1. Introducción

El sistema desarrollado es una aplicación Android de PreVentas de platos de comida, que permite registrar pedidos diarios, gestionar su estado (PEDIDO, ENTREGADO, CANCELADO) y generar reportes y estadísticas por rango de fechas.

La app está construida en Android Studio (API 24) y utiliza Firebase Realtime Database como backend NoSQL para almacenar platos, pedidos y datos básicos de administrador.

2. Estructura de la Base de Datos

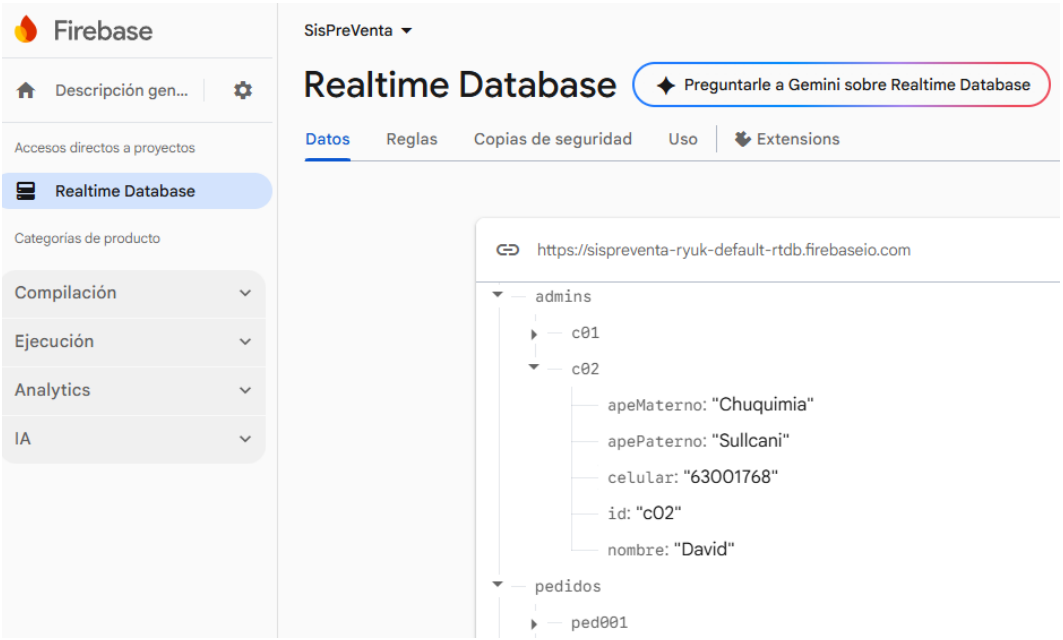


Figura 1. Pantalla consola de Firebase.

La base de datos en Firebase se estructura en tres nodos principales: admins, platos y pedidos, equivalentes a las tablas definidas en el modelo lógico DBML.

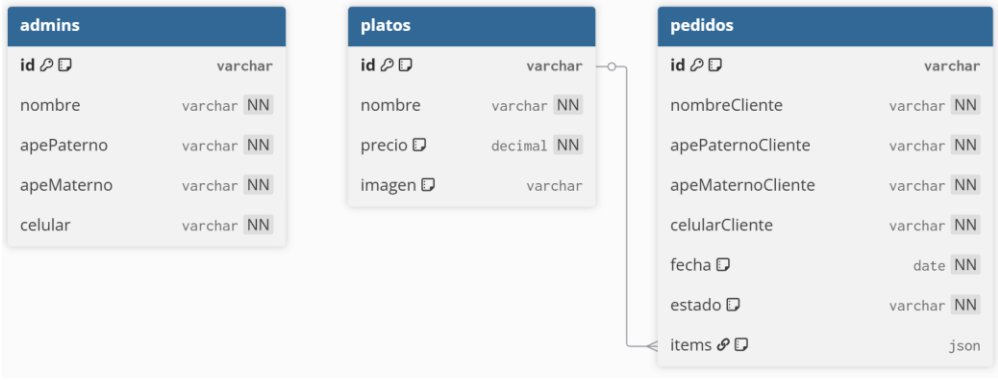


Figura 2. Estructura de base de datos usada para el programa.

2.1. Nodo admins

- id (string): ID del administrador.
- nombre, apePaterno, apeMaterno (string).
- celular (string).

Equivale a la tabla admins del DBML, representando al usuario que gestiona la app.

2.2. Nodo platos

- id (clave del nodo, ej. p01).
- nombre (string).
- precio (number).
- imagen (string, URL).

Se corresponde con la tabla platos, donde cada documento es un plato disponible para pedido.

2.3. Nodo pedidos

Cada pedido tiene la siguiente estructura:

- id (clave del nodo, ej. ped001).
- nombreCliente, apePaternoCliente, apeMaternoCliente, celularCliente (string).
- fecha (string en formato YYYY-MM-DD).
- estado (string: PEDIDO, ENTREGADO, CANCELADO).
- items (objeto JSON con idPlato:cantidad, por ejemplo "items": {"p01": 2, "p02": 1}).

En DBML esto se refleja como tabla pedidos con una columna items json y una relación lógica hacia platos mediante los IDs de plato contenidos en ítems.

2.4. Recursos de platos en GitHub

Además de Firebase, se utiliza un repositorio en GitHub como fuente de recursos estáticos para los platos.

- Las imágenes de cada plato (p01.jpeg, p02.jpeg, etc.) se almacenan en GitHub, y en Firebase solo se guarda la URL pública en el campo imagen del nodo platos.
- También se dispone de un archivo JSON en GitHub con nombres y precios de platos, que se usó como base para cargar y mantener sincronizado el catálogo de platos en Firebase durante el desarrollo.

En ejecución, la app lee desde Firebase nombre, precio e imagen, y carga las imágenes desde GitHub mediante la URL, separando claramente datos dinámicos (pedidos, estados) de recursos estáticos (catálogo de platos).

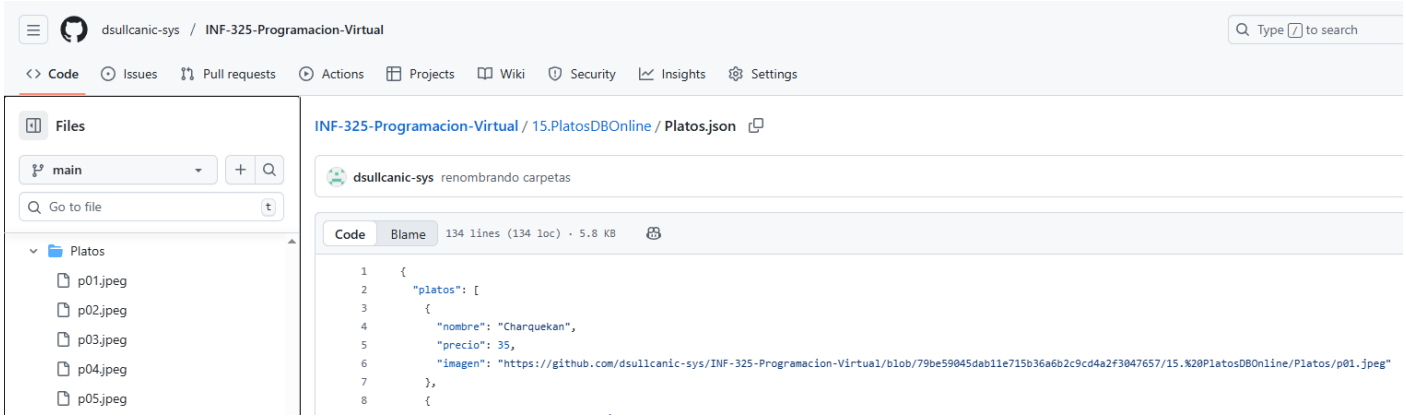
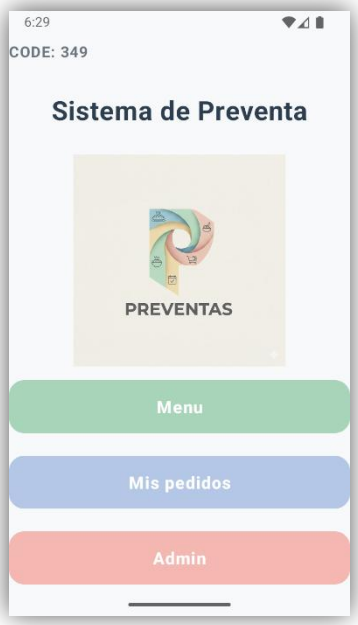


Figura 3. Pantalla repositorio de Github.

3. Descripción de la Aplicación

3.1 Pantalla de inicio



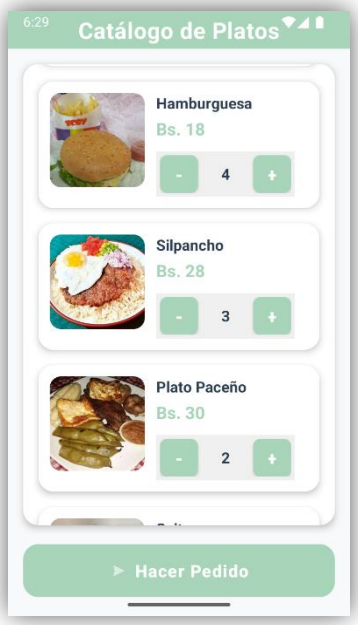
Al iniciar la aplicación, se muestra una pantalla de inicio que funciona como menú principal. En ella se presentan botones para acceder al menú de platos (toma de pedido), a la sección de pedidos (si se utiliza) y al módulo de administración, donde se encuentran los reportes y estadísticas. Esta pantalla simplifica la navegación, permitiendo que el administrador escoja rápidamente la tarea que desea realizar.

La pantalla de inicio actúa como menú principal y ofrece acceso directo a las funciones clave del sistema: realizar pedidos, ver pedidos y acceder al módulo de administración (reportes y estadísticas).

Se implementa como una Activity con botones dispuestos verticalmente.

Cada botón navega a la sección correspondiente (lista de platos, listado de pedidos, o pantalla de administración):

3.2. Pantalla de listado de platos / toma de pedido

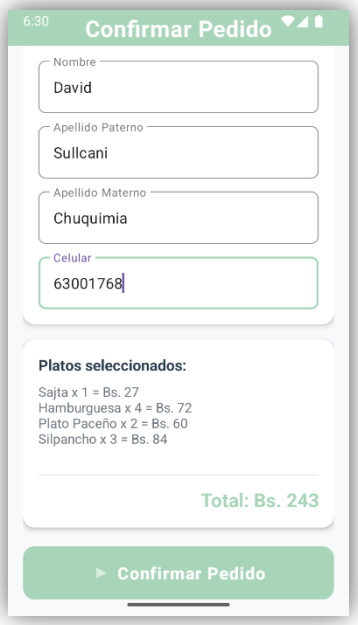


Cuando se accede al menú de platos, la aplicación consulta el nodo de "platos" en Firebase y muestra el catálogo en forma de lista: cada plato aparece con su imagen (cargada desde una URL de GitHub), su nombre y su precio. En cada tarjeta se disponen controles para aumentar o disminuir la cantidad del plato, de modo que el administrador puede configurar el pedido según la solicitud del cliente.

Esta pantalla muestra un listado de platos en un RecyclerView usando tarjetas (CardView), donde cada ítem incluye imagen, nombre, precio y controles para seleccionar la cantidad.

- Los botones + y − son ImageButton con íconos vectores ic_add/ic_remove del set de Material Icons, para un control de cantidad más limpio.
- El adaptador arma la estructura items (mapa de idPlato → cantidad) que luego se envía a Firebase al confirmar el pedido.

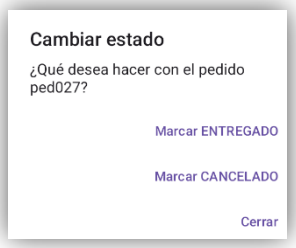
3.3. Pantalla de confirmación y registro de pedido



Cuando se accede al menú de platos, la aplicación consulta el nodo de "platos" en Firebase y muestra el catálogo en forma de lista: cada plato aparece con su imagen (cargada desde una URL de GitHub), su nombre y su precio. En cada tarjeta se disponen controles para aumentar o disminuir la cantidad del plato, de modo que el administrador puede configurar el pedido según la solicitud del cliente.

Una vez seleccionadas las cantidades, se pasa a una pantalla de confirmación donde se presenta un resumen del pedido (platos, cantidades y costos) y se solicitan los datos del cliente (nombre, apellidos y celular). Al confirmar, la aplicación genera un identificador de pedido, toma la fecha actual en formato año-mes-día y guarda en Firebase un nuevo registro con: datos del cliente, fecha, estado inicial "PEDIDO" y un listado de platos con sus cantidades.

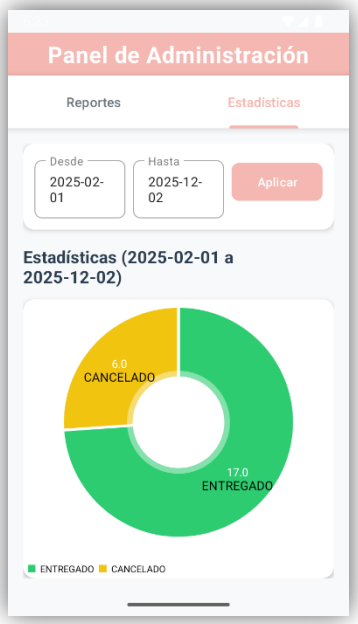
4. Gestión de pedidos (FragmentReportes)



Desde el módulo de administración, la sección de reportes permite filtrar y visualizar los pedidos registrados. En la parte superior se presentan dos campos para seleccionar una fecha "Desde" y una fecha "Hasta", que se rellenan mediante un cuadro de diálogo de calendario para evitar errores de formato. Al aplicar el filtro, la aplicación recorre todos los pedidos almacenados en Firebase y muestra únicamente aquellos cuya fecha se encuentra dentro del rango especificado.

Cada pedido se muestra como una tarjeta con información relevante (identificador, cliente, fecha y estado actual). Desde esta misma lista es posible cambiar el estado de un pedido: al seleccionar la acción correspondiente se muestra un cuadro de diálogo que permite marcar el pedido como ENTREGADO o CANCELADO. Cuando se elige una opción, la app actualiza el campo "estado" en la base de datos y refresca la lista, reflejando la nueva situación de ese pedido.

5. Estadísticas y comparativas



La pantalla de estadísticas se enfoca en ofrecer una vista global de la gestión de pedidos en un determinado período. Al igual que en reportes, se seleccionan una fecha inicial y una fecha final utilizando cuadros de diálogo de calendario; la aplicación normaliza internamente las fechas para considerar solo el día, evitando conflictos por diferencias de horas.

Con el rango definido, el sistema recorre los pedidos y cuenta cuántos están en estado ENTREGADO y cuántos en estado CANCELADO dentro de esas fechas. A partir de esos totales se genera un gráfico de torta donde cada porción representa la cantidad de pedidos de un estado. Esto permite observar de manera inmediata, por ejemplo, qué proporción de los pedidos de un mes llegaron a entregarse y qué proporción se canceló.

6. Tecnologías y aspectos técnicos

- Android Studio, lenguaje Java, con un mínimo de API 24 para asegurar compatibilidad con versiones recientes de Android.
- Firebase Realtime Database como backend NoSQL, con nodos admins, platos y pedidos para almacenar administradores, catálogo de platos y pedidos con su estado.
- GitHub como repositorio de recursos estáticos, donde se alojan las imágenes de los platos y un JSON inicial con nombres y precios, referenciados por URL desde el nodo platos en Firebase.
- Uso de RecyclerView y CardView para mostrar listas dinámicas y estilizadas de platos y pedidos.
- Uso de DatePickerDialog y SimpleDateFormat (formato yyyy-MM-dd) para selección y filtrado de pedidos por rango de fechas.
- Librería MPAndroidChart, utilizando el componente PieChart para generar gráficos de torta en la pantalla de estadísticas, comparando pedidos ENTREGADOS y CANCELADOS en el rango seleccionado.

7. Conclusiones

El sistema de PreVentas desarrollado cumple con los requisitos planteados: registra pedidos por fecha, permite actualizar su estado de PEDIDO a ENTREGADO o CANCELADO y genera reportes por rango de fechas, tanto en forma de lista como mediante gráficos de estadísticas.

La combinación de Firebase Realtime Database, recursos estáticos en GitHub y componentes estándar de Android (RecyclerView, CardView, DatePickerDialog) junto con MPAndroidChart permite una solución ligera, extensible y visualmente clara, que se puede ampliar en el futuro con funcionalidades adicionales como autenticación de administradores o nuevos tipos de reportes.