

Proyecto Lista de contactos

Alejandro Ruiz Chilet

Índice

Ejercicio 1. Pantalla inicial (home).....	3
.....	4
Ejercicio 2. Pantalla de “Datos de Contacto”	5
Ejercicio 3. Crea la base de datos Firebase.....	8

Ejercicio 1. Pantalla inicial (home)

En este ejercicio se ha desarrollado la pantalla inicial de la aplicación, encargada de mostrar las categorías de contactos y la lista de contactos existentes.

La pantalla Home muestra tres categorías por defecto: Trabajo, Personal y Otros, junto con un desplegable que permite visualizar los contactos según la categoría seleccionada.

Debajo de las categorías se presenta la lista de contactos. Si no existen contactos, se muestra el mensaje “No hay contactos disponibles”. En caso contrario, cada contacto se muestra con su nombre, primer apellido y categoría, incluyendo dos botones de acción: Editar y Eliminar.

Fragmentos de código:

Categorías precargadas y filtro:

```
final List<String> categorias = ['Trabajo', 'Personal', 'Otros'];
String filtroCategoria = 'Todas';
```

Filtro de contactos por categoría:


```
List<Contacto> get contactosFiltrados {
  if (filtroCategoria == 'Todas') return contactos;
  return contactos.where((c) => c.categoria == filtroCategoria).toList();
}
```

Visualización de cada contacto con acciones:

```
return Card(
  child: ListTile(
    title: Text('$nombre $apellido1 - $categoria'),
    trailing: Row(
      mainAxisAlignment: MainAxisAlignment.min,
      children: [
        IconButton(
          icon: const Icon(Icons.edit),
          onPressed: () async {
            // editar: abre formulario con Contacto construido desde Firestore
            final c = Contacto(
              id: doc
                .id
                .hashCode,
              categoria: categoria,
              nombre: nombre,
              primerApellido: apellido1,
              segundoApellido:
                (data['segundoApellido'] ?? '').
                  toString(),
              telefono: (data['telefono'] ?? '').
                toString(),
              correo: (data['correo'] ?? '').toString(),
            ); // Contacto
```

Lista de Contactos

Categorías

 Trabajo




Contactos existentes

No hay contactos disponibles.

Lista de Contactos

Categorías

 Trabajo



Contactos existentes

Pepe García - Trabajo



Ejercicio 2. Pantalla de “Datos de Contacto”

Se ha desarrollado la pantalla “Datos de Contacto”, encargada de la creación y edición de contactos en la aplicación.

La pantalla contiene un formulario con los siguientes campos: categoría, nombre, apellidos, teléfonos y correo electrónico. Además, los campos obligatorios incluyen validaciones para asegurar la correcta introducción de los datos.

La pantalla puede funcionar en modo crear, para añadir contactos, y en modo editar, donde se cargan los datos automáticamente para modificarlos.

Al pulsar el botón guardar, se valida el formulario y, si los datos son correctos, el contacto se guarda y se regresa a la pantalla principal.

Fragmentos de código:

Formulario con clave y validación:


```
child: Form(
  key: _formKey,
  child: ListView(
    children: [
      TextFormField(
        controller: _nombreCtrl,
        decoration: const InputDecoration(labelText: 'Nombre'),
        validator: validarNombre,
      ), // TextFormField
      TextFormField(
        controller: _apellido1Ctrl,
        decoration: const InputDecoration(labelText: 'Primer Apellido'),
      ), // TextFormField
      TextFormField(
        controller: _apellido2Ctrl,
        decoration: const InputDecoration(
          labelText: 'Segundo Apellido',
        ), // InputDecoration
      ), // TextFormField
      TextFormField(
        controller: _tel1Ctrl,
        decoration: const InputDecoration(labelText: 'Teléfono'),
        keyboardType: TextInputType.phone,
        validator: validarTelefono,
      ), // TextFormField
      TextFormField(
        controller: _emailCtrl,
        decoration: const InputDecoration(
          labelText: 'Correo Electrónico',
        ), // InputDecoration
        keyboardType: TextInputType.emailAddress,
        validator: validarCorreo,
      ), // TextFormField
      const SizedBox(height: 16),
      DropdownButtonFormField<String>(
        value: categoria,
```

Campo de categoría con valor por defecto:

```
const Dropdown(height: 10),
DropdownButtonFormField<String>(  
  value: categoria,  
  decoration: const InputDecoration(labelText: 'Categoría'),  
  items: ['Trabajo', 'Personal', 'Otros']  
    .map(  
      (cat) => DropdownMenuItem(value: cat, child: Text(cat)),  
    )  
    .toList(),  
  onChanged: (value) => {  
    setState(() {  
      categoria = value!;  
    });  
  },  
) , // DropdownButtonFormField
```

Método para el botón guardar, con control de modos y redirección a la pantalla principal:

```
void guardarContacto() async {  
  if (!_formKey.currentState!.validate()) return;  
  
  final data = {  
    'nombre': _nombreCtrl.text.trim(),  
    'primerApellido': _apellido1Ctrl.text.trim(),  
    'segundoApellido': _apellido2Ctrl.text.trim(),  
    'telefono': _tel1Ctrl.text.trim(),  
    'correo': _emailCtrl.text.trim(),  
    'categoria': categoria,  
  };  
  
  if (esEdicion) {  
    await FirebaseFirestore.instance  
      .collection('contactos')  
      .doc(widget.contacto!.id.toString())  
      .set(data);  
  } else {  
    await FirebaseFirestore.instance.collection('contactos').add(data);  
  }  
  
  Navigator.pop(context);  
}
```

 Crear contacto

Nombre

Primer Apellido

Segundo Apellido


Teléfono

Correo Electrónico

Categoría

Otros ▼

Guardar

 Editar contacto

Nombre

Pepe

Primer Apellido

García

Segundo Apellido

Teléfono

666555444

Correo Electrónico

pepe@gmail.com

Categoría

Trabajo ▼

Guardar

Ejercicio 3. Crea la base de datos Firebase

Se ha creado y configurado una base de datos en Firebase (Cloud Firestore) para almacenar la información de los contactos de la aplicación.

Se ha creado un proyecto en Firebase y se ha habilitado Firestore Database. Los contactos se almacenan en una colección llamada contactos.

Para permitir el desarrollo y las pruebas, se ajustaron las reglas de seguridad de Firestore para permitir lectura/escritura durante el desarrollo. Una vez configurado Firebase en el proyecto Flutter, la aplicación puede guardar datos en Firestore y consultarlos desde la consola.

Pantalla principal del proyecto en Firebase:

The screenshot shows the Firebase console interface for a project named "listatareasfirebase". The left sidebar contains navigation options: "Descripción gen...", "Accesos directos a proyectos", "Firestore Database", "Categorías de producto", and a menu with "Compilación", "Ejecución", "Analytics", and "IA". The main content area displays the project name, a "Plan Spark" badge, and a list of apps: "4 apps", "contactos (android)", "contactos (ios)", and "contactos (windo...". Below this is the "Estadísticas" section, which includes an "Analytics" card showing "Usuarios activos por día" and "Retención del día 1" for the last 14 days. The analytics card indicates "No hay datos for the last 14 days". At the bottom, there is a table showing data for three apps: "contactos (...)", "contactos (...)", and "contactos (...)", with columns for DAU, Retención del día 1, and Ingresos.

App	DAU	Retención del día 1	Ingresos
contactos (...)	0	0%	\$0.00
contactos (...)	0	0%	\$0.00
contactos (...)	0	0%	\$0.00

La colección de contactos en Firestore Database:

The screenshot shows the Firebase console interface. On the left, the sidebar includes the Firebase logo, navigation links for 'Descripción gen...', 'Accesos directos a proyectos', 'Firestore Database', and 'Categorías de producto'. The main content area is titled 'listatareasfirebase' and shows 'Cloud Firestore >'. A search bar contains the text 'base de' and a suggestion to 'Preguntarle a Gemini sobre los conceptos básicos para usar Firestor'. Below this, there are tabs for 'Datos', 'Reglas', 'Índices', 'Recuperación ante desastres', 'Uso', and 'Extension'. The 'Datos' tab is active, showing a 'Vista del panel' button and a 'Compilador de consultas' button. The main area displays a collection named 'contactos' with a document 't0PD6e26Uk37mABTkNZh'. The document contains the following fields:

Field	Value
categoria	"Trabajo"
correo	"pepe@gmail.com"
nombre	"Pepe"
primerApellido	"García"
segundoApellido	""
telefono	"666555444"

Inicialización de Firebase en main.dart mediante el archivo firebase_options.dart generado por FlutterFire:

```
main.dart X
contactos > lib > main.dart > MyApp > MyApp
1 import 'package:flutter/material.dart';
2 import 'routes/rutas.dart';
3 import 'package:firebase_core/firebase_core.dart';
4 import 'firebase_options.dart';
5
Run | Debug | Profile
6 void main() async {
7   WidgetsFlutterBinding.ensureInitialized();
8   await Firebase.initializeApp(
9     options: DefaultFirebaseOptions.currentPlatform,
10  );
11  runApp(const MyApp());
12 }
```

Guardado de contactos en Firestore en la función guardarContacto():

```
void guardarContacto() async {
  if (!_formKey.currentState!.validate()) return;

  final data = {
    'nombre': _nombreCtrl.text.trim(),
    'primerApellido': _apellido1Ctrl.text.trim(),
    'segundoApellido': _apellido2Ctrl.text.trim(),
    'telefono': _tel1Ctrl.text.trim(),
    'correo': _emailCtrl.text.trim(),
    'categoria': categoria,
  };

  if (esEdicion) {
    await FirebaseFirestore.instance
      .collection('contactos')
      .doc(widget.contacto!.id.toString())
      .set(data);
  } else {
    await FirebaseFirestore.instance.collection('contactos').add(data);
  }

  Navigator.pop(context);
}
```