Descripción técnica de DiabeticAwareness2.0

Contenido

[Introducción 1](#_Toc140143904)

[Ambiente de desarrollo 1](#_Toc140143905)

[Versiones 1](#_Toc140143906)

[Estructura de la aplicación 2](#_Toc140143907)

[Paquete “controllers” 3](#_Toc140143908)

[Paquete “my\_clases” 4](#_Toc140143909)

[Paquete “my\_widgets” 4](#_Toc140143910)

[Paquete “pages” 5](#_Toc140143911)

[Repository 6](#_Toc140143912)

[Archivos fuera de paquetes 6](#_Toc140143913)

[Assets 6](#_Toc140143914)

# Introducción

En este documento se hace la descripción de los aspectos técnicos con los cuales fue desarrollada la aplicación móvil denominada como DiabeticAwareness2.0. La intención es proporcionar la información suficiente para que futuros colaboradores puedan configurar su ambiente de desarrollo para continuar con la construcción o proporcionar mantenimiento a la aplicación.

# Ambiente de desarrollo

La aplicación fue desarrollada usando la plataforma Flutter la cual permite el desarrollo simultáneo de aplicaciones para Android y para iOS usando el lenguaje de desarrollo Dart en el IDE Android Studio.

Se utilizaron librerías para conexión con la plataforma de servicios en la nube Google Firebase. Para la conexión en general se utilizó la librería “firebase\_core”. Para la funcionalidad de autenticación con correo y contraseña se usó “firebase\_auth”; para los OAuth con Facebook y Google se utilizaron: “flutter\_facebook\_auth” y “google\_sign\_in”; para habilitar la posibilidad de redireccionamiento a url se utilizó “url\_launcher”; para compartir contenido en otras aplicaciones disponibles dentro del dispositivo se utilizó “share\_plus”; para habilitar la recepción de mensajes de notificaciones push se utilizó “firebase\_messaging” y para la conexión con la base de datos de firebase se utilizó “cloud\_firestore”. Para la aplicación se designó una versión mínima de SDK 21 para Android, lo que vendría siendo equivalente a Android 5 y para iOS la versión es 1.0.0.

## Versiones

* Flutter: 3.10.3
* Dart:3.0.3
* Android Studio: 2022.1.1.19
* Versión mínima de SDK de Android: 21
* Versión de iOS: 1.0.0
* Firebase\_core: 2.7.0
* Firebase\_auth: 4.2.9
* Flutter\_facebook\_auth: 5.0.8
* Google\_sign\_in: 6.0.2
* Url\_launcher: 6.0.9
* Share\_plus: 7.0.2
* Firebase\_messaging: 14.6.4
* Cloud\_firestore: 4.8.2

# Estructura de la aplicación

La aplicación fue estructurada por paquetes para una mayor organización e identificación de las responsabilidades de los archivos que están contenidos en cada uno de ellos. De manera general los paquetes del proyecto Flutter son los siguientes:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Todo el código fuente de la aplicación se encuentra contenido dentro de la carpeta “lib”. Dentro se encuentran los paquetes que se pueden ver en la imagen.

Texto

Descripción generada automáticamente

## Paquete “controllers”

Dentro de este paquete se encuentran los archivos que control de ciertos procesos que se ejecutan al momento de iniciar la aplicación o que requieren conexión con servicios externos. Estos procesos realizan lecturas de archivos contenidos dentro de la carpeta “assets” que es la carpeta que contiene únicamente los archivos de lectura. Los archivos dentro de esta carpeta no pueden ser sobrescritos en tiempo de ejecución.

Texto

Descripción generada automáticamente con confianza media

**Login\_controller.dart:** Contiene métodos de validación de mensajes que son recibidos por la implementación de los métodos de autenticación de Firebase cuya lógica se encuentra en la clase auth.dart.

**News\_controller.dart:** Se encarga de leer el archivo “assets/news\_files/news.json” para preparar los cuadros de las noticias que serán mostradas en la página principal de la aplicación.

**Quiz\_controller.dart:** Contiene toda la lógica para la lectura del archivo json “assets/question\_files/preguntas.json”,y el archivo “progreso.json” que se resguarda en un espacio de la memoria del dispositivo en donde si es posible la lectura y modificación de archivos; y se encarga igualmente de controlar la lógica del quiz, como cargar las preguntas de un determinado nivel, seleccionar una pregunta del nivel seleccionado, reestablecer el quiz y sus valores, así como, determinar el progreso que ha tenido el usuario de la aplicación.

## Paquete “my\_clases”

Dicho paquete contiene entidades necesarias para el funcionamiento del programa desde el aspecto lógico o del “back”.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Model:** La carpeta Model contiene a la clase user\_model.dart la cual posee atributos que caracterizan a los datos necesarios para resguardo en la base de datos, a su vez, posee métodos “setter y getters”. Facilita el tratamiento de los datos recibidos de la base de datos y para envío también.

**Auth.dart:** Es la clase que contiene los métodos de autenticación y registro de usuarios implementando las librerías de Firebase, de igual manera, resguarda la instancia del usuario autenticado. Utiliza una instancia de la interfaz con el repositorio de Firebase para la creación de un documento en caso de que se registre un nuevo usuario.

**News.dart:** Contiene una clase con atributos que caracterizan a los datos recibidos del archivo json “news.json” que son necesarios para la obtención de las imágenes, así como, los url de los artículos a los cuales se desea acceder.

**Notifications.dart:** Se encarga de realizar los procesos necesarios para habilitar la conexión del dispositivo con la plataforma Firebase para la recepción de las notificaciones.

**Progress.dart:** Posee atributos que son de utilidad para determinar el progreso del usuario en la aplicación. El objetivo de la clase es facilitar el tratamiento de los datos recuperados del archivo json donde se guarda el registro.

**Quiz\_question.dart:** Facilita el tratamiento de los datos necesarios para cargar la pregunta y sus opciones correctas en la interfaz del quiz.

## Paquete “my\_widgets”

En este paquete se encuentran los archivos que definen una clase de tipo Widget, los cuales son considerados como objetos con los cuales se puede construir la interfaz del usuario y permitir las interacciones.

Texto

Descripción generada automáticamente

**Congrats\_card\_widget.dart:** Posee la estructura y la lógica de la carta que se muestran al concluir el quiz.

**Menu\_button\_widget.dart:** Posee la estructura del menú despegable de la aplicación que suele colocarse en el AppBar del Scaffold de la aplicación.

**News\_card\_widget.dart:** Posee la estructura y la lógica de la carta que se muestra con las noticias o avisos en la página principal de la aplicación. Su información es obtenida haciendo uso de la clase news.dart.

**Question\_card\_widget.dart:**  Posee la estructura y la lógica de la carta que aparece en el quiz de la aplicación. Carga los datos de un objeto de la clase quiz\_question.dart.

**Quiz\_option\_widget.dart:** Es el botón de las opciones de respuesta a una pregunta del quiz. Contiene igualmente la lógica para tratar realizar el proceso en el caso de que se presione el botón.

## Paquete “pages”

En este paquete se encuentran los archivos donde se estructuran los Widgets de las páginas o vistas o los espacios donde se puede navegar dentro de la aplicación.

Texto

Descripción generada automáticamente

**Config\_page.dart:** Este archivo posee la estructura de la página en donde el usuario puede ver su información y modificarla una vez ha iniciado sesión. Es igualmente la página en donde se hace “logout”.

**Home\_page.dart:** Contiene la estructura de la página principal de la aplicación. Es donde se cargan las noticias y se presentan al usuario.

**Login\_register\_page.dart:** Es el archivo donde se presenta la estructura de la página de registro de usuario e inicio de sesión con el método correo/contraseña o utilizando OAuth para validar la autenticación con Facebook.

**Quiz\_lobby\_page.dart:**  En el archivo se estructura la vista donde se eligen los niveles del quiz. Carga el quiz y las preguntas.

**Quiz\_page.dart:** Se presenta la estructura de la actividad y de las preguntas y carga los gif representativos del desempeño del usuario.

## Repository

Contiene a la clase “user\_repository.dart” que se encarga de la conexión con el repositorio de Firebase donde recibe y envía los datos usando un UserModel.

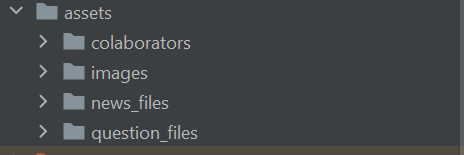
## Archivos fuera de paquetes

**Main.dart:** Contiene el método principal de la aplicación que es el que se ejecuta y permite el funcionamiento de la aplicación.

**Widget\_tree.dart**: Es un archivo que se carga desde el archivo main y carga la página principal de la aplicación en dependencia del estado del usuario. Se puede modificar para iniciar la aplicación en una página distinta en el caso de que haya un usuario autenticado.

# Assets

Dentro de esta carpeta del proyecto, es posible encontrar los archivos de los cuales se cargan ciertos datos necesarios para la aplicación.



Dentro de “colaborators” se encuentra el JSON con el nombre de los colaboradores estructurado de tal manera que permita la facilidad de lectura para una futura creación de una página en donde se presenten los nombres de estas personas como forma de agradecimiento. La estructura del JSON es la siguiente:

Texto

Descripción generada automáticamente

Dentro de “images” se encuentra la imagen del personaje del quiz al igual que los .gif que se despliegan dependiendo de la opción seleccionada en el quiz.

En la carpeta “news\_files” está el archivo JSON “news.json” del cual se carga la información de las noticias que se presenta en la página principal. La estructura de dicho JSON es:

Texto

Descripción generada automáticamente

Por último, dentro de “question\_files” se encuentra el archivo del cual se toman las preguntas dependiendo del nivel seleccionado. Si se selecciona el nivel 1, entonces se toman los datos del espacio 0 del arreglo. La estructura del JSON es la siguiente:

