

## **Taller3: Integración con Firebase**

**Autor**

**Daniela Erazo Marin**

**Docente**

**David Steven Duran Vallejo**

**Unidad Central Del Valle**

**Facultad De Ingeniería**

**Ingeniería De Sistemas**

**Tuluá – Valle Del Cauca**

**2025**

**Repositorio:** [https://github.com/daniela-erazo-marin/curso\\_flutter](https://github.com/daniela-erazo-marin/curso_flutter)

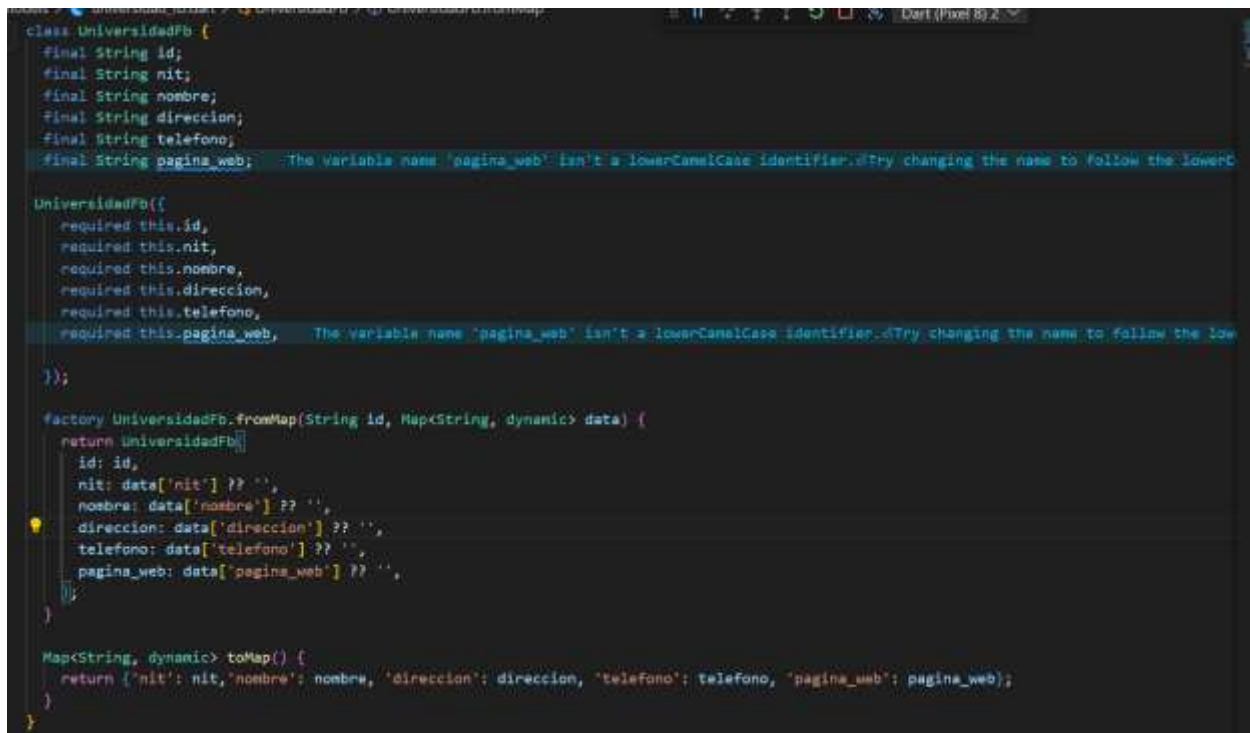
## Objetivo

**Desarrollar un módulo en Flutter** que integre Firebase y cree/gestione la colección universidades con los campos:

- nit (string)
- nombre (string)
- direccion (string)
- telefono (string)
- pagina\_web (string, URL)

El módulo debe **conectarse a Firebase**, realizar **operaciones CRUD** y mostrar una **vista de evidencia** con datos sincronizados (listar).

Se desarrolló el modelo universidad:



```
class UniversidadFb {
  final String id;
  final String nit;
  final String nombre;
  final String direccion;
  final String telefono;
  final String pagina_web;

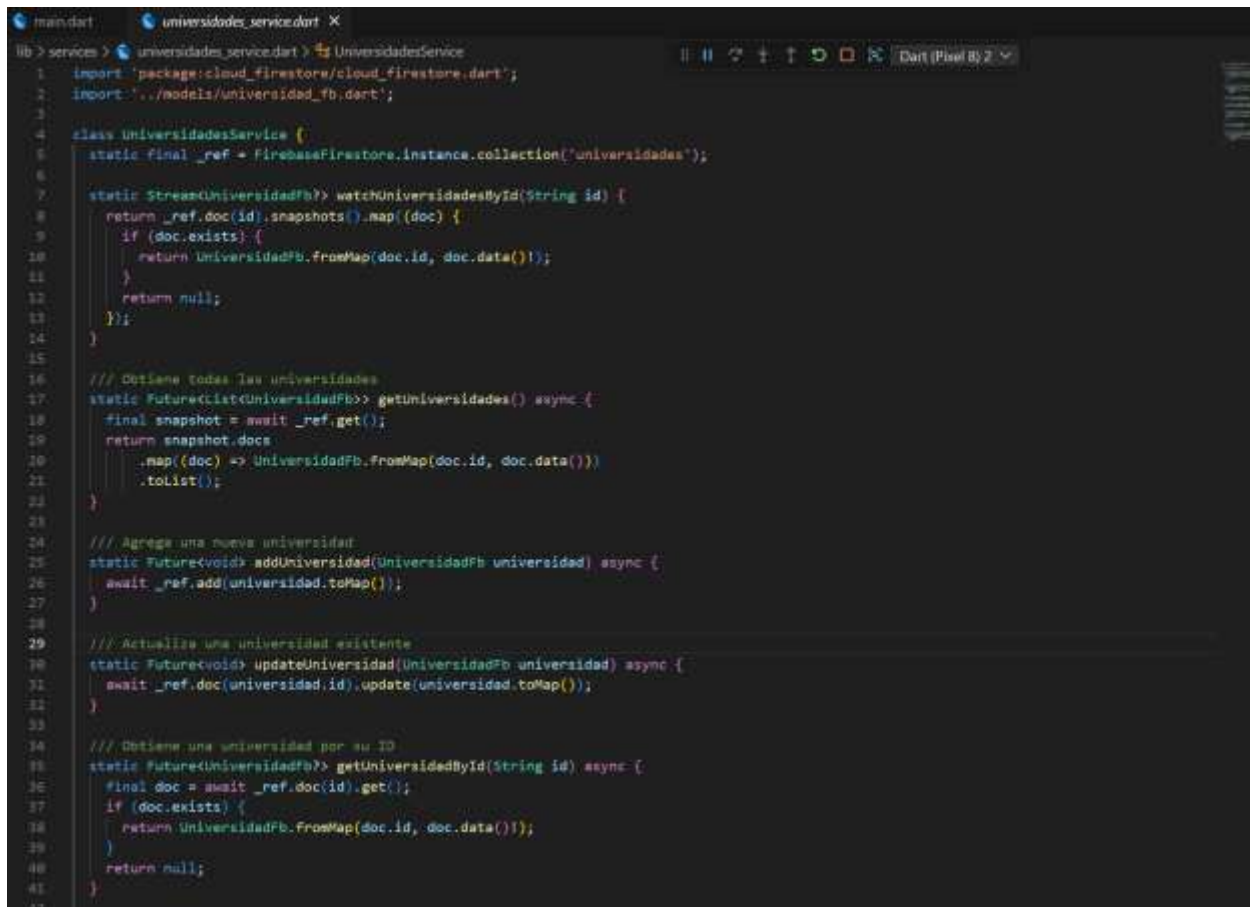
  UniversidadFb({
    required this.id,
    required this.nit,
    required this.nombre,
    required this.direccion,
    required this.telefono,
    required this.pagina_web,

  });

  factory UniversidadFb.fromMap(String id, Map<String, dynamic> data) {
    return UniversidadFb(
      id: id,
      nit: data['nit'] ?? '',
      nombre: data['nombre'] ?? '',
      direccion: data['direccion'] ?? '',
      telefono: data['telefono'] ?? '',
      pagina_web: data['pagina_web'] ?? '',
    );
  }

  Map<String, dynamic> toMap() {
    return {'nit': nit, 'nombre': nombre, 'direccion': direccion, 'telefono': telefono, 'pagina_web': pagina_web};
  }
}
```

Se desarrolló el servicio universidad:



```
lib > services > universidades_service.dart X
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import '../models/universidad_fb.dart';
3
4 class UniversidadesService {
5   static final _ref = FirebaseFirestore.instance.collection('universidades');
6
7   static Stream<UniversidadFb> watchUniversidadesById(String id) {
8     return _ref.doc(id).snapshots().map((doc) {
9       if (doc.exists) {
10         return UniversidadFb.fromMap(doc.id, doc.data());
11       }
12       return null;
13     });
14   }
15
16   /// Obtiene todas las universidades
17   static Future<List<UniversidadFb>> getUniversidades() async {
18     final snapshot = await _ref.get();
19     return snapshot.docs
20       .map((doc) => UniversidadFb.fromMap(doc.id, doc.data()))
21       .toList();
22   }
23
24   /// Agrega una nueva universidad
25   static Future<void> addUniversidad(UniversidadFb universidad) async {
26     await _ref.add(universidad.toMap());
27   }
28
29   /// Actualiza una universidad existente
30   static Future<void> updateUniversidad(UniversidadFb universidad) async {
31     await _ref.doc(universidad.id).update(universidad.toMap());
32   }
33
34   /// Obtiene una universidad por su ID
35   static Future<UniversidadFb> getUniverasidedById(String id) async {
36     final doc = await _ref.doc(id).get();
37     if (doc.exists) {
38       return UniversidadFb.fromMap(doc.id, doc.data());
39     }
40     return null;
41   }
42 }
```

## Alcance funcional

### 1. Conexión con Firebase.

```
universidad_fb_list_view.dart  firebase.json  app_router.dart  universidades_service.dart

firebase.json > ...
1  {"flutter":{"platforms":{"android":{"default":{"projectId":"universidades-5cc96","appId":"1:188483447759:android:d2882ccca98231a85994ba","%
```

```
EXPLORER  firebase_options.dart  app_router.dart  universidades

CURSO_FLUTTER
└─ lib
   └─ themes
   └─ views
      └─ auth
         └─ categoria_fb
            ├── categoria_fb_form_view.dart
            └─ categoria_fb_list_view.dart
      └─ cdt
         └─ cdt_list_view.dart
      └─ cido_vida
      └─ future
      └─ home
      └─ isolate
      └─ meal
      └─ paso_parametros
      └─ paso_parametros2
      └─ pokemons
      └─ timer
      └─ universidad_fb
         ├── universidad_fb_form_view.dart
         └─ universidad_fb_list_view.dart
      └─ widgets_demo
      └─ widgets
         ├── base_view.dart
         ├── custom_drawer.dart
         └─ firebase_options.dart
      └─ main.dart

lib > firebase_options.dart > ...
68
69  static const FirebaseOptions macos = FirebaseOptions(
70    apiKey: 'AIzaSyCAP6i4LM3iNEDMp8rI8gu5Zq3RbcplRU',
71    appId: '1:188483447759:ios:c838ad83f73d156f5994ba',
72    messagingSenderId: '188483447759',
73    projectId: 'universidades-5cc96',
74    storageBucket: 'universidades-5cc96.firebaseioapp',
75    iosBundleId: 'com.example.cursoFlutter',
76  );
77
78  static const FirebaseOptions windows = FirebaseOptions(
79    apiKey: 'AIzaSyA0JgMgZu17Q5FuYV79B6OA2776VHYbJg',
80    appId: '1:188483447759:web:34b158c8659382e15994ba',
81    messagingSenderId: '188483447759',
82    projectId: 'universidades-5cc96',
83    authDomain: 'universidades-5cc96.firebaseioapp.com',
84    storageBucket: 'universidades-5cc96.firebaseioapp',
85  );
86  }
87
```

## 2. Colección universidades en Firestore.

### Agregar un documento

Ruta superior  
/universidades

ID de documento ⓘ  
BoUdnfFEosYoEKbsZDA

Campo

Tipo

=

nit

string

⊖

String  
890.123.456-7

Campo

Tipo

=

nombre

string

⊖

String  
UCEVA

Campo

Tipo

=

direccion

string

⊖

Cancelar Guardar

universidades

Cloud Firestore

Base de datos

Protege tus recursos de Cloud Firestore contra los abusos, como fraudes de facturación o phishing.

Configura la Verificación de aplicaciones

Vista del panel

Compilador de consultas

universidades > BoUdnfFEosYoEKbsZDA

Más funciones en Google Cloud

(default)

universidades

BoUdnfFEosYoEKbsZDA

+ Iniciar colección

+ Agregar documento

+ Iniciar colección

categorias

BoUdnfFEosYoEKbsZDA

+ Agregar campo

universidades

BoUdnfFEosYoEKbsZDA

direccion: "Cra 27A 84B-144, Talus - Valle"

BoUdnfFEosYoEKbsZDA

nit: "890.123.456-7"

nombre: "UCEVA"

pagina\_web: "https://www.uceva.edu.co"

telefono: "+57 602 2242202"

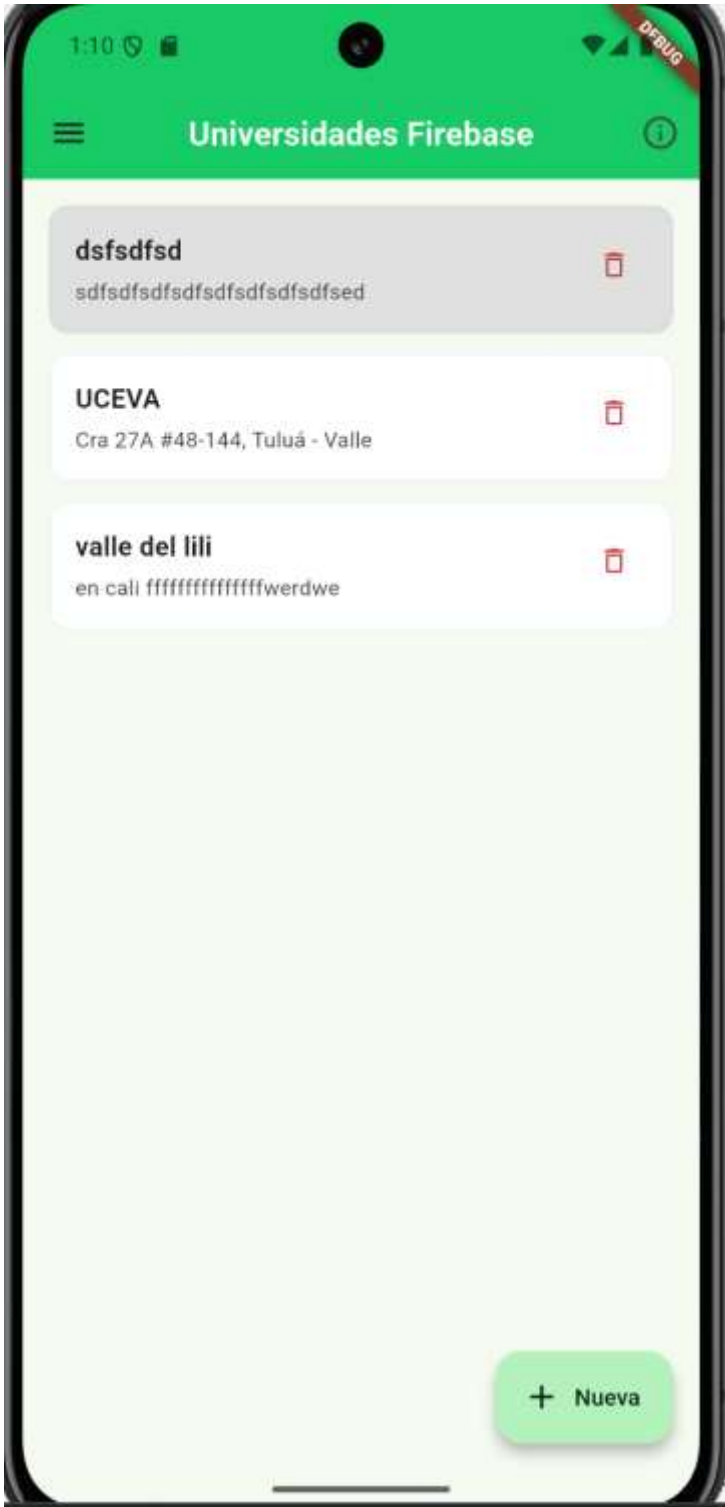
3. **Crear** (formulario) y **listar** universidades en tiempo real.

**Crear:**



The screenshot shows a mobile application interface for creating a university. At the top, there is a green header bar with a back arrow on the left and the title "Crear Universidad" in the center. Below the header, the main content area has a light green background. A white rounded rectangle contains the form, which is titled "Información básica" in a small, bold font. The form consists of five text input fields stacked vertically, each with a placeholder label: "Nit", "Nombre", "Direccion", "Telefono", and "Página web". At the bottom of the form, there are two buttons: a dark green button labeled "Guardar" and a white button with a green border labeled "Cancelar". The entire screen is framed by a black border, and a small portion of the status bar at the top shows the time "1:08" and some icons.

Listar:



4. **Validación** básica de campos (no vacíos y URL válida en pagina\_web).

```

16 class _UniversidadFbFormViewState extends State<UniversidadFbFormView> {
17   Widget _buildFormulario(BuildContext context, {required String? id}) {
18     textCapitalization: TextCapitalization.words,
19     validator: (value) {
20       if (value == null || value.trim().isEmpty) {
21         return 'El telefono es requerido';
22       }
23       if (value.trim().length < 15) {
24         return 'El telefono debe tener al menos 15 caracteres';
25       }
26       return null;
27     },
28   }, // TextFormField
29   const SizedBox(height: 15),
30   TextFormField(
31     controller: _pagina_webController,
32     decoration: InputDecoration(
33       labelText: 'Página web',
34       hintText: 'Ingresar la página web de la universidad',
35       border: OutlineInputBorder(
36         borderRadius: BorderRadius.circular(8),
37       ), // OutlineInputBorder
38     ), // InputDecoration
39     keyboardType: TextInputType.url,
40     textCapitalization: TextCapitalization.none,
41     validator: (value) {
42       if (value == null || value.trim().isEmpty) {
43         return 'La página web es requerida';
44       }
45       final url = value.trim();
46       // Verifica si es una URL válida
47       final uri = Uri.tryParse(url);
48       if (uri == null || !uri.hasScheme || !uri.hasAuthority) {
49         return 'Ingresar una URL válida (ejemplo: https://www.universidad.edu.co)';
50       }
51       return null;
52     },
53   ), // TextFormField

```



## Validaciones:

The screenshot shows a mobile application interface for creating a university. The title bar is green with a back arrow and the text 'Crear Universidad'. The status bar at the top shows the time 1:12 and a 'DEBUG' label. The form is titled 'Información básica' and contains five input fields. The first field, labeled 'Nit', contains the value '1234567890' and has a red border with the error message 'El nit debe tener al menos 11 caracteres' below it. The second field, labeled 'Nombre', contains 'Universidad del Valle'. The third field, labeled 'Direccion', contains 'Calle 13 #100-00, Sede Meléndez'. The fourth field, labeled 'Telefono', is empty and has a red border with the error message 'El telefono es requerido' below it. The fifth field, labeled 'Página web', is empty and has a red border with the error message 'La página web es requerida' below it. At the bottom, there are two buttons: 'Guardar' (green) and 'Cancelar' (white with a green border).

1:12

DEBUG

← Crear Universidad

Información básica

Nit

1234567890

El nit debe tener al menos 11 caracteres

Nombre

Universidad del Valle

Direccion

Calle 13 #100-00, Sede Meléndez

Telefono

El telefono es requerido

Página web

La página web es requerida

Guardar Cancelar

Validación de correo:

The screenshot shows a mobile application interface for creating a university profile. The title bar is green with a back arrow and the text "Crear Universidad". The status bar at the top shows the time 1:24, signal strength, and a "DEBUG" label. The form is titled "Información básica" and contains five input fields: "Nit" (890.123.456-7), "Nombre" (UCEVA), "Direccion" (Cra 27A #48-144, Tuluá - Valle), "Telefono" (+57 602 2242202), and "Página web" (www.uceva.edu.co). The "Página web" field has a red border and a red error message below it: "Ingresa una URL válida (ejemplo: https://www.univers...". At the bottom, there are two buttons: "Guardar" (green) and "Cancelar" (white with green border).

1:24

← Crear Universidad

Información básica

Nit  
890.123.456-7

Nombre  
UCEVA

Direccion  
Cra 27A #48-144, Tuluá - Valle

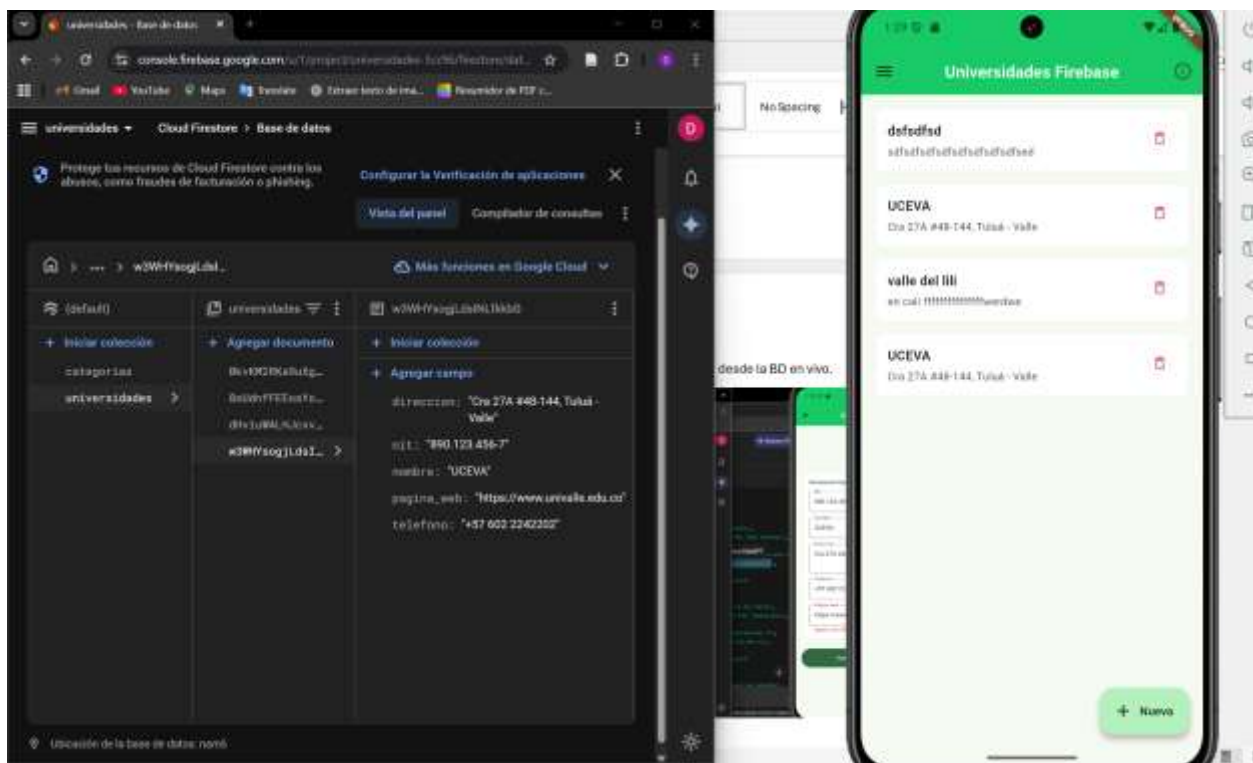
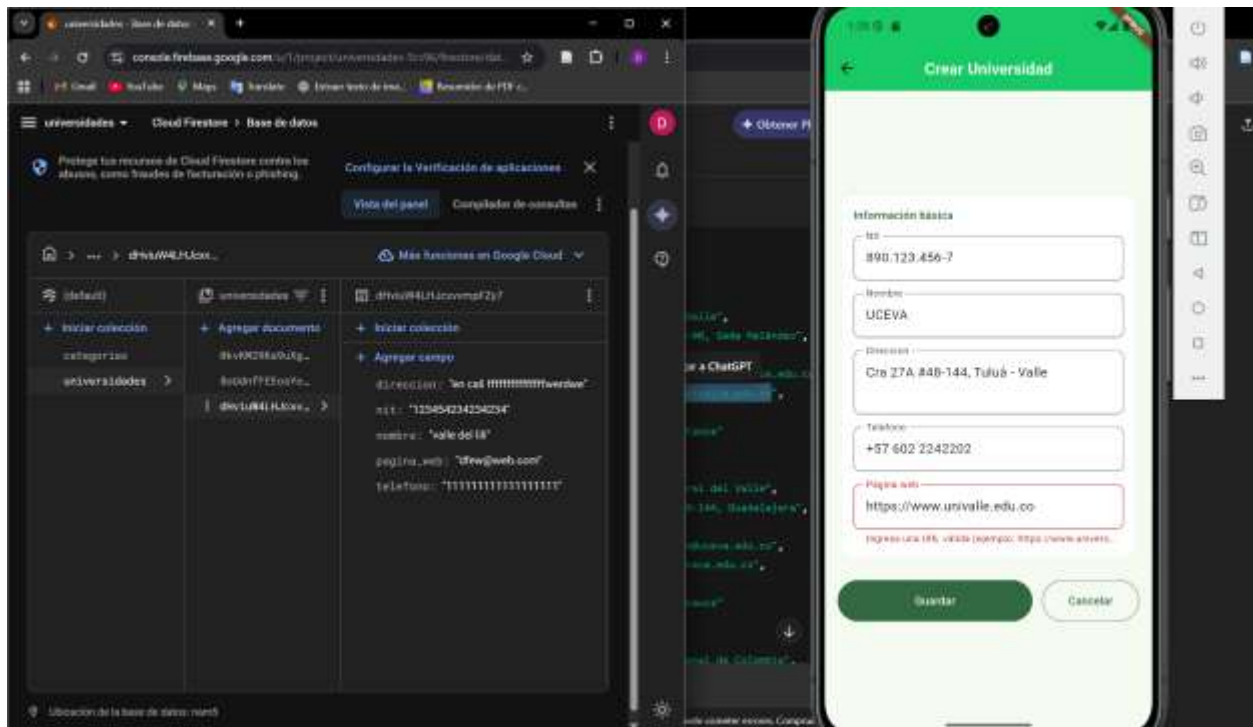
Telefono  
+57 602 2242202

Página web  
www.uceva.edu.co

Ingresa una URL válida (ejemplo: https://www.univers...)

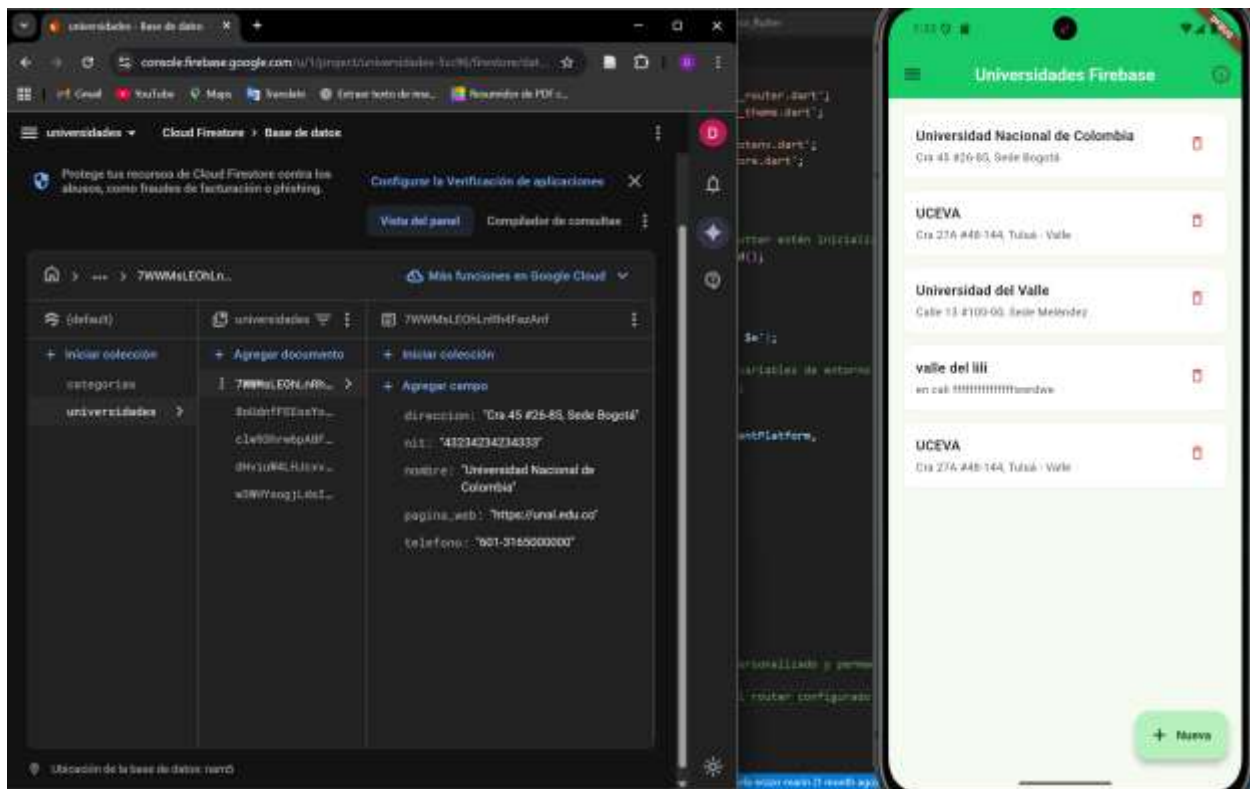
Guardar Cancelar

## 5. Vista de evidencia mostrando datos leídos desde la BD en vivo.



Eliminación:





## Descripción técnica del taller

Durante el desarrollo del taller se construyó una aplicación en Flutter orientada a la gestión de universidades, implementando una arquitectura modular basada en la separación por capas: modelo, servicios e interfaces gráficas.

### 1. Modelo de datos (Entidad Universidad)

Se diseñó la clase Universidad como modelo base, la cual contiene los atributos principales:

- `nit` (string)
- `nombre` (string)
- `direccion` (string)
- `telefono` (string)
- `pagina_web` (string, URL)

Esta estructura fue implementada en Dart y métodos `fromMap()` y `toMap()` para permitir su integración con Firebase.

### 2. Servicio de datos (UniversidadService)

Se creó un servicio encargado de la comunicación con la base de datos cloud **Firebase**, gestionando las operaciones **CRUD**:

- `getUniversidades()` → obtiene la lista de universidades.
- `addUniversidad()` → agrega un nuevo registro.
- `updateUniversidad()` → actualiza los datos existentes.
- `deleteUniversidad()` → elimina un registro.

El servicio utiliza la librería `cloud_firestore` para manejar la persistencia en la nube de manera asíncrona, garantizando integridad y respuesta en tiempo real.

### 3. Configuración de Firebase

El proyecto se integró con **Firebase** mediante la consola de desarrollador, registrando la aplicación y descargando los archivos de configuración `firebase_options.dart` y `firebase.json`. Se habilitó **Firestore Database** en modo de prueba y se crearon las colecciones necesarias (universidades).

La estructura de cada documento contiene los campos mencionados en el modelo, con sus tipos de datos definidos (String).

### 4. Interfaz gráfica (UI)

Se desarrolló una interfaz visual responsiva utilizando componentes de **Material Design**:

- Formulario de registro con `TextFormField` y validadores personalizados (por ejemplo, validación de formato para correos y URLs).
- Listado tipo grid implementado con `GridView` y `_buildGridContent()`, mostrando los datos de cada universidad con un diseño limpio, botones de acción (`IconButton`) y manejo de eventos (`onPressed`, `showDialog`, etc.).
- Uso de `ColorScheme` para adaptar los colores a los temas claro/oscuro del sistema.

## 5. **Validaciones técnicas**

Se implementaron validaciones en los campos del formulario para garantizar la integridad de los datos:

- El nombre y dirección no pueden estar vacíos.
- La página web se valida usando `Uri.tryParse()` para asegurar un formato correcto de URL.

## 6. **Pruebas con datos ficticios**

Finalmente, se cargaron registros de prueba en la colección de Firebase con información simulada de universidades colombianas (nit, nombre, dirección, teléfono y página web) para verificar el correcto funcionamiento del flujo completo de registro, lectura, actualización y eliminación.