguimiento detallado del proceso de desarrollo. GitHub es utilizado para almacenar su código en repositorios, trabajar en diferentes ramas para nuevas características o correcciones y fusionar esos cambios de manera efectiva.

3. HERRAMIENTAS

El desarrollo de la aplicación se realiza en el entorno de desarrollo Android Studio usando como lenguaje de programación Java. Para el almacenamiento de la información se hace uso de la plataforma de desarrollo Firebase que mediante la base de datos en tiempo real permite almacenar, sincronizar y buscar datos en tiempo real manteniendo una estructura NoSQL. Además, para las consultas de la información nutricional de los alimentos colombianos se utiliza la aplicación de Google Sheets con el cual, mediante un script se implementa una API que se encarga de entregar toda la información en formato JSON para su posterior análisis en la aplicación.

La aplicación permite que cada uno de los usuarios, nuevos o registrados puedan agregar información acerca de la cantidad y el tipo de alimentos ingeridos en cada una de las 3 principales comidas diarias y luego ver reflejado el aporte nutricional en general e individual de las comidas mostrando una advertencia según los valores estándares de carbohidratos, colesterol, energía, grasas saturadas, lípidos, proteínas y sodio.

3.1 Bases de datos

Para este proyecto se hace necesario el uso de una base de datos NoSQL en tiempo real brindada por firebase que está enfocada en el desarrollo de aplicaciones móviles y la herramienta de Google Sheets que permite almacenar información.

3.1.1 Firebase realtime database

En esta base de datos en tiempo real NoSQL almacena información en formato JSON creando una estructura de diagrama de árbol trabajando con claves y valores. Para el desarrollo de la aplicación se realiza la siguiente estructura:

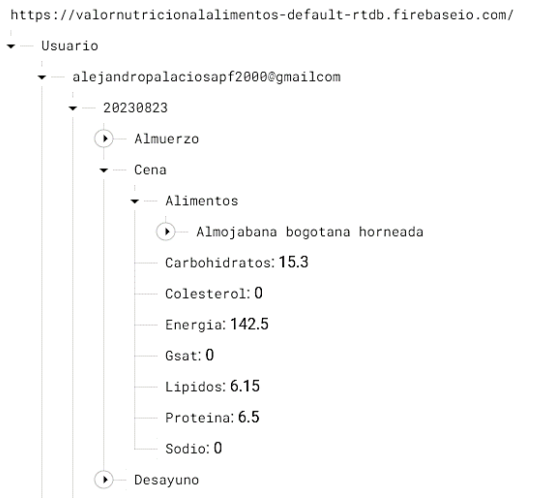


Figura 2 Estructura base de datos firebase

Dentro de la clave principal “Usuarios” se almacena la información de todos los usuarios registrados que se van a identificar por el correo electrónico que ingresan al registrarse cada uno de los usuarios. Además, al momento de registrarse o iniciar sesión un día diferente a los que están almacenados se crea un nuevo nodo con la fecha del día de ingreso en el siguiente formato: año, mes, día. Dentro de este nodo se crean 3 elementos con cada una de las 3 comidas principales “Desayuno”, “Almuerzo” y “Cena”. Estos nodos guardan la misma estructura compuesta por 7 valores de índices nutricionales tales como carbohidratos, colesterol, energía, grasas saturadas, lípidos, proteínas y sodio, además del elemento “Alimentos” que almacena a su vez la misma información anteriormente dicha, pero con los valores específicos de los alimentos seleccionados en el proceso de agregar alimentos.

3.1.2 **API Google Sheets**

Mediante el uso de la herramienta de Google Sheets y su entorno de programación mediante scripts se crea una API que permita acceder a la información que esta almacenada en la hoja de cálculo que contiene los valores nutricionales de los 773 alimentos que están registrados en la base de datos del Instituto Colombiano de Bienestar Familiar ICBF en su Tabla de Composición de Alimentos Colombianos (TCAC).

Para este proceso se tiene en la hoja de cálculo llamada nutrición la información separada en 8 columnas identificadas como Nombre, Energía, Proteína,

Carbohidrato, Colesterol, Lípidos, Gsat y Sodio como se muestra en la figura a continuación:

	A	B	C	D	E	F	G	H
1	Nombre	Energia	Proteina	Carbohidratos	Colesterol	Lipidos	Gsat	Sodio
2	Almidon de maiz crudo	370	0.3	91.4	0	0.1	0	9
3	Almojabana, bogotana, horneada	285	13	30.6	0	12.3	0	0
4	Almojabana, vallecaucana, horneada	341	17.5	34.6	0	14	0	0
5	Arepa de maiz precocido con sal	163	3.3	35	0	0.9	0	188
6	Arepa de maiz precocido sin sal	156	3.4	34.5	0	0.5	0	0
7	Arepa de maiz asada	162	4.1	36.3	0	0	0	0
8	Arepa de maíz, con queso asada	211	4.8	29	0	8.4	0	0
9	Arepa de maíz. Frita	325	3.4	32.1	0	20.3	0	0
10	Arroz blanco cocido sin sal	11	2.3	32.5	0	2.1	0.7	4
11	Arroz blanco crudo	353	6.7	80.1	0	0.4	0.2	2
12	Arroz integral, cocido, sin sal	30	7.5	77.3	0	2.6	0.5	4
13	Avena e n hojuelas. pre cocida	411	16.9	64.1	0	7.5	1.3	3
14	Brownie de chocolate horneado	427	4.8	64.2	17	16.3	4.2	286
15	Cebada, perlada, cruda	385	12.5	72.1	0	1.8	0.3	7
16	Cereal, hojueJas de maíz, sin azúcar	383	8.1	85	0	0.4	0.2	680
17	Croissam de queso horneado	445	9.2	48.9	53	23	10.2	423
18	Cuchuco de cebada crudo	358	8.4	79.4	0	0.7	0	0
19	Cuchuco de trigo crudo	358	12.8	74	0	1.2	0	0
20	Envuelto de mazorca, precocido	186	4.8	23.5	0	7.9	0.5	132
21	Galletas dulces, ccn relleno	516	3.8	70.1	0	24.1	12.3	103
22	Galletas dulces cucas	390	7.6	83.2	0	3	0	0
23	Ganetas dulces, de avena con uvas pasas	451	6.5	68.6	33	16.2	3.2	538
24	Galletas dulces, sin relleno	472	6.4	67.1	12	19.6	7.7	306

Figura 3 Base de datos de Google Sheets

Teniendo en cuenta la estructura de la hoja de cálculo de hace el script en Google Apps Script en dos funciones:

- Función getData()

  En esta función se obtienen los valores de la hoja de cálculo analizando cada una de las columnas fila por fila retornando un JSON que tiene todos los valores de la tabla.

- Función doGet()

  En esta función se hace una solicitud GET que si contiene los parámetros del ID de implementación y el nombre de la hoja de cálculo retorna la información en formato JSON obtenida de la función getData().

Cuando se hace una consulta a la API con la información correcta en sus métodos se obtiene el archivo como se muestra a continuación:

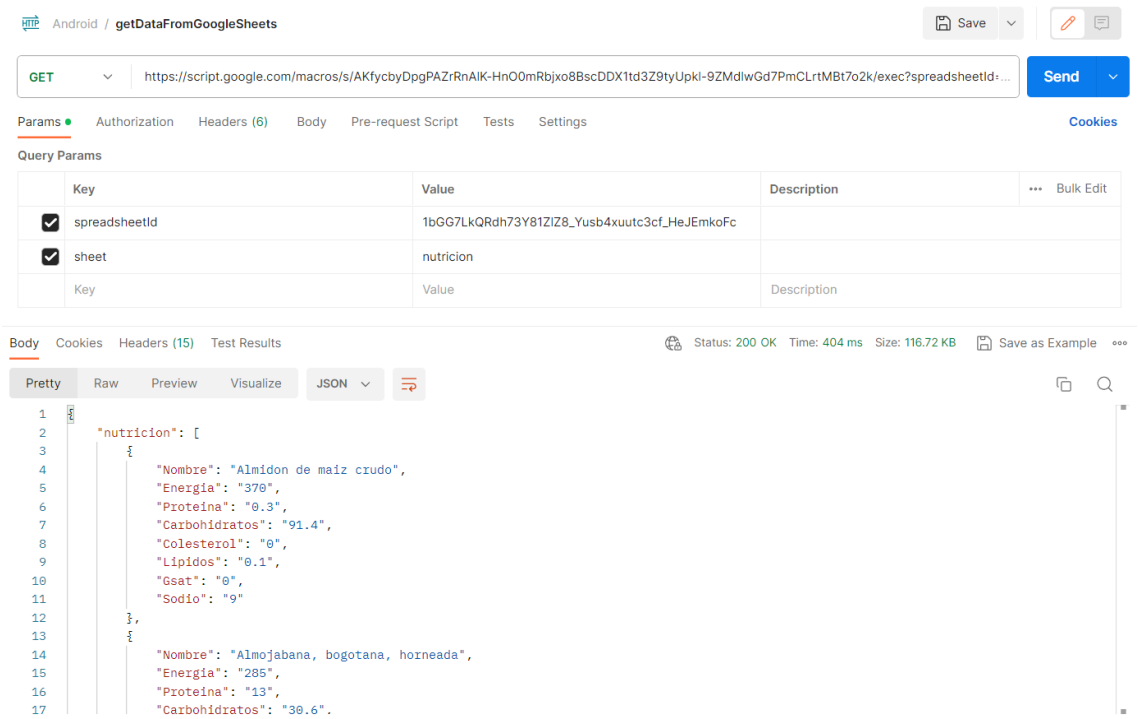


Figura 4 Consulta postman a API

En esta consulta ejecutada en Postman se especifica el tipo de requerimiento que se va a realizar en la URL y se le ingresan los valores de ID de implementación y nombre de la hoja de cálculo lo que permite que se obtenga un archivo JSON con los 773 alimentos registrados en el TCAC.

3.2 Interfaz grafica

Usando el entorno de desarrollo de Android Studio y su facilidad para diseñar aplicaciones móviles mediante bloques y codigo en XML que permite ver en tiempo real como se refleja la aplicación en el dispositivo se diseñan cada una de las vistas teniendo en cuenta las necesidades y los objetivos a cumplir creando una interfaz amigable e intuitiva para el usuario.

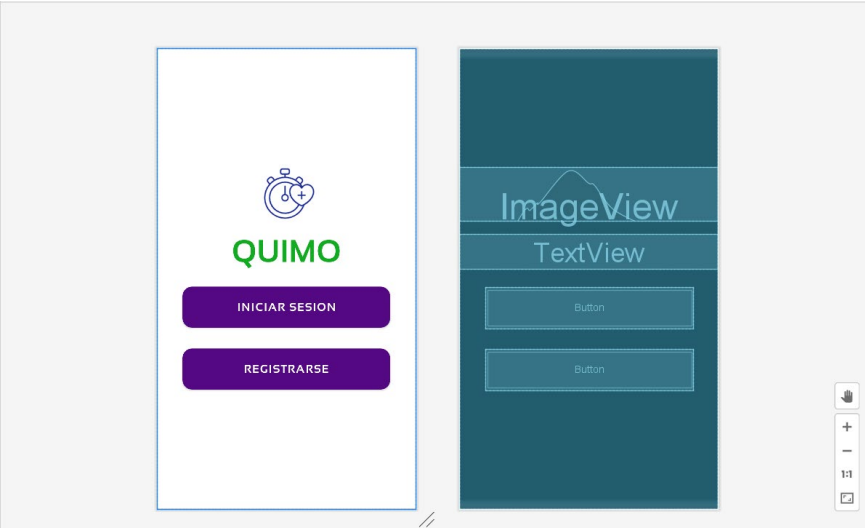


Figura 5 Interfaz gráfica Android Studio

3.3 GitHub

Utilizando la función de controlador de versiones de GitHub se logra tener todo el codigo del aplicativo móvil en la nube para ser revisado y mejorado por algún otro desarrollador. Esta herramienta permitió trabajar en diferentes elementos de la app de forma sincrónica para que este proceso se manejara de la manera mas efectiva posible. Al final todos los cambios se establecieron en la rama principal.

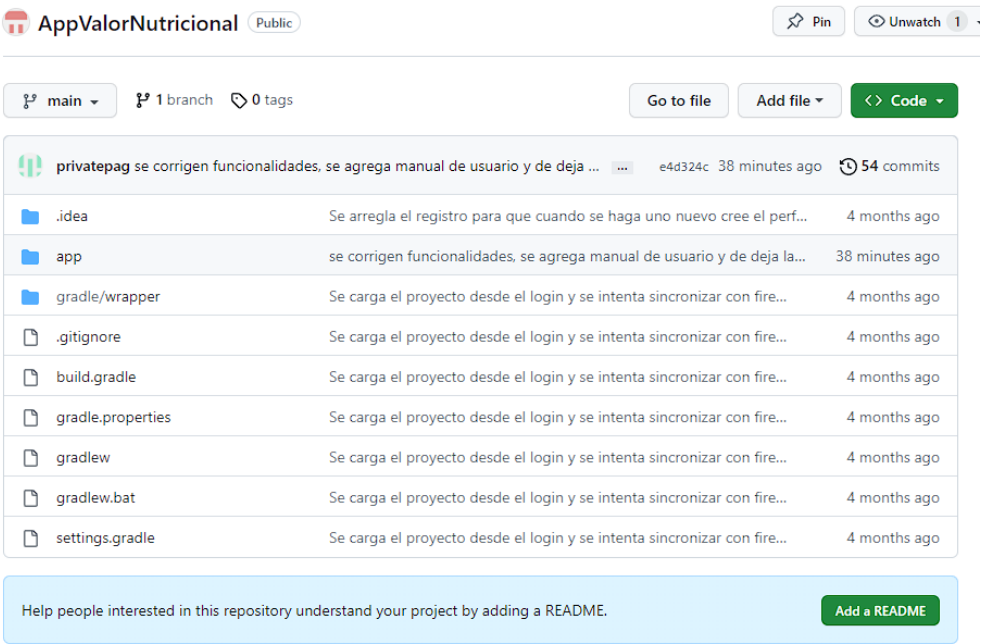


Figura 6 Repositorio en GitHub

4. DESARROLLO DE APLICATIVO MOVIL

4.1 Configuración

4.1.1 **AndroidManifest.xml**

En este componente se realizan las configuraciones principales del proyecto, asi como los permisos que tiene la aplicación, se declaran las actividades (Main, Registrar, Iniciar sesión, Información, Agregar información y Agregar Alimento), se le otorga permiso para la conexión a internet y se cambia el nombre y logo de la aplicación.

4.1.2 **Build.gradle**

Se ajustan las librerías necesarias para la implementación del proyecto. En este caso solo se va a agregar para el manejo de firebase. La aplicación se desarrolla en la versión 33 del SDK, pero al momento de generar el apk la versión de SDK mínimo pasa a ser la 24 y se marca como objetivo teléfonos que manejen el SDK 29. Esto va a permitir que dispositivos que tengan desde Android 7.0 para adelante puedan instalar la aplicación sin problemas de compatibilidad significando una cobertura al 95.4% de dispositivos disponibles.

4.2 Vistas

La aplicación está estructurada en 6 vistas en actividades por separado, cada una diseñada para cumplir una función en específico y dependiendo de las necesidades se utilizan herramientas extra para cumplir su función.

En los anexos se encuentra el diagrama de flujo del aplicativo móvil diseñado.

4.2.1 **Bienvenida**

En esta vista se tiene 4 componentes conformados por una imagen del logo, el nombre de la app “QUIMO” y dos botones que van a redirigir a la vista correspondiente con su nombre según la necesidad de cada usuario, si es un usuario registrado o es un usuario nuevo que no ha tenido una cuenta en la aplicación. La vista diseñada se presenta a continuación:

Figura 7 Vista Bienvenida

El funcionamiento de esta vista está representado por el siguiente diagrama de flujo:

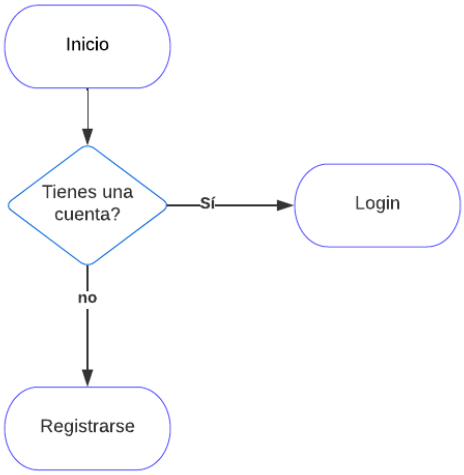


Figura 8 Diagrama de flujo Bienvenida

La estructura para implementar la interacción se basa en iniciar una actividad cuando el botón sea presionado. Cuando la interacción de cambiar de una vista a otra sea completada esta vista se cierra para evitar el consumo de recursos en segundo plano.

```
BtnRegistrarse.setOnClickListener(new View.OnClickListener() {
    AlejoPal
    @Override
    public void onClick(View view) {
        Intent intent = new Intent( packageContext: MainActivity.this, Registrar.class);
        startActivity(intent);
        finish();
    }
});
```

Figura 9 Estructura actividad botones

4.2.2 **Iniciar sesión**

Compuesta por dos campos para ingresar texto, el primero con formato de correo para ingresar el correo con el que fue registrado el usuario y el segundo para colocar la contraseña numérica previamente establecida. Además, de un botón que será el encargado de realizar la consulta a la base de datos para verificar si el correo y la clave corresponden con alguno de los registrados.

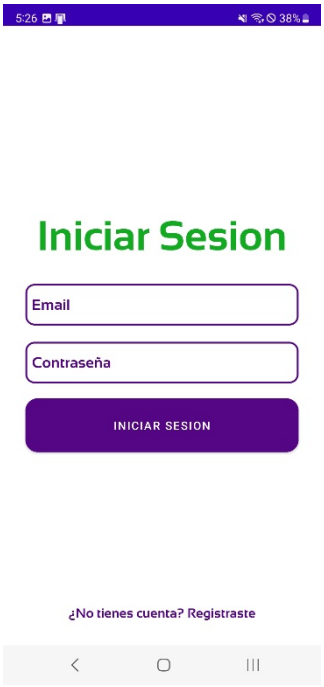


Figura 10 Vista Iniciar sesión

En la figura 11 se muestra el diagrama de flujo de esta vista.

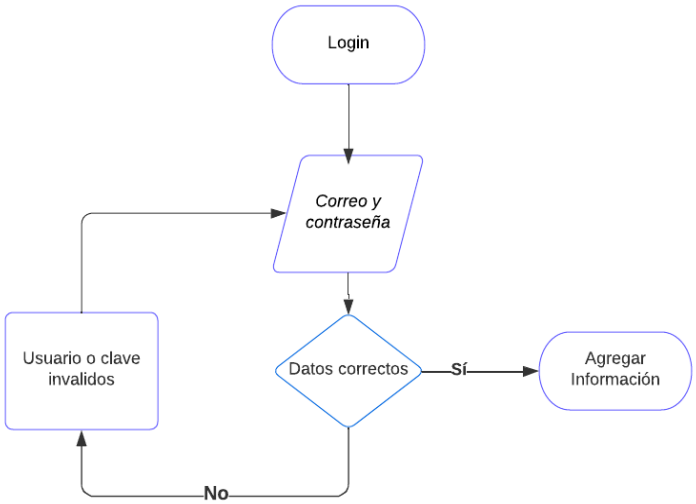


Figura 11 Diagrama de flujo login

Para la autenticación con Firebase se hacen uso de dos valores que se obtienen de los campos email y contraseña, estos se ingresan a la consulta y según el resultado de este proceso se decide si realizar el proceso de iniciar sesión y pasar a la siguiente vista o mostrar un mensaje de error diciendo “Usuario y contraseña invalidas”.

```
mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
    1 usage
    Date date = new Date();
    AlejoPal *
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        if (task.isSuccessful()){
            FirebaseDatabase database = FirebaseDatabase.getInstance();
            DatabaseReference mRootReference = database.getReference();
            DatabaseReference usuarioRef = mRootReference.child( pathString: "Usuario").child(email.replace( target: ".",  replacement: ""));
            SimpleDateFormat fechaC = new SimpleDateFormat( pattern: "yyyyMMdd");
            String sfecha = fechaC.format(date);
            cargarDatosFirebaseI(sfecha, correo.getText().toString().replace( target: ".",  replacement: ""));
            Toast.makeText( context: InicarSesion.this,  text: "Login", Toast.LENGTH_SHORT).show();
            AlejoPal
            new Handler().postDelayed(new Runnable() {
                AlejoPal
                @Override
                public void run() {
                    FirebaseUser user = mAuth.getCurrentUser();
                    Intent intent = new Intent( packageContext: InicarSesion.this, AgregarInfo.class);
                    intent.putExtra( name: "Correo", correo.getText().toString().replace( target: ".",  replacement: ""));
                    startActivity(intent);
                    finish();
                }
            }, delayMillis: 200);

        }else {

            Toast.makeText( context: InicarSesion.this,  text: "Usuario o Contraseña invalidas", Toast.LENGTH_SHORT).show();

        }
    }
});
```

Figura 12 Autenticación login firebase

A continuación, se muestra el resultado de poner un usuario y contraseña que no han sido registrados o cuando la contraseña no corresponde con la información del correo ingresado.

Figura 13 Vista iniciar sesión con advertencia

Cuando los datos ingresados son correctos y la respuesta a la consulta es satisfactoria se empieza un nuevo proceso para cambiar de vista y redirigirlo a la vista de agregar información. Al momento de realizar este proceso, la aplicación revisa en la base de datos en tiempo real si el día ingresado ya tiene valores para mostrar, si la respuesta es afirmativa solo llevara a la siguiente vista la variable del correo electrónico usado para ingresar y realizar las consultas en la base de datos en tiempo real. Por el contrario, si la consulta en negativa se procese a crear un nuevo nodo dentro de la ubicación en la base de datos correspondiente a ese usuario con la fecha del día que se está entrando, dejando los valores de energía, proteína, carbohidrato, colesterol, lípidos, gsat y sodio en 0 para cada una de las comidas diarias. Para este proceso se hace el uso de un mapa que va a contener la información de las 3 comidas diarias, cada una con un mapa de los valores nutricionales como se muestra a continuación:

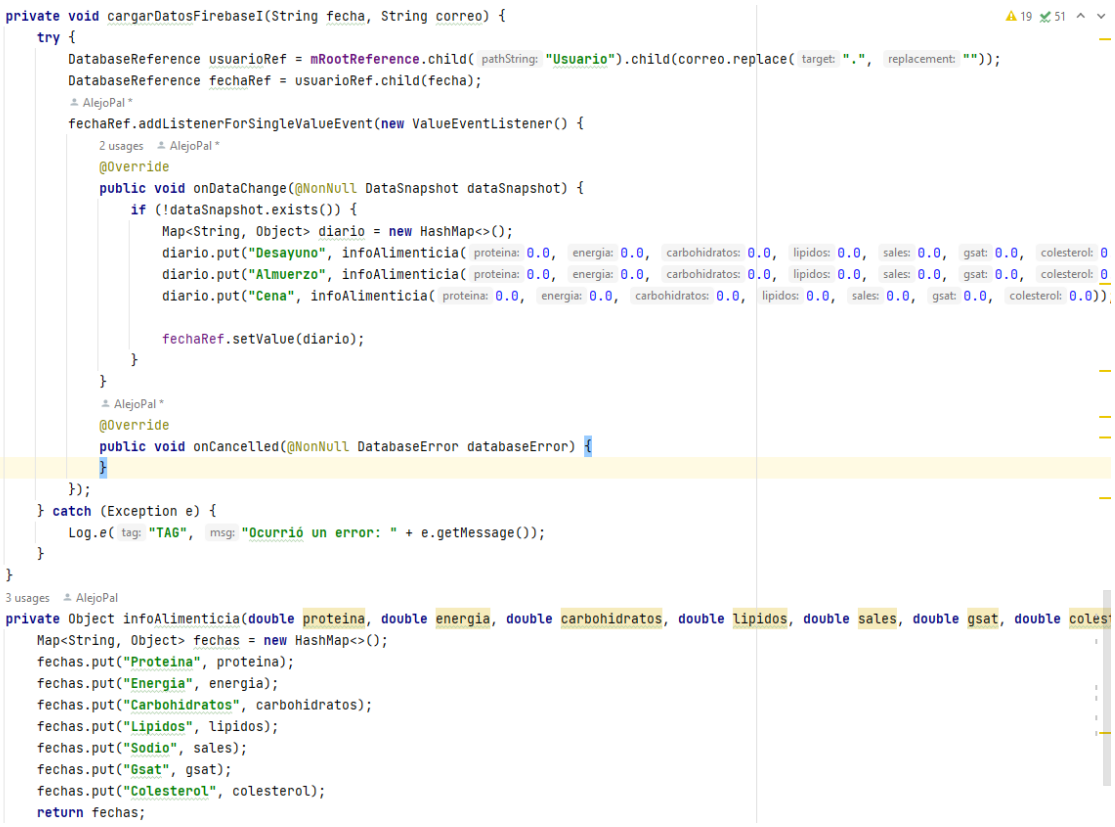


Figura 14 Crear datos en firebase un nuevo día

El proceso se realiza con la anterior estructura para que al momento de iniciar sesión solo sea necesario establecer conexión con la base de datos una vez teniendo una variable con toda la información requerida a actualizar en la nube y no consumir demasiados recursos al realizar un envío masivo de datos.

Por último, al final de la pantalla se puede observar el mensaje: “¿No tienes cuenta? Regístrate”, el cual es un atajo a la vista de registrarse en caso de que el usuario presione la opción incorrecta al momento de iniciar en su aplicación.

4.2.3 **Registrar**

Esta vista está compuesta por 4 campos de textos. El primero es un texto plano que va a contener el nombre del usuario que se está registrando, luego un campo para ingresar el correo electrónico personal y por último dos campos de contraseña numérica de mínimo 6 caracteres y siguiendo la misma estructura al final de la pantalla se muestra un mensaje que dice “¿Ya tienes cuenta? Inicia sesión” que enviara al usuario a la vista de iniciar sesión en caso de ya estar registrado.

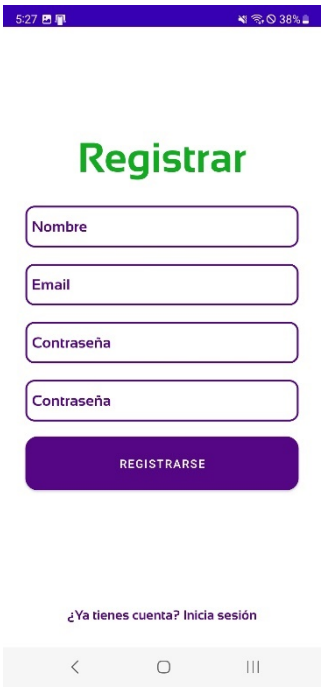


Figura 15 Vista Registrarse

Su funcionamiento queda reflejado en el siguiente diagrama de flujo:

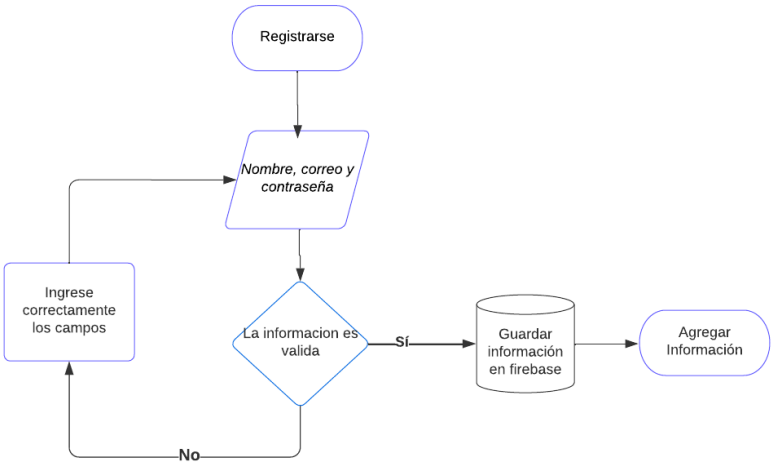


Figura 16 Diagrama de flujo Agregar información

Como medidas de precaución en esta vista se tienen mensajes de advertencia cuando el usuario presiona el botón de registrarse y no ha llenado los campos mostrando el mensaje “Por favor complete todos los campos”. Además, cuando las contraseñas que ingresan son demasiado cortas muestra el mensaje de “La

contraseña debe tener al menos 6 caracteres” y en caso de que las contraseñas ingresadas sean diferentes, muestra el mensaje “Las contraseñas no coinciden”.

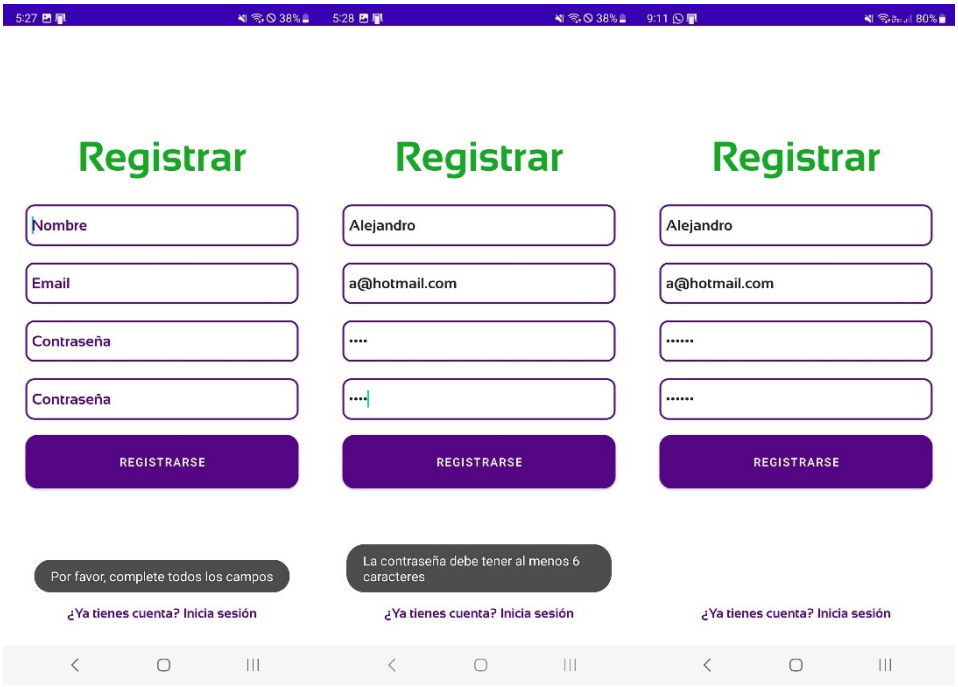
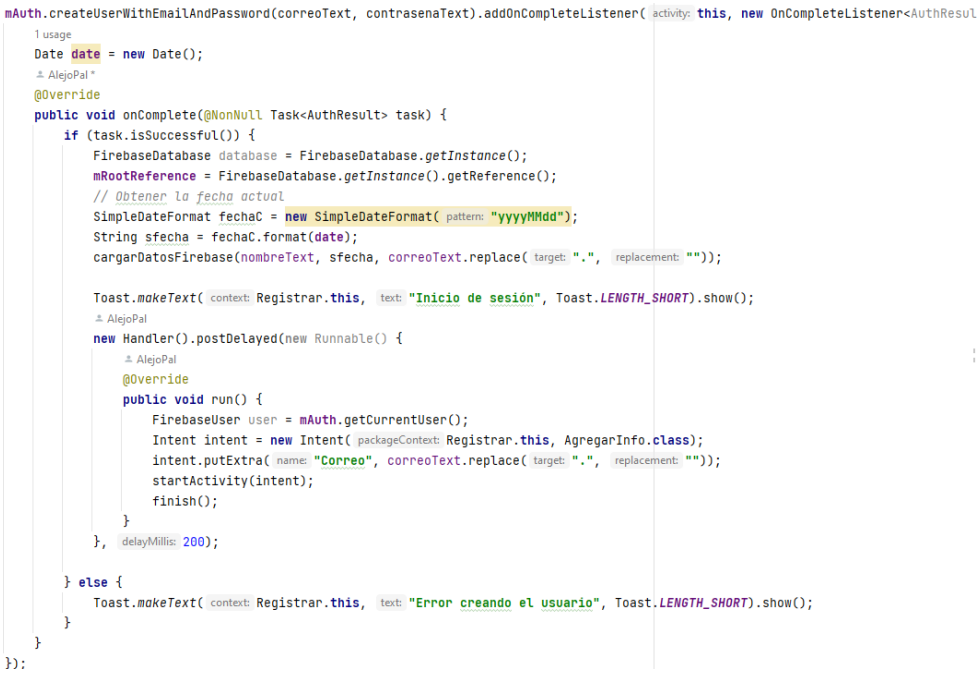


Figura 17 Vista registrarse con advertencia

Cuando la información esta correctamente diligenciada mediante el método “createUserWithEmailAndPassword” se crea en firebase la información con un nuevo usuario en su herramienta de autenticación.



```
mAuth.createUserWithEmailAndPassword(correoText, contrasenaText).addOnCompleteListener( activity: this, new OnCompleteListener<AuthResul
    1 usage
    Date date = new Date();
    AlejoPal *
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        if (task.isSuccessful()) {
            FirebaseDatabase database = FirebaseDatabase.getInstance();
            mRootReference = FirebaseDatabase.getInstance().getReference();
            // Obtener la fecha actual
            SimpleDateFormat fechaC = new SimpleDateFormat( pattern: "yyyyMMdd");
            String sfecha = fechaC.format(date);
            cargarDatosFirebase(nombreText, sfecha, correoText.replace( target: ".",  replacement: ""));

            Toast.makeText( context: Registrar.this,  text: "Inicio de sesión", Toast.LENGTH_SHORT).show();
            AlejoPal
            new Handler().postDelayed(new Runnable() {
                AlejoPal
                @Override
                public void run() {
                    FirebaseUser user = mAuth.getCurrentUser();
                    Intent intent = new Intent( packageContext: Registrar.this, AgregarInfo.class);
                    intent.putExtra( name: "Correo", correoText.replace( target: ".",  replacement: ""));
                    startActivity(intent);
                    finish();
                }
            }, delayMillis: 200);

        } else {
            Toast.makeText( context: Registrar.this,  text: "Error creando el usuario", Toast.LENGTH_SHORT).show();
        }
    }
});
```

Figura 18 Crear usuario en firebase

Cuando esta tarea se complete satisfactoriamente se crea un nuevo nodo en la base de datos en tiempo real en la rama usuarios conteniendo como valor un mapa con clave principal el correo electrónico, al que se le eliminan los puntos para que no tenga problemas al ser almacenado en firebase que a su vez crea un mapa con la información de los valores de energía, proteína, carbohidrato, colesterol, lípidos, gsat y sodio en 0 para cada una de las comidas diarias de la misma manera que se hizo al iniciar sesión.

Además, a los mapas de las 3 comidas diarias se agrega una entrada de datos que almacena el nombre del usuario registrado para tenerlo en cuenta en la siguiente vista.

```java
private void cargarDatosFirebase(String nombre, String fecha, String Correo) {
    try {
        Map<String, Object> Diario = new HashMap<>();
        Diario.put("Desayuno", infoAlimenticia( proteina: 0.0, energia: 0.0, carbohidratos: 0.0, lipidos: 0.0, sales: 0.0, gsat: 0.0, colesterol: 0.0));
        Diario.put("Almuerzo", infoAlimenticia( proteina: 0.0, energia: 0.0, carbohidratos: 0.0, lipidos: 0.0, sales: 0.0, gsat: 0.0, colesterol: 0.0));
        Diario.put("Cena", infoAlimenticia( proteina: 0.0, energia: 0.0, carbohidratos: 0.0, lipidos: 0.0, sales: 0.0, gsat: 0.0, colesterol: 0.0));

        Map<String, Object> datosUsuario = new HashMap<>();
        datosUsuario.put("Nombre", nombre);
        datosUsuario.put(fecha, Diario );
        if (mRootReference != null) {
            // Verificar que la variable "Correo" contenga un valor válido antes de utilizarla como clave
            if (Correo != null && !Correo.isEmpty()) {
                mRootReference.child( pathString: "Usuario").child(Correo).setValue(datosUsuario);
            } else {
                Toast.makeText( context: Registrar.this,  text: "Sin correo", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText( context: Registrar.this,  text: "Sin direccion", Toast.LENGTH_SHORT).show();
        }
    } catch (Exception e) {
        Log.e( tag: "TAG",  msg: "Ocurrió un error: " + e.getMessage());

    }
}
3 usages  AlejoPal
private Object infoAlimenticia(double proteina, double energia, double carbohidratos, double lipidos, double sales, double gsat, double cole
    Map<String, Object> fechas = new HashMap<>();
    fechas.put("Proteina", proteina);
    fechas.put("Energia", energia);
    fechas.put("Carbohidratos", carbohidratos);
    fechas.put("Lipidos", lipidos);
    fechas.put("Sodio", sales);
    fechas.put("Gsat", gsat);
    fechas.put("Colesterol", colesterol);
    return fechas;
}
```

Figura 19 Crear nodo de información para nuevo usuario en Firebase

En caso de que la creación del usuario falle va a mostrar el mensaje “Error creando el usuario”.

Cuando los datos son creados correctamente se empieza un nuevo proceso para cambiar de vista y redirigirlo a la vista de agregar información teniendo en cuenta que este proceso empezará 200ms despues de presionar el botón para que el tiempo de carga del nuevo nodo en firebase no genere un error al no tener información para empezar en la nueva vista.

4.2.4 **Agregar información**

Esta vista sirve de intermediaria entre la vista de Información General y Agregar Alimento. Los 3 primeros botones se encargan de empezar una actividad que lleven a la vista de Agregar alimento a cada una de las tres comidas diarias dependiendo del botón que presionen. Por otro lado, si lo que desea el usuario es revisar el historial de los alimentos que ha agregado previamente presiona el botón de “HISTORIAL” que lleva a la vista de Información General.

Figura 20 Vista Agregar información

La operación de esta vista se ilustra mediante el siguiente diagrama de flujo:

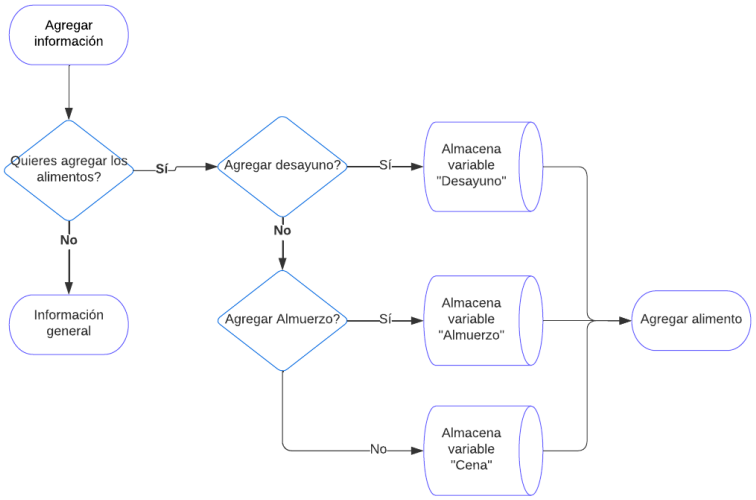


Figura 21 Diagrama de flujo Agregar información

Adicionalmente, al final de la pantalla se tiene un botón que redirecciona al manual de uso de la aplicación en formato pdf almacenado en la nube para que pueda resolver cualquier duda sobre el funcionamiento de la app.

Figura 22 Manual de usuario

4.2.5 **Agregar alimento**

Esta vista está compuesta por dos campos de texto y 3 botones. El primer campo de texto se usa con el elemento AutoCompleteTextView que va a permitir cargar la información de los nombres de los alimentos obtenidos del TCAC en Google Sheets.

Figura 23 Vista agregar alimento

El proceso a seguir se representa en la figura 24.

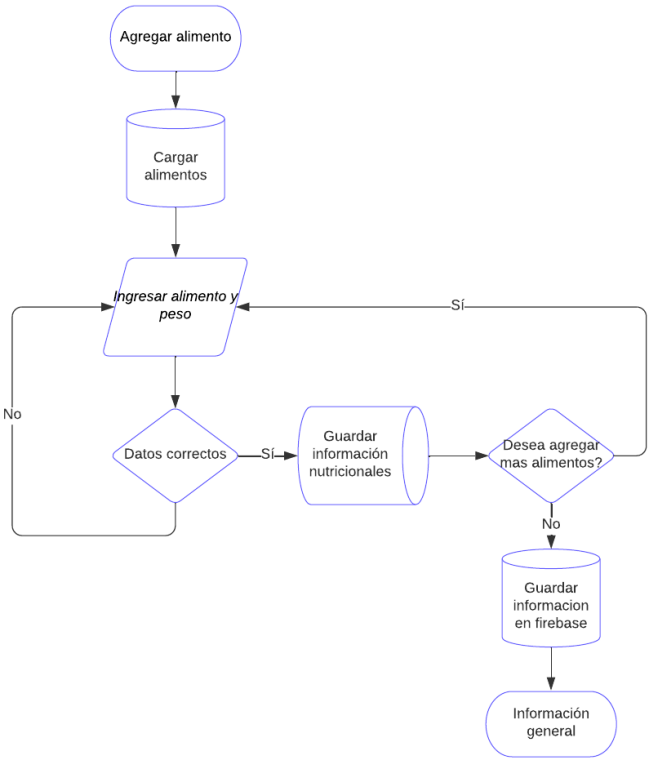


Figura 24 Diagrama de flujo Agregar alimento

Cuando el usuario ingrese una letra en ese campo de texto este mostrará una sugerencia con todas las coincidencias que tenga del arreglo que se ha ingresado.

Figura 25 Vista buscando alimento

Para vincular la lista con los nombres al AutoCompleteTextView se realiza una consulta a la API previamente creada que entrega el archivo en formato JSON al realizarla.

Esta búsqueda se hace de manera asíncrona cuando se llama la vista para que al momento de presionar el campo pueda buscar y sugerir respuestas lo más pronto posible.

```
@SuppressLint("StaticFieldLeak")
private void obtenerDatosColumnaDesdeSheets() {

    Button agregarButton = findViewById(R.id.btnAgregar);
    Button regresarButton = findViewById(R.id.btnRegresar);

    AlejoPal
    new AsyncTask<Void, Void, List<Alimento>>() {
        AlejoPal
        @Override
        protected List<Alimento> doInBackground(Void... voids) {
            // URL de la API con el spreadsheetId y sheet
            String apiUrl = "https://script.google.com/macros/s/AKfycbyDpgPAZrRnAlK-HnO0mRbjxo8BscDDX1td3Z9tyUpkl-9ZMdlwGd7PmCLrtMBt7o2k/exe

            try {
                // Realizar solicitud HTTP GET a la API
                URL url = new URL(apiUrl);
                HttpURLConnection connection = (HttpURLConnection) url.openConnection();
                connection.setRequestMethod("GET");
                connection.connect();

                // Leer la respuesta de la API
                InputStream inputStream = connection.getInputStream();
                BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));
                StringBuilder response = new StringBuilder();
                String line;
                while ((line = bufferedReader.readLine()) != null) {
                    response.append(line);
                }
```

Figura 26 Obtención de datos desde la API

Como se ve en la función anterior se crea una lista de “Alimento” que corresponden a una clase que tiene las variables correspondientes a nombre, energía, proteína, carbohidrato, colesterol, lípidos, gsat y sodio con los valores que se obtiene desde la hoja de cálculo de Google Sheets. Este objeto tiene su constructor, getters y setters correspondientes para su funcionamiento normal.

```
1 usage   AlejoPal
public Alimento(String nombre, String energia, String proteina, String carbohidratos, String colesterol, String lipidos, String gsat, String
    this.nombre = nombre;
    this.energia = energia;
    this.proteina = proteina;
    this.carbohidratos = carbohidratos;
    this.colesterol = colesterol;
    this.lipidos = lipidos;
    this.gsat = gsat;
    this.sodio = sodio;
}
```

Figura 27 Constructor Alimento

Entonces, cuando la consulta a la API resulta satisfactoria y la respuesta ha sido guardada en la variable “jsonResponse” se analiza cada uno de los nodos dentro de la clave “nutrición” que mediante un ciclo pasa uno por uno de los valores de él, extrayendo cada una de las 8 columnas de la hoja de cálculos que corresponden con la clase “Alimento” y mediante su constructor se crea un elemento con esas esas 8 variables, las cuales, posteriormente son agregadas a la lista de salida que entrega la función.

```
// Analizar la respuesta JSON
JSONObject jsonResponse = new JSONObject(response.toString());
JSONArray nutricionArray = jsonResponse.getJSONArray( name: "nutricion");

// Recorrer el arreglo de objetos nutricion y crear instancias de Alimento
List<Alimento> alimentosList = new ArrayList<>();
for (int i = 0; i < nutricionArray.length(); i++) {
    JSONObject nutricionObj = nutricionArray.getJSONObject(i);
    String nombre = nutricionObj.getString( name: "Nombre");
    String energia = nutricionObj.getString( name: "Energia");
    String proteina = nutricionObj.getString( name: "Proteina");
    String carbohidratos = nutricionObj.getString( name: "Carbohidratos");
    String colesterol = nutricionObj.getString( name: "Colesterol");
    String lipidos = nutricionObj.getString( name: "Lipidos");
    String gsat = nutricionObj.getString( name: "Gsat");
    String sodio = nutricionObj.getString( name: "Sodio");

    // Crear una instancia de Alimento y asignar los valores
    Alimento alimento = new Alimento(nombre, energia, proteina, carbohidratos, colesterol, lipidos, gsat, sodio);
    alimentosList.add(alimento);
}

return alimentosList;
```

Figura 28 Creación de lista de alimentos

Teniendo la lista de alimentos ya completa mediante otro ciclo se extrae únicamente el valor del nodo “nombre” para que almacenados en un arreglo se establezcan como adaptador del campo “autoCompleteTextView”.

```
if (result != null) {
    // Configurar el ArrayAdapter con la lista de valores (nombres de los alimentos)
    valuesList = new ArrayList<>();
    for (Alimento alimento : result) {
        Log.d( tag: "INSERTAR",  msg: "Nombre: " + alimento.getNombre());
        valuesList.add(alimento.getNombre());
    }

    // Configurar el ArrayAdapter con la lista de alimentos
    ArrayAdapter<String> adapter = new ArrayAdapter<>( context: AgregarAlimento.this, android.R.layout.simple_dropdown_item_1line,

    // Asignar el adaptador al AutoCompleteTextView
    autoCompleteTextView.setAdapter(adapter);
```

Figura 29 Configuración AutoCompleteTextView

De esta manera cuando se selecciona una de las opciones que aparecen en la lista que se despliega siempre va a corresponder con los nombres que se tiene en la base de datos y no se podrá seleccionar algún alimento que no esté presente.

Figura 30 Mensaje instructivo sobre funcionamiento

Cuando se vaya a ingresar el peso sale un mensaje que explica que para finalizar el proceso tiene que presionar finalizar al agregar todos los elementos. Además, que cuando se presiona añadir y no se ha ingresado ningún peso muestra un mensaje diciendo “Por favor ingresa un peso valido y selecciona un alimento” en caso de que alguno de los dos campos este mal diligenciado.

Figura 31 Mensaje de advertencia por información incompleta

Cuando se ingresa el peso ingerido del producto seleccionado se presiona añadir para que este se almacene en memoria y en caso de que quiera seguir agregando alimentos pueda colocar otro. Ya que en la TCAC los valores de aporte nutricional están dados en 100 gramos de alimentos se hace una proporción para sacar la información que corresponda al peso ingresado.

Figura 32 Vista agregar alimento diligenciada

Cuando se presione finalizar, ya ingresado los alimentos ingeridos se actualizan en la base de datos los valores para la comida correspondiente del día en cuestión. Además, se crea un nuevo ítem que va a almacenar la información de cada uno de los alimentos ingeridos para que en la vista de Información General se pueda tener un control más detallado de la información.

Figura 33 Estructura en firebase luego de agregar alimentos

Por último, al final de la vista se encuentra un mensaje con el texto “Consulta los alimentos registrados aquí” que es un vínculo a la TCAC en pdf de la que se está sacando la información de cada uno de los alimentos.

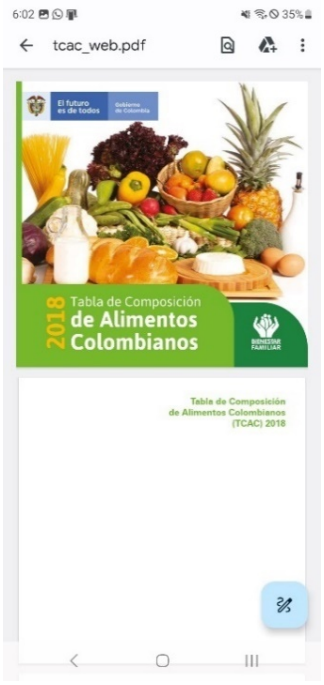


Figura 34 PDF de TCAC

4.2.6 **Información general**

Se muestra toda la información obtenida en los procesos anteriores mediante una consulta a la base de datos ya que toda la información ha sido almacenada en la nube.

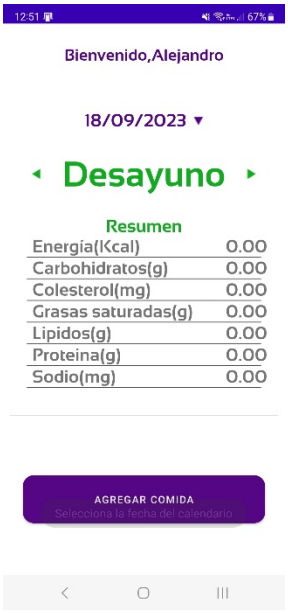


Figura 35 Vista información general

Las interacciones de esta vista se presentan a continuación:

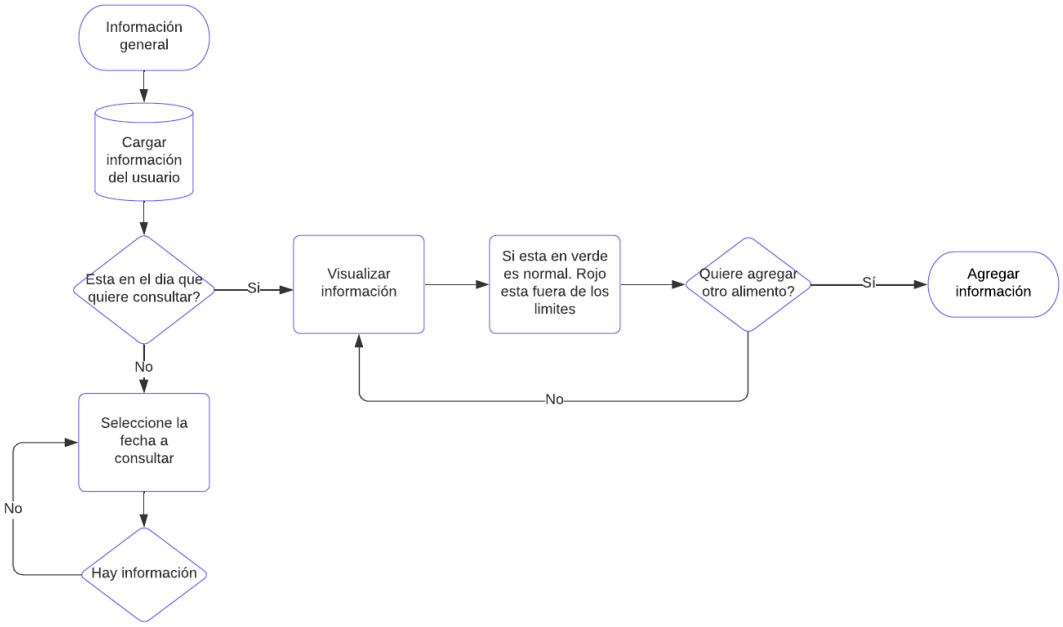


Figura 36 Diagrama de flujo Información general

Para lograr este resultado la vista se compone de dos partes. En la primera se tiene colocado el nombre y la fecha que al momento de presionarla despliega el widget del calendario de Google para que se pueda interactuar con todo el calendario y navegar por los diferentes meses y seleccionar un día de manera interactiva. Este proceso se hace con la función DatePickerDialog como se muestra a continuación:

```
textFecha.setOnClickListener(new View.OnClickListener() {
    AlejoPal
    @Override
    public void onClick(View v) {
        Calendar cal = Calendar.getInstance();
        int anio = cal.get(Calendar.YEAR);
        int mes = cal.get(Calendar.MONTH);
        int dia = cal.get(Calendar.DAY_OF_MONTH);

        AlejoPal
        DatePickerDialog dpd = new DatePickerDialog( context: InformacionGeneral.this,R.style.DatePickerDialogTheme, new DatePickerDialog.(
            1 usage  AlejoPal
            @Override
            public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
                String fecha = dayOfMonth + "/" + String.format("%02d", (month + 1)) + "/" + year; // Formato de mes con dos dígitos
                String fechaSeleccionada = "" + year + String.format("%02d", (month + 1)) + String.format("%02d", dayOfMonth); // Format

                obtenerInformacionComidas(fechaSeleccionada, correo);
                textFecha.setText(fecha);
            }
        }, anio, mes, dia);
        dpd.show();
    }
});
```

Figura 37 Interacción para seleccionar fecha

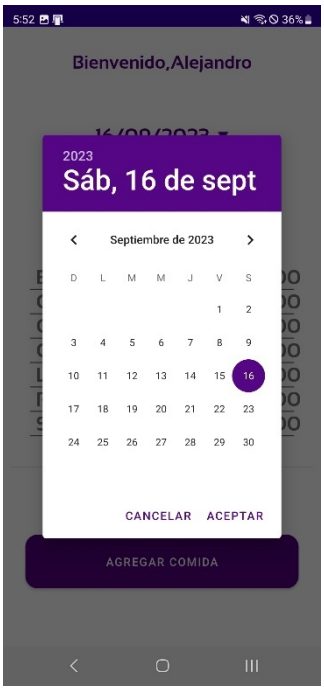


Figura 38 Widget de calendario desplegado

En caso de que la fecha seleccionada en el calendario no corresponda con una fecha ingresada previamente y no se encuentre información en la base de datos va a desplegar un mensaje “No se encontraron datos en la base de datos”.

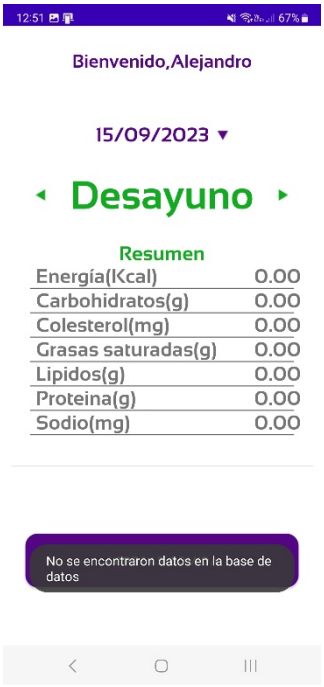


Figura 39 Mensaje de advertencia al no encontrar información de esa fecha

En caso de seleccionar una fecha que tiene información en la base de datos la información que se despliega en pantalla cambia por los elementos que tenga en cada ítem. Para implementar esto se implementa un view pager que permite crear una serie de vistas con un formato especifico que se puedan desplazar horizontalmente.

En las figuras a continuación se muestra como se ve cada una de las ventanas del view pager en sus 3 posiciones:

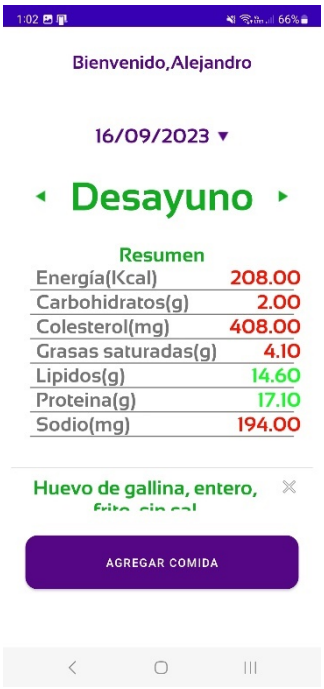


Figura 40 Información del desayuno cuando se ingresa

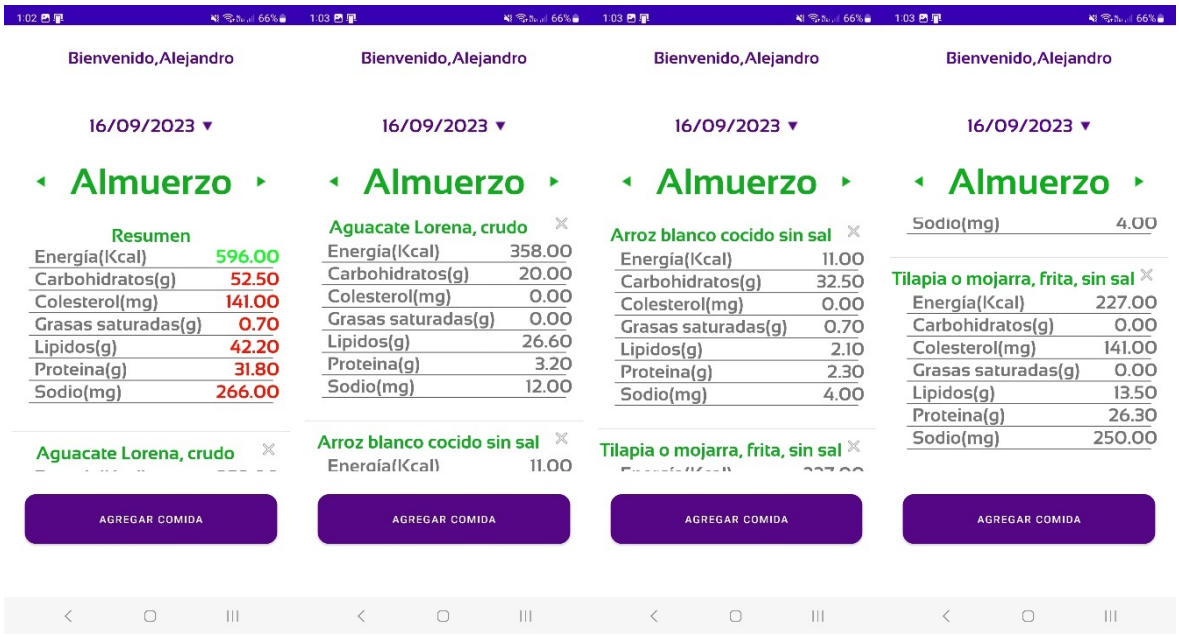


Figura 41 Información del almuerzo con varios alimentos agregados

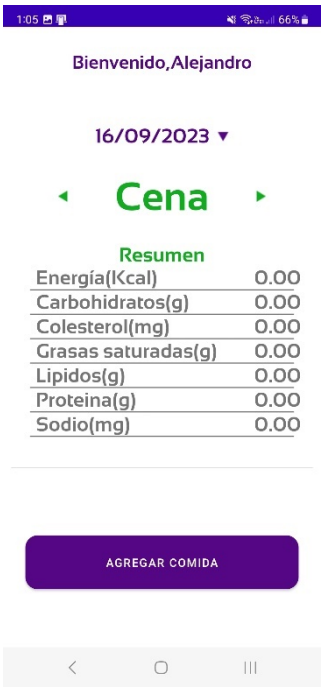


Figura 42 Información de cena sin completar

Para lograr estas interacciones es necesario modularizar las funciones para que de manera independiente se adapte a la cantidad de información que hay en la base de datos ya que se desconoce la cantidad de alimentos que el usuario vaya a agregar por comida.

Al crear la vista, usando una función llamada ObtenerInformacionComidas()se hace una consulta a la base de datos en la dirección del correo y la fecha seleccionada.

```
private void obtenerInformacionComidas(String fechaSeleccionada, String correo) {
    // Crear una lista vacía para cada comida
    List<ViewPagerItem> listaDesayuno = new ArrayList<>();
    List<ViewPagerItem> listaAlmuerzo = new ArrayList<>();
    List<ViewPagerItem> listaCena = new ArrayList<>();

    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference mRootReference = database.getReference();

    // Obtener la referencia al nodo del usuario
    DatabaseReference databaseRef = mRootReference.child( pathString: "Usuario").child(correo).child(fechaSeleccionada);

    AlejoPal
    databaseRef.addListenerForSingleValueEvent(new ValueEventListener() {
```

Figura 43 Consulta base de datos

Cuando la consulta a la base de datos en el nodo de referencia exista se procede a realizar un ciclo para pasar por cada uno de los nodos hijos que serán las 3 comidas diarias. En primer lugar, se obtiene la información de los valores acumulados de

cada uno de los alimentos que estén almacenados que cuando se capturan son almacenados en un objeto del tipo “TablaItem” que este a su vez será guardado en una lista para luego ser usado en el constructor del view pager.

```
for (DataSnapshot comidaSnapshot : dataSnapshot.getChildren()) {
    String comida = comidaSnapshot.getKey();

    // Obtener la información de la comida actual
    List<TablaItem> alimentosList = new ArrayList<>();
    Double carbohidratos = comidaSnapshot.child( path: "Carbohidratos").getValue(Double.class);
    Double colesterol = comidaSnapshot.child( path: "Colesterol").getValue(Double.class);
    Double energia = comidaSnapshot.child( path: "Energia").getValue(Double.class);
    Double gsat = comidaSnapshot.child( path: "Gsat").getValue(Double.class);
    Double lipidos = comidaSnapshot.child( path: "Lipidos").getValue(Double.class);
    Double proteinas = comidaSnapshot.child( path: "Proteina").getValue(Double.class);
    Double sodio = comidaSnapshot.child( path: "Sodio").getValue(Double.class);

    // Formatea los valores con un máximo de dos decimales
    String carbohidratosFormatted = String.format("%.2f", carbohidratos);
    String colesterolFormatted = String.format("%.2f", colesterol);
    String energiaFormatted = String.format("%.2f", energia);
    String gsatFormatted = String.format("%.2f", gsat);
    String lipidosFormatted = String.format("%.2f", lipidos);
    String proteinasFormatted = String.format("%.2f", proteinas);
    String sodioFormatted = String.format("%.2f", sodio);

    // Crea un nuevo objeto TablaItem usando los valores formateados
    TablaItem tablaItemR = new TablaItem(fechaSeleccionada,correo, tiempo: "Resumen",  comidas: "Resumen", carbohidratosFormatte
    alimentosList.add(tablaItemR);
```

Figura 44 Obtención de la información del resumen de los alimentos

Ya con la lectura de los elementos analizados se entra dentro del nodo alimentos de cada una de las comidas que tiene la información de las comidas ingeridas. En caso de que no haya este nodo solo se realiza la primera lectura general y se deja el proceso. En cada uno de los hijos del nodo “Alimentos” se realiza el mismo proceso para obtener los valores nutricionales de cada uno de los alimentos y su nombre para ser agregados a la misma lista usando en formato “TablaItem”.

```
DataSnapshot alimentosSnapshot = comidaSnapshot.child( path: "Alimentos");
for (DataSnapshot alimentoSnapshot : alimentosSnapshot.getChildren()) {
    String alimento = alimentoSnapshot.getKey();
    Double carbohidratosA = alimentoSnapshot.child( path: "Carbohidratos").getValue(Double.class);
    Double colesterolA = alimentoSnapshot.child( path: "Colesterol").getValue(Double.class);
    Double energiaA = alimentoSnapshot.child( path: "Energia").getValue(Double.class);
    Double gsatA = alimentoSnapshot.child( path: "Gsat").getValue(Double.class);
    Double lipidosA = alimentoSnapshot.child( path: "Lipidos").getValue(Double.class);
    Double proteinasA = alimentoSnapshot.child( path: "Proteina").getValue(Double.class);
    Double sodioA = alimentoSnapshot.child( path: "Sodio").getValue(Double.class);

    // Formatea los valores con un máximo de dos decimales
    String carbohidratosFormatted1 = String.format("%.2f", carbohidratosA);
    String colesterolFormatted1 = String.format("%.2f", colesterolA);
    String energiaFormatted1 = String.format("%.2f", energiaA);
    String gsatFormatted1 = String.format("%.2f", gsatA);
    String lipidosFormatted1 = String.format("%.2f", lipidosA);
    String proteinasFormatted1 = String.format("%.2f", proteinasA);
    String sodioFormatted1 = String.format("%.2f", sodioA);

    // Crea un objeto TablaItem con los valores formateados
    TablaItem tablaItem = new TablaItem(fechaSeleccionada,correo,comida,alimento, carbohidratosFormatted1, colesterolFo
    alimentosList.add(tablaItem);
}
```

Figura 45 Obtención de información de los alimentos agregados

Ya con todos los elementos almacenados en la lista “alimentoslist” se crea cada una de las posiciones del view pager para cada una de las comidas diarias según el nodo de comida que se esté analizando.

```
ViewPagerItem viewPagerItem = new ViewPagerItem(alimentosList, comida);

// Agregar el objeto ViewPagerItem a la lista correspondiente
if (comida.equals("Desayuno")) {
    listaDesayuno.add(viewPagerItem);
} else if (comida.equals("Almuerzo")) {
    listaAlmuerzo.add(viewPagerItem);
} else if (comida.equals("Cena")) {
    listaCena.add(viewPagerItem);
}
```

Figura 46 Creación de cada una de las posiciones del view pager

Este proceso se realiza para Desayuno, Almuerzo y Cena independientemente de la cantidad de elementos que tenga cada uno.

El view pager es un recurso que se utiliza de manera separada a la vista que permite desplazarse horizontalmente entre fragmentos. El diseño del view pager se presenta a continuación:

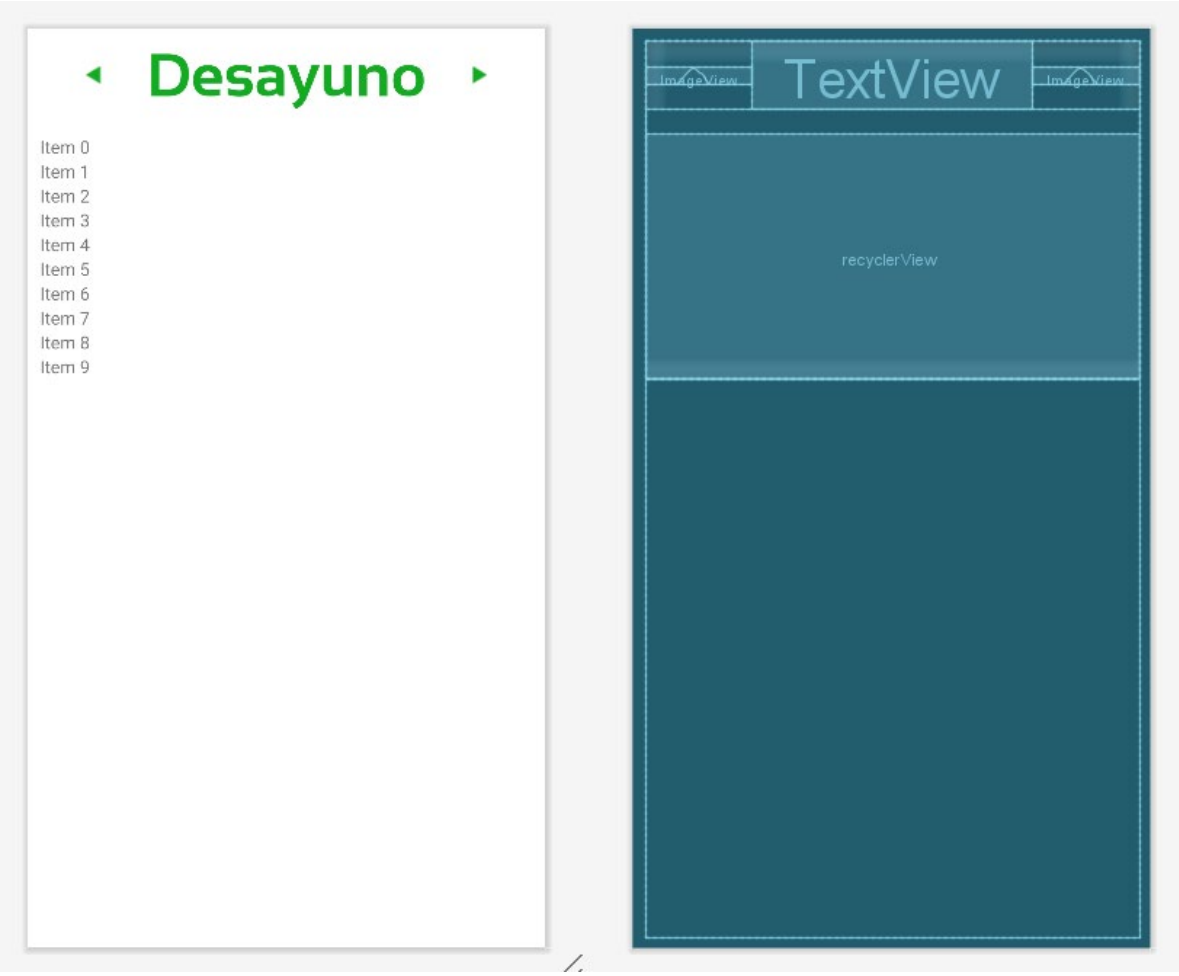


Figura 47 Interfaz del view pager

La vista del view pager está compuesta por un campo de texto que va a cambiar según el parámetro que se le ingrese en el constructor además de un recycler view que permite crear una cantidad indefinida de elementos según el requerimiento lo indique.

```
1 usage   AlejoPal
public ViewPagerItem(List<TablaItem> tablaItems, String Alimento) {
    this.tablaItems = tablaItems;
    this.Alimento = Alimento;

}
```

Figura 48 Constructor view pager

El constructor del view pager tiene como parámetros dos elementos: una lista de objetos “TablaItem” y una variable alimento que indica cada una de las comidas diarias.

Para el correcto funcionamiento del view pager es necesario crear un adaptador con el fin de asignar a cada uno de los desplazamientos la información correcta. En la función onBindViewHolder se define que elemento se va a asignar a cada una de las posiciones.

```
@Override
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
    ViewPagerItem viewPagerItem = viewPagerItemsList.get(position);
    List<TablaItem> tablaItems = viewPagerItem.getTablaItems();

    String alimento = viewPagerItem.getAlimento(); // Obtén el nombre del alimento

    TablaAdapter tablaAdapter = new TablaAdapter(tablaItems, alimento); // Pasa el alimento
    holder.recyclerView.setAdapter(tablaAdapter);
    holder.txtAlimento.setText(viewPagerItem.getAlimento());
}
```

Figura 49 Adaptador view pager

Como este elemento tiene el componente recycler view se requiere definir los parámetros que se van a ingresar. En la figura 43 se muestra la estructura de este componente que se encarga de generar cada una de las tablas según la cantidad de ítems que tenga el listado entregado por el view pager.

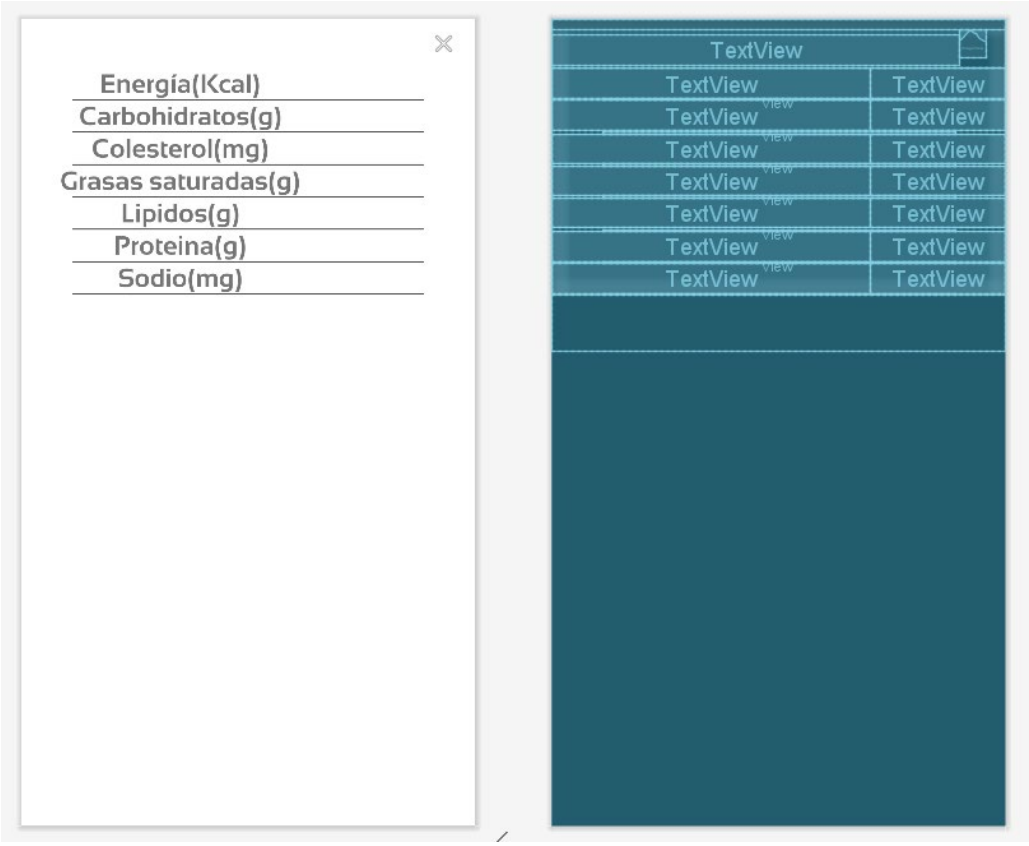


Figura 50 Vista recycler view

Dentro de los componentes que se encuentran en el recycler view se encuentra un botón en forma de “X” que cumple la función de eliminar el elemento de la vista y de la base de datos.

En el constructor de este componente se reciben los valores de fecha, correo, comida, nombre, energía, proteína, carbohidrato, colesterol, lípidos, gsat y sodio que van a ser reemplazados en la ubicación correspondiente al campo de texto que sea definido en su adaptador.

```
public TablaItem(String fecha, String correo, String tiempo, String comidas, String carbohidratos, String colesterol, String energia,

    this.fecha = fecha;
    this.correo = correo;
    this.tiempo = tiempo;
    this.comidas = comidas;
    this.carbohidratos = carbohidratos;
    this.colesterol = colesterol;
    this.energia = energia;
    this.gsat = gsat;
    this.lipidos = lipidos;
    this.proteinas = proteinas;
    this.sodio = sodio;
}
```

Figura 51 Constructor recycler view

En la interacción del componente se tendrá en cuenta los elementos que tenga el constructor que en primer lugar asignará los valores a cada uno de los campos de texto:

```
// Asignar los valores de TablaItem a las vistas en la fila del RecyclerView
holder.txtAlimentos.setText(item.getComidas());
holder.txtcarbohidratos.setText(item.getCarbohidratos());
holder.txtcolesterol.setText(item.getColesterol());
holder.txtenergia.setText(item.getEnergia());
holder.txtgsaturadas.setText(item.getGsat());
holder.txtlipidos.setText(item.getLipidos());
holder.txtproteinas.setText(item.getProteinas());
holder.txtsodio.setText(item.getSodio());
```

Figura 52 Adaptador recycler view

Luego solo para el elemento en la posición 0 que corresponde a la tabla con la información del resumen de todos los alimentos ingeridos se cambia el color del texto teniendo en cuenta unos valores de referencia según la comida que se vaya a tener como se ve en la tabla 1. Donde rojo indica que los valores no están en los dentro del rango correcto, verde que está en niveles buenos y en gris cuando no se ha realizado ningún cambio a este valor.

Carbohidratos (g)	
Desayuno	45 - 60
Almuerzo	60 - 80
Cena	35 - 45

Colesterol (mg)	
Desayuno	0 - 100
Almuerzo	0 - 100
Cena	0 - 100

Energía (kcal)	
Desayuno	350 - 400
Almuerzo	550 -600
Cena	400 - 450

Lípidos (g)	
Desayuno	13 - 18
Almuerzo	18 - 24
Cena	13 - 18

Proteína (g)	
Desayuno	15 - 20
Almuerzo	20 - 30
Cena	12 - 025

Sodio (mg)	
Desayuno	300 - 400
Almuerzo	400 - 600
Cena	300 - 400

Grasas saturadas (g)	
Desayuno	13 - 18
Almuerzo	18 - 24
Cena	13 - 18

Tabla 1 Limite para alimentos en cada comida

Mediante una función se hace el análisis de los límites para cambiar el color

```
    if (position == 0) {
        double carbohidratosValue = Double.parseDouble(item.getCarbohidratos());
        double colesterolValue = Double.parseDouble(item.getColesterol());
        double energiaValue = Double.parseDouble(item.getEnergia());
        double gsatValue = Double.parseDouble(item.getGsat());
        double lipidosValue = Double.parseDouble(item.getLipidos());
        double proteinasValue = Double.parseDouble(item.getProteinas());
        double sodioValue = Double.parseDouble(item.getSodio());
        changeTextColorBasedOnValue(holder.txtcarbohidratos, carbohidratosValue, limID: 45, limSD: 60, limIA: 60, limSA: 80, limIC: 35, limSC: 45);
        changeTextColorBasedOnValue(holder.txtcolesterol, colesterolValue, limID: 0, limSD: 100, limIA: 0, limSA: 100, limIC: 0, limSC: 100);
        changeTextColorBasedOnValue(holder.txtenergia, energiaValue, limID: 350, limSD: 400, limIA: 550, limSA: 600, limIC: 400, limSC: 450);
        changeTextColorBasedOnValue(holder.txtgsaturadas, gsatValue, limID: 13, limSD: 18, limIA: 18, limSA: 24, limIC: 13, limSC: 18);
        changeTextColorBasedOnValue(holder.txtlipidos, lipidosValue, limID: 13, limSD: 18, limIA: 18, limSA: 24, limIC: 13, limSC: 18);
        changeTextColorBasedOnValue(holder.txtproteinas, proteinasValue, limID: 15, limSD: 20, limIA: 20, limSA: 30, limIC: 12, limSC: 25);
        changeTextColorBasedOnValue(holder.txtsodio, sodioValue, limID: 300, limSD: 400, limIA: 400, limSA: 600, limIC: 300, limSC: 500);
    }
}
7 usages   AlejoPal
private void changeTextColorBasedOnValue(TextView textView, double value, double limID, double limSD, double limIA, double limSA, double li
    if (alimento.equals("Desayuno")) {
        if ((value > limSD || value < limID) && value != 0) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorPeligro));
        } else if (value > limID && value < limSD) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorEstable));
        }
    } else if (alimento.equals("Almuerzo")) {
        if ((value > limSA || value < limIA) && value != 0) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorPeligro));
        } else if (value > limIA && value < limSA) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorEstable));
        }
    }else if (alimento.equals("Cena")) {
        if ((value > limSC || value < limIC) && value != 0) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorPeligro));
        } else if (value > limIC && value < limSC) {
            textView.setTextColor(textView.getContext().getColor(R.color.colorEstable));
        }
```

Figura 53 Función de cambio de color de texto

Analizando la última funcionalidad de este componente se analiza el botón de eliminar un alimento. Como se tiene que hacer un cambio en la base de datos y actualizar la información en el resumen de los datos es necesario actualizar todo el nodo del alimento en la fecha que se está haciendo el cambio. Para esto se elimina el elemento de la base de datos y se realiza otra vez el ciclo para sacar el valor del resumen con la nueva cantidad de elementos que se tiene en la base de datos.

```
alimentosRef.addListenerForSingleValueEvent(new ValueEventListener() {
    2 usages  AlejoPal *
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for (DataSnapshot alimentoSnapshot : dataSnapshot.getChildren()) {
            // Obtén los valores de cada alimento y suma a los totales
            double carbohidratos = alimentoSnapshot.child( path: "Carbohidratos").getValue(Double.class);
            double colesterol = alimentoSnapshot.child( path: "Colesterol").getValue(Double.class);
            double energia = alimentoSnapshot.child( path: "Energia").getValue(Double.class);
            double gsat = alimentoSnapshot.child( path: "Gsat").getValue(Double.class);
            double lipidos = alimentoSnapshot.child( path: "Lipidos").getValue(Double.class);
            double proteinas = alimentoSnapshot.child( path: "Proteina").getValue(Double.class);
            double sodio = alimentoSnapshot.child( path: "Sodio").getValue(Double.class);

            totalCarbohidratos[0] += carbohidratos;
            totalColesterol[0] += colesterol;
            totalEnergia[0] += energia;
            totalGsat[0] += gsat;
            totalLipidos[0] += lipidos;
            totalProteinas[0] += proteinas;
            totalSodio[0] += sodio;
        }

        // Actualiza los valores en el nodo "Resumen" en Firebase
        DatabaseReference resumenRef = mRootReference.child( pathString: "Usuario")
                .child(itemToRemove.getCorreo())
                .child(itemToRemove.getFecha())
                .child(itemToRemove.getTiempo());


        resumenRef.child( pathString: "Carbohidratos").setValue(totalCarbohidratos[0]);
        resumenRef.child( pathString: "Colesterol").setValue(totalColesterol[0]);
        resumenRef.child( pathString: "Energia").setValue(totalEnergia[0]);
        resumenRef.child( pathString: "Gsat").setValue(totalGsat[0]);
        resumenRef.child( pathString: "Lipidos").setValue(totalLipidos[0]);
        resumenRef.child( pathString: "Proteina").setValue(totalProteinas[0]);
        resumenRef.child( pathString: "Sodio").setValue(totalSodio[0]);
```

Figura 54 Función para actualizar información luego de borrar un elemento

Para finalizar el proceso se notifica que se acaba de hacer un cambio en los ítems del recycler view para que se actualicen y cambien las posiciones. Y de esta manera cada vez que se elimine un objeto no van a haber errores con los valores mostrados ya que se evitan al realizar la consulta a la base de datos cada vez que hay un cambio.

En los anexos del documento se incluye el diagrama de flujo completo que detalla todas las interacciones y la estructura integral de la aplicación.

5. RECOMENDACIONES

Teniendo en cuenta la arquitectura de la aplicación cualquier funcionalidad adicional para procesar la información podría ser implementada ya que los datos recolectados se encuentran almacenados en la nube.

En caso de que el ICBF actualice la información de la tabla de composición de alimentos colombianos se podrán modificar los cambios en la hoja de cálculos de Google Sheets para evitar procesar nuevamente la información.

Teniendo en cuenta a las estructuras de la base de datos se puede cambiar por una base de datos relacional que permita conectar los datos de los usuarios, alimentos y comidas diarias mediante el uso de llaves foráneas que agilice el proceso de consultas.

Además, se pueden agregar más protocolos de seguridad en la base de datos en tiempo real para que el acceso a la información de los patrones alimenticios de los usuarios se encuentre bajo una capa mayor de protección.

6. CONCLUSIONES

- La aplicación desarrollada permite almacenar la información de manera independiente de cada una de las 3 comidas diarias, además de mostrar los alimentos ingeridos, su composición nutricional individual y un resumen de cada una de las comidas diarias mostrando una advertencia según los niveles establecidos de una dieta saludable. De esta manera el usuario podrá gestionar y mejorar su dieta siendo consciente de sus hábitos alimenticios.
- Ya que la tabla de composición de alimentos colombianos recopila el contenido de nutrientes y aporte de energía de los alimentos por cada 100 gramos de alimento se saca la proporción con respecto a la cantidad ingresada por el usuario para tener el valor de los nutrientes ingeridos.
- Usando los servicios de Firebase se crea una comunicación estable entre el dispositivo del usuario y el servidor que brinda la característica de sincronización de los datos en tiempo real para que la información esté actualizada en todo momento.
- Cada una de las funcionalidades del aplicativo fue puesta a prueba en diferentes condiciones con el objetivo de identificar puntos débiles o fallos en el funcionamiento del aplicativo. De esta manera se puede asegurar que cada una de las características operan de manera correcta y consistente cumpliendo los requisitos y proporcionando una experiencia de usuario fluida y satisfactoria.

7. BIBLIOGRAFÍA

[1]. UNICEF. (2019, 15 de octubre). La mala alimentación perjudica la salud de los niños en todo el mundo, advierte UNICEF. [Comunicado de prensa]. https://www.unicef.org/es/comunicados-prensa/la-mala-alimentaci%C3%B3n-perjudica-la-salud-de-los-ni%C3%B1os-en-todo-el-mundo-advierte

[2]. UNICEF. (2019). Estado Mundial de la Infancia 2019. [Informe]. https://www.unicef.org/es/informes/estado-mundial-de-la-infancia-2019

[3]. Asociación Española de Dietistas-Nutricionistas. (2017). Introducción a la alimentación y la salud. [Documento PDF]. http://www.asociacionasaco.es/wp-content/uploads/2018/01/Introducci%C3%B3n.-Alimentaci%C3%B3n-y-Salud.pdf

[4]. Público. (2018, 3 de enero). 5 apps para conocer el valor nutricional de los alimentos. [Artículo de blog]. https://www.publico.es/ahorro-consumo-responsable/5-apps-para-conocer-el-valor-nutricional-de-los-alimentos/

[5]. Organización Mundial de la Salud. (2021, 16 de marzo). Malnutrición. [Hoja informativa]. https://www.who.int/es/news-room/fact-sheets/detail/malnutrition#:~:text=Las%20enfermedades%20no%20transmisibles%20relacionadas%20con%20la%20alimentaci%C3%B3n%20abarcan%20las,algunos%20c%C3%A1nceres%2C%20y%20la%20diabetes.

[6]. Proyecto Evite. (s. f.). ¿Qué niveles de IMC y perímetro abdominal debo mantener? [Imagen]. Recuperado el 3 de mayo de 2023, de https://proyectoevite.es/que-niveles-de-imc-y-perimetro-abdominal-debo-mantener/

[7]. Organización Mundial de la Salud. (2015, 14 de octubre). Obesity and diabetes: The slow-motion disaster. Keynote address at the 47th meeting of the National Academy of Medicine. [Discurso]. https://www.who.int/es/director-general/speeches/detail/obesity-and-diabetes-the-slow-motion-disaster-keynote-address-at-the-47th-meeting-of-the-national-academy-of-medicine

[8]. Android. (s.f.). ¿Qué es Android? Recuperado de: https://www.android.com/intl/es_es/what-is-android/

[9]. Digital55. (s.f.). Qué es Firebase: Funcionalidades, ventajas y conclusiones. Recuperado de: https://digital55.com/blog/que-es-firebase-funcionalidades-ventajas-conclusiones/

[10]. Firebase. (s.f.). Firebase. Recuperado de: https://firebase.google.com/?hl=es

[11]. Java. (s.f.). ¿Qué es Java? Recuperado de: https://www.java.com/es/download/help/whatis_java.html

[12]. Amazon Web Services. (s.f.). ¿Qué es Java? Recuperado de: https://aws.amazon.com/es/what-is/java/

[13]. Hostinger. (s.f.). ¿Qué es GitHub? Recuperado de https://www.hostinger.co/tutoriales/que-es-github#%C2%BFQue_es_GitHub

[14]. Android Developers. (s.f.). Versiones de plataformas Android. Recuperado de: https://developer.android.com/studio/releases/platforms?hl=es-419

[15]. SocialDiabetes. (s.f.). Aprendiendo a contar carbohidratos. Recuperado de:https://blog.socialdiabetes.com/aprendiendo-a-contar-carbohidratos/#:~:text=M%C3%A9todo%20de%20conteo%20de%20carbohidratos,-Existen%20alimentos%20saludables&text=La%20ADA%20(Asociaci%C3%B3n%20Americana%20de,carbohidratos%20en%20cada%20comida%20principal.

[16]. MedicalNewsToday. (s.f.). ¿Qué es el colesterol bueno? Recuperado de: https://www.medicalnewstoday.com/articles/es/que-es-colesterol-bueno#Resumen

[17]. Ministerio de Sanidad. (s.f.). Alimentación saludable - Distribución diaria. Recuperado de: https://estilosdevidasaludable.sanidad.gob.es/alimentacionSaludable/queSabemos/enLaPractica/distribuir/diario/home.htm

[18]. MedlinePlus. (s.f.). Consejos sobre grasas. Recuperado de: https://medlineplus.gov/spanish/ency/patientinstructions/000838.htm#:~:text

=No%20debe%20obtener%20m%C3%A1s%20del,de%20grasas%20saturadas%20al%20d%C3%ADa.

[19]. COPE. (2018). ¿Cuánta grasa podemos consumir al día? Recuperado de: https://www.cope.es/blogs/t-cuidamos/2018/02/26/cuanta-grasa-podemos-consumir-diario/

[20]. Mayo Clinic News Network. (2017). ¿Está consumiendo demasiada proteína? Recuperado de: https://newsnetwork.mayoclinic.org/es/2017/03/15/esta-consumiendo-demasiada-proteina/#:~:text=Las%20recomendaciones%20generales%20son%20de,de%20una%20sesi%C3%B3n%20de%20ejercicio.

[21]. Organización Mundial de la Salud. (s.f.). Reducción de la sal. Recuperado de: https://www.who.int/es/news-room/fact-sheets/detail/salt-reduction#:~:text=Para%20los%20adultos%3A%20la%20OMS,sal%20por%20d%C3%ADa%20(1).

8. MANUAL DE USO

Reconocimiento de la aplicación

La aplicación móvil una vez instalada en su dispositivo móvil se visualiza con el nombre quimo y un icono representado de la siguiente forma:

Figura 55 Icono de la aplicación en el dispositivo

Vista Principal

Al tocar el icono de la aplicación, se abrirá la siguiente pantalla de inicio:

Figura 56 Vista principal de la aplicación

Una vez cargada la pantalla principal o de bienvenida, se encontrará un botón de inicio de sesión y un de registro los cuales de maneras diferentes le permitirán el acceso a la aplicación.

Si ya se encuentra registrado omita el paso de registro y presione el botón de inicio de sesión.

Registro

Si aún no cuenta con un usuario de ingreso, presione el botón de registrarse, el cual, lo llevará a una nueva pantalla, en la pantalla de registro encontrará cuatro recuadros donde se le pedirá información pertinente como lo son su nombre, email, contraseña y la confirmación de su contraseña, para un registro exitoso.

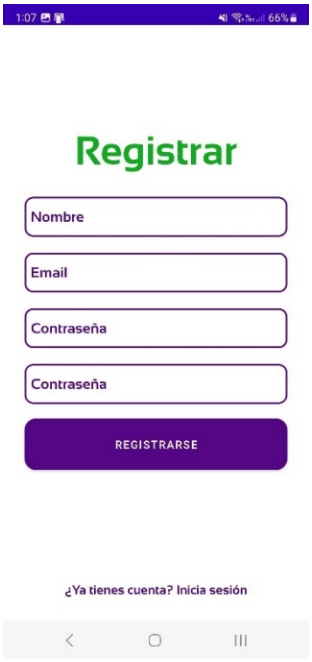


Figura 57 Vista de registro de usuario

Relleno de Datos

Su nombre será necesario para la identificación personal al momento de estar en contacto con el aplicativo.

El email será su usuario de ingreso.

La contraseña será la manera en que el ingreso se podrá realizar de manera segura y solo por el usuario que asigno la contraseña o de manera autorizada, en este campo es necesario que el usuario ingrese una contraseña de al menos seis caracteres

numéricos y posteriormente su rectificación para confirmar que la contraseña sea la deseada.

Una vez completado cada uno de los campos requeridos podrá dirigirse a la pantalla de inicio de sesión oprimiendo en la parte inferior de la pantalla, en el siguiente texto:

¿Ya tienes cuenta? Inicia sesión

Inicio de sesión

Al oprimir en el botón inicio de sesión que se encuentra en la pantalla principal o en la parte inferior de registro, se abrirá la siguiente pantalla.

Figura 58 Vista de inicio de sesión

En esta pantalla podrá ingresar los datos antes dados en el registro, para garantizar un acceso exitoso. Si aún no completa su registro, por favor vuelva al paso de REGISTRO mediante el siguiente vinculo.

¿No tienes cuenta? Regístraste

Al terminar el proceso de inicio de sesión aparecerá la siguiente pantalla:

Figura 59 Vista de elección comidas del día

En esta pantalla se brindan tres botones (desayuno, almuerzo y cena) que brindarán la opción de agregar los alimentos a ingerir o ingeridos, un botón de historial y un vínculo para descargar el manual de usuario presentado aquí.

Agregar comidas

En caso de seleccionar alguno de los tres botones para agregar alimentos (desayuno, almuerzo y cena) se desplazará la siguiente vista:

Figura 60 Vista para agregar alimentos a comida del día

En esta ventana encontrará dos campos para llenar información, el primero le permitirá ingresar el alimento deseado, mediante se busque el alimento se desplazará un cuadro donde le permitirá visualizar las coincidencias con el alimento a seleccionar para una búsqueda eficaz, si al ingresar el alimento no se genera una sugerencia, significará que el alimento no se encuentra registrado en la Tabla de Composición de Alimentos Colombianos.

Figura 61 Búsqueda de alimentos

En caso de querer la certeza de cuales alimentos se encuentran a disposición, puede verificar en la TCAC brindada por el ICBF. El cual encontrara en la parte inferior de la vista de la siguiente manera.

Ya seleccionado el alimento deberá ingresar el peso correspondiente en gramos a la comida ingresada y presionar el botón añadir.

En caso de querer seguir agregando alimentos, repita el proceso de AGREGAR COMIDAS.

Si considera que ya ha ingresado todos los alimentos deseados, oprima el botón finalizar ubicado debajo de añadir.

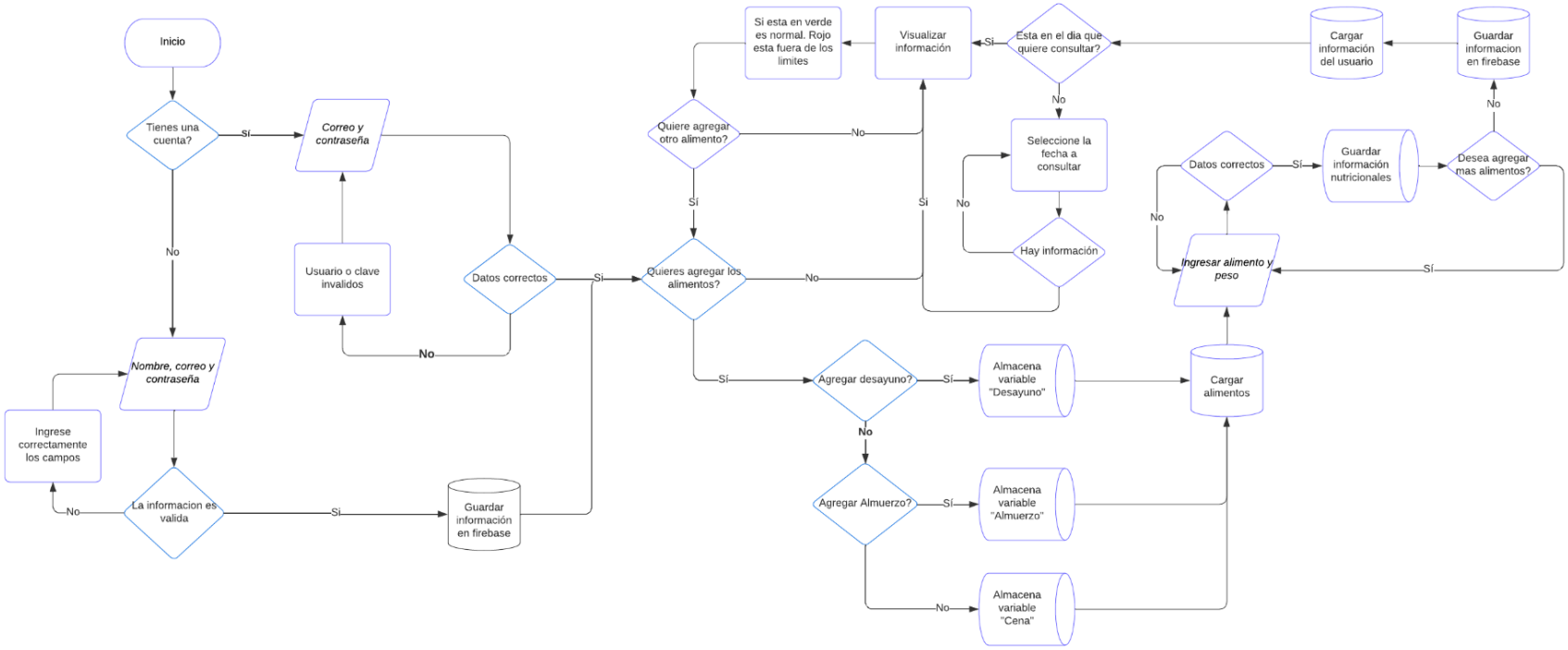
Historial

Se encontrará también un botón que le permitirá como su nombre lo indica ir a la ventana de historial, la cual también se desplegará al terminar el proceso de agregar comidas.

Figura 62 Vista de historial

En esta vista se mostrará la información procesada de todos los alimentos ingeridos en las 3 comidas diarias. Para visualizar las otras comidas podrá desplazarse de izquierda a derecha para navegar entre las ventanas.

9. ANEXOS


Inicio
Tienes una cuenta?
Sí
Correo y contraseña
No
Nombre, correo y contraseña
Ingrese correctamente los campos
No
La informacion es valida
Sí
Guardar información en firebase
Usuario o clave invalidos
No
Datos correctos
Sí
Quieres agregar los alimentos?
No
Sí
Quiere agregar otro alimento?
No
Sí
Si esta en verde es normal. Rojo esta fuera de los limites
Visualizar información
Sí
Esta en el dia que quiere consultar?
No
Seleccione la fecha a consultar
Hay información
No
Sí
Agregar desayuno?
Sí
Almacena variable "Desayuno"
No
Agregar Almuerzo?
Sí
Almacena variable "Almuerzo"
No
Almacena variable "Cena"
Cargar alimentos
Ingresar alimento y peso
Datos correctos
Sí
Guardar información nutricionales
Desea agregar mas alimentos?
No
Sí
No
Cargar información del usuario
Guardar informacion en firebase
No