

**TECNOLÓGICO JOSÉ MARIO MOLINA
PASQUEL Y ENRRIQUEZ UNIDAD
ACADEMICA ZAPOTLANEJO**

MANUAL DE PROGRAMADOR

**DESARROLLO DE APLICACIONES PARA
DISPOSITIVOS MOVILES**

Evelyn Esmeralda Lomeli Venegas



EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

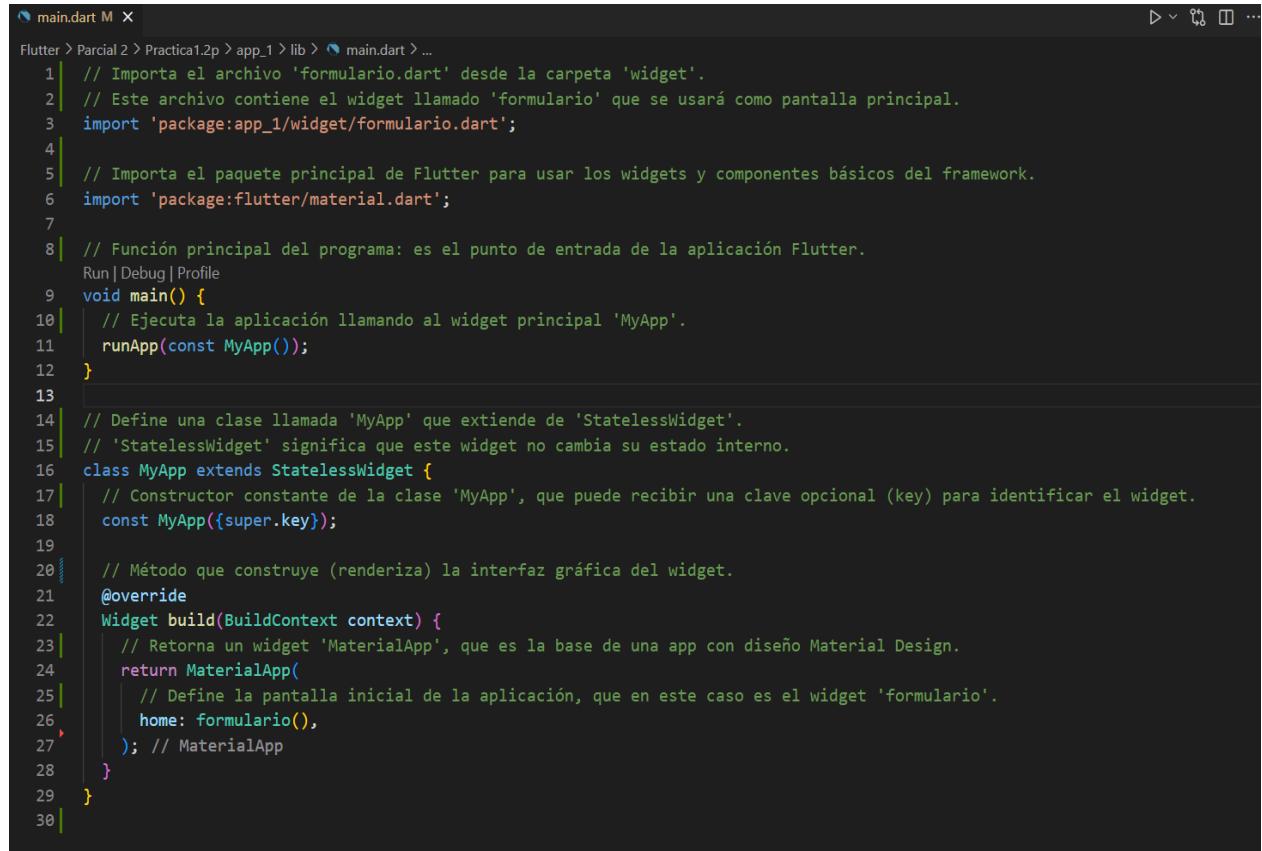
EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

Contenido

Programa 1 Parcial II	3
Programa 2 Parcial II	8
Programa 3 Parcial II	13
Practica 5 Parcial II	29

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

Programa 1 Parcial II



The screenshot shows a code editor window with the file 'main.dart' open. The code is written in Dart and defines the main application entry point. It imports the 'formulario.dart' widget from the 'widget' folder and the 'Material.dart' library. The 'main()' function runs the application using the 'runApp' method with a 'MyApp' widget. The 'MyApp' class extends 'StatelessWidget' and overrides the 'build' method to return a 'MaterialApp' widget with a 'formulario' widget as its home screen.

```
Flutter > Parcial 2 > Practica1.2p > app_1 > lib > main.dart > ...
1 // Importa el archivo 'formulario.dart' desde la carpeta 'widget'.
2 // Este archivo contiene el widget llamado 'formulario' que se usará como pantalla principal.
3 import 'package:app_1/widget/formulario.dart';
4
5 // Importa el paquete principal de Flutter para usar los widgets y componentes básicos del framework.
6 import 'package:flutter/material.dart';
7
8 // Función principal del programa: es el punto de entrada de la aplicación Flutter.
Run | Debug | Profile
9 void main() {
10   // Ejecuta la aplicación llamando al widget principal 'MyApp'.
11   runApp(const MyApp());
12 }
13
14 // Define una clase llamada 'MyApp' que extiende de ' StatelessWidget'.
15 // ' StatelessWidget' significa que este widget no cambia su estado interno.
16 class MyApp extends StatelessWidget {
17   // Constructor constante de la clase 'MyApp', que puede recibir una clave opcional (key) para identificar el widget.
18   const MyApp({super.key});
19
20   // Método que construye (renderiza) la interfaz gráfica del widget.
21   @override
22   Widget build(BuildContext context) {
23     // Retorna un widget 'MaterialApp', que es la base de una app con diseño Material Design.
24     return MaterialApp(
25       // Define la pantalla inicial de la aplicación, que en este caso es el widget 'formulario'.
26       home: formulario(),
27     ); // MaterialApp
28   }
29 }
30 }
```

Main.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
Practica1.2p > app_1 > lib > widget > formulario.dart > Disenio
1 // Importa el paquete principal de Flutter con los widgets y componentes de Material Design.
2 import 'package:flutter/material.dart';
3 // Importa funcionalidades para aplicar restricciones en la entrada de texto (inputFormatters).
4 import 'package:flutter/services.dart';
5
6 // Se declara un widget con estado llamado 'formulario'.
7 // Un StatefulWidget permite que su contenido cambie mientras la app está en ejecución.
8 class formulario extends StatefulWidget {
9     @override
10    State< StatefulWidget> createState() {
11        // Retorna la clase 'Disenio', que contendrá el diseño y la lógica del formulario.
12        return Disenio();
13    }
14 }
15
16 // Clase que define el diseño y el comportamiento del formulario.
17 class Disenio extends State< formulario > {
18     // Controladores que permiten leer y modificar el texto de los campos de texto.
19     final TextEditingController num1 = TextEditingController();
20     final TextEditingController num2 = TextEditingController();
21
22     // Clave global para identificar y manejar el estado del formulario.
23     final _formkey = GlobalKey< FormState >();
24
25     // Expresión regular que solo permite letras (mayúsculas/minúsculas) y espacios.
```

```
25     // Expresión regular que solo permite letras (mayúsculas/minúsculas) y espacios.
26     final RegExp _onlyLettersExp = RegExp(r'^[a-zA-Z\s]+');
27
28     // Función que valida el texto ingresado en los campos del formulario.
29     String? aplicarvalidacion(String? valor) {
30         // Si el campo está vacío, muestra un mensaje de error.
31         if (valor == null || valor.isEmpty) {
32             return 'Ingresa un dato';
33         }
34         // Si el texto NO coincide con la expresión regular (no son solo letras), muestra un error.
35         if (! _onlyLettersExp.hasMatch(valor)) {
36             return 'Ingresa solo letras';
37         }
38         // Si pasa las validaciones, retorna null (sin errores).
39         return null;
40     }
41
42     // Función que se ejecuta al presionar el botón "Validar".
43     void validardatos() {
44         // Comprueba si todas las validaciones del formulario fueron exitosas.
45         if (_formkey.currentState?.validate() ?? false) {
46             // Muestra un mensaje temporal indicando que los datos son correctos.
47             ScaffoldMessenger.of(context).showSnackBar(
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
48     |           SnackBar(content: Text('Datos correctos')),
49     |       );
50   } else {
51     // Si alguna validación falla, muestra un mensaje de error.
52     ScaffoldMessenger.of(context).showSnackBar(
53       |           SnackBar(content: Text('Error')),
54     );
55   }
56 }
57
58 // Método que se ejecuta cuando el widget deja de existir.
59 // Libera la memoria usada por los controladores de texto.
60 @override
61 void dispose() {
62   num1.dispose();
63   num2.dispose();
64   super.dispose();
65 }
66
67 // Método que construye la interfaz visual del formulario.
68 @override
69 Widget build(BuildContext context) {
```

```
69 Widget build(BuildContext context) {
70   return Scaffold(
71     // Crea la estructura base con barra superior y cuerpo.
72     appBar: AppBar(
73       title: Text('Formulario'), // Título que aparece en la parte superior.
74       backgroundColor: Colors.green, // Color de fondo del AppBar.
75     ), // AppBar
76     body: Padding(
77       // Agrega un margen interno alrededor del formulario.
78       padding: EdgeInsets.all(15),
79       // El widget Form agrupa campos de texto y permite su validación conjunta.
80       child: Form(
81         key: _formkey, // Asigna la clave del formulario para validaciones.
82         child: Column(
83           mainAxisSize: MainAxisSize.center, // Centra verticalmente los elementos.
84           children: [
85             // Primer campo de texto (nombre 1).
86             TextFormField(
87               controller: num1, // Controlador del campo.
88               // Solo permite letras y espacios mediante un filtro de entrada.
89               inputFormatters: [
90                 FilteringTextInputFormatter.allow(RegExp('[a-zA-Z\s]'))
91               ],
92             ),
93           ],
94         ),
95       ),
96     ),
97   );
98 }
```

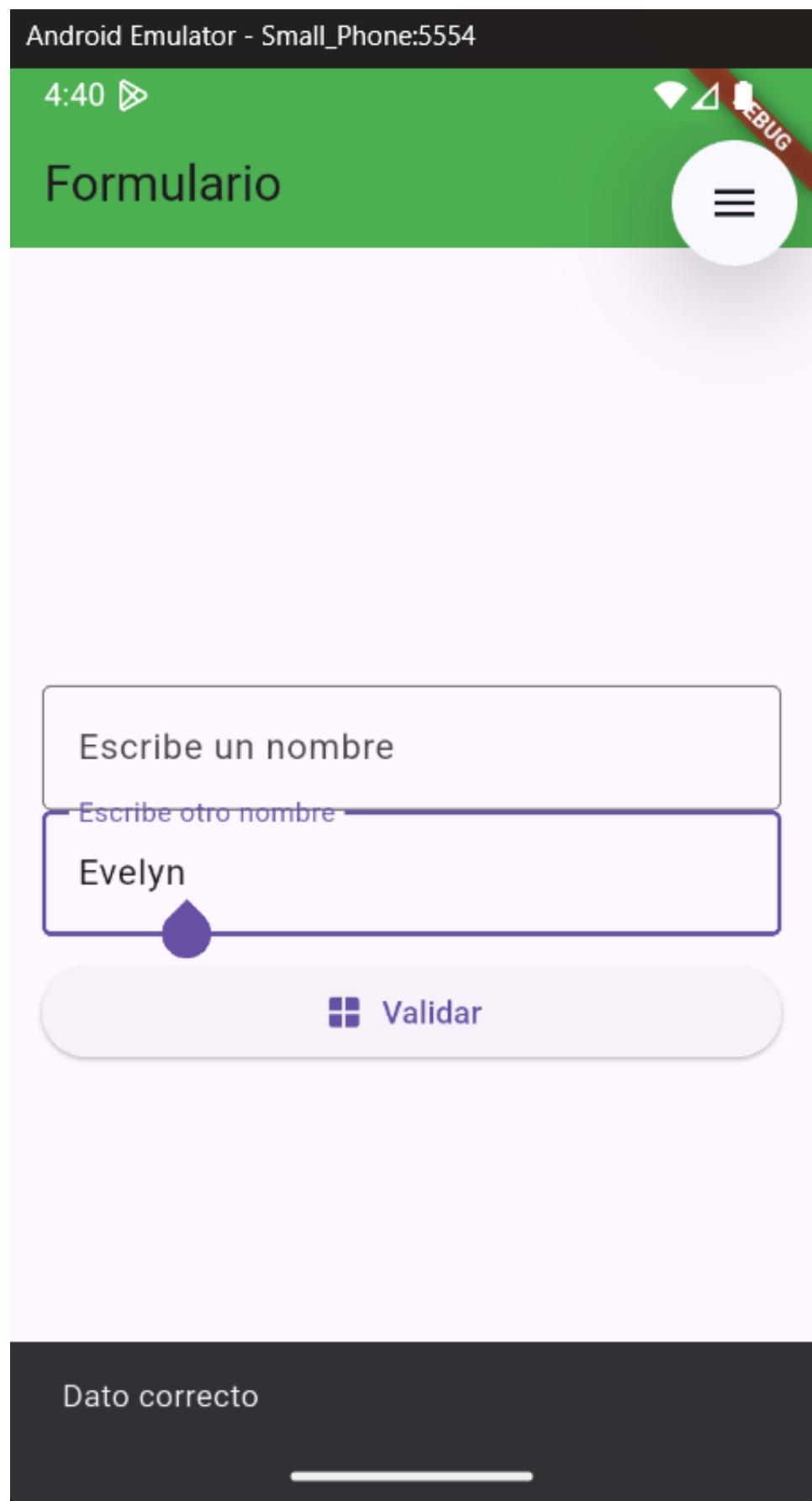
EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
92 |         keyboardType: TextInputType.text, // Tipo de teclado: texto.
93 |         decoration: InputDecoration(
94 |             labelText: 'Escribe un nombre', // Etiqueta del campo.
95 |             border: OutlineInputBorder(), // Borde alrededor del campo.
96 |         ), // InputDecoration
97 |         validator: aplicarvalidacion, // Aplica la validación definida arriba.
98 |     ), // TextFormField
99 |     SizedBox(height: 10), // Espacio entre los campos.
100 |
101    // Segundo campo de texto (nombre 2).
102    TextFormField(
103        controller: num2,
104        inputFormatters: [
105            FilteringTextInputFormatter.allow(RegExp(r'[a-zA-Z\s]'))
106        ],
107        keyboardType: TextInputType.text,
108        decoration: InputDecoration(
109            labelText: 'Escribe otro nombre',
110            border: OutlineInputBorder(),
111        ), // InputDecoration
112        validator: aplicarvalidacion, // Usa la misma validación.
113    ), // TextFormField
```

```
115 |     SizedBox(height: 10), // Espacio antes del botón.
116 |
117     // Botón que ocupa todo el ancho disponible.
118     SizedBox(
119         width: double.infinity,
120         child: ElevatedButton.icon(
121             onPressed: validardatos, // Ejecuta la función al presionarse.
122             label: Text('Validar'), // Texto del botón.
123             icon: Icon(Icons.check), // Ícono junto al texto.
124         ), // ElevatedButton.icon
125     ) // SizedBox
126     ],
127     ), // Column
128     ), // Form
129     ), // Padding
130 ); // Scaffold
131 }
132 }
133 }
```

Formulario. dart



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

Programa 2 Parcial II

```
app_2 > lib > main.dart > ...
1 // Importa el archivo Pagina1.dart que contiene el widget Pagina1
2 import 'package:app_2/widget/Pagina1.dart';
3
4 // Importa el paquete material.dart de Flutter que contiene los widgets Material Design
5 import 'package:flutter/material.dart';
6
7 // Función principal que es el punto de entrada de la aplicación
Run | Debug | Profile
8 void main() {
9     // Ejecuta la aplicación Flutter, iniciando con el widget MyApp
10    runApp(const MyApp());
11 }
12
13 // Define una clase MyApp que extiende StatelessWidget (widget sin estado)
14 class MyApp extends StatelessWidget {
15     // Constructor constante que recibe una key opcional
16     const MyApp({super.key});
17
18     // Método build obligatorio que describe cómo mostrar el widget
19     @override
20     Widget build(BuildContext context) {
21         // Retorna un MaterialApp, que es el widget raíz de una app con Material Design
22         return MaterialApp(
23             // Propiedad home define la pantalla principal de la aplicación
24             home: Pagina1() // Instancia el widget Pagina1 como pantalla inicial
25         ); // MaterialApp
26     }
27 }
28
```

Main.dart

```
1 // Importa los widgets personalizados desde diferentes archivos
2 import 'package:app_2/widget/Izquierdo.dart';
3 import 'package:app_2/widget/centro.dart';
4 import 'package:app_2/widget/derecho.dart';
5 // Importa el paquete material.dart de Flutter
6 import 'package:flutter/material.dart';
7
8 // Define Pagina1 como un StatefulWidget (widget con estado)
9 class Pagina1 extends StatefulWidget{
10     // Constructor constante que recibe una key opcional
11     const Pagina1({super.key});
12
13     // Método obligatorio que crea el estado del widget
14     @override
15     State< StatefulWidget> createState() {
16         // Retorna una instancia de la clase Disenio que manejará el estado
17         return Disenio();
18     }
19 }
20
21 // Clase Disenio que extiende State<Pagina1> para manejar el estado
22 class Disenio extends State< Pagina1 >{
23     // Variable de estado que controla el índice de navegación (0, 1, 2)
24     int _index = 0;
25 }
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
26 // Método build que construye la interfaz de usuario
27 @override
28 Widget build(BuildContext context) {
29     // Retorna un Scaffold (estructura básica de pantalla Material Design)
30     return Scaffold(
31         // Barra superior de la aplicación
32         appBar: AppBar(
33             // Título de la barra de aplicación
34             title: Text('ButtonNavigationBar'),
35             // Estilo del texto del título
36             style: TextStyle(
37                 color: const Color.fromARGB(255, 0, 0, 0)
38             ), // TextStyle
39         ), // Text
40         // Color de fondo de la AppBar
41         backgroundColor: Colors.lightGreenAccent), // AppBar
42
43         // Cuerpo principal de la pantalla
44         body: _index == 0 ? Izquierdo():(_index == 1 ? Centro(): Derecho()),
45
46         // Barra de navegación inferior
47         bottomNavigationBar: BottomNavigationBar(
48             // Color de fondo de la barra de navegación
49             backgroundColor: Colors.lightGreen,
50
51             // Array de items de navegación
52             items: [
53                 // Primer item de navegación - Izquierda
54                 BottomNavigationBarItem(icon:Icon(Icons.airlines),
55                 label: 'Izquierda',
56             ), // BottomNavigationBarItem
57                 // Segundo item de navegación - Centro
58                 BottomNavigationBarItem(icon:Icon(Icons.add_call),
59                 label: 'Centro',
60             ), // BottomNavigationBarItem
61                 // Tercer item de navegación - Derecha
62                 BottomNavigationBarItem(icon: Icon(Icons.zoom_out),
63                 label: 'Derecha') // BottomNavigationBarItem
64             ],
65             // Índice actual seleccionado
66             currentIndex: _index,
67             // Color del item seleccionado
68             selectedItemColor: Colors.deepPurple,
69             // Función que se ejecuta al tocar un item
70             onTap: (index){
71                 // Actualiza el estado con el nuevo índice
72                 setState(() {
73                     _index = index;
74                 });
75             },
76         ), // BottomNavigationBar
77     ); // Scaffold
78 }
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa el paquete widgets.dart de Flutter (contiene widgets básicos)
2 import 'package:flutter/widgets.dart';
3 // Define Izquierdo como un StatefulWidget (widget con estado)
4 class Izquierdo extends StatefulWidget{
5     // Constructor constante que recibe una key opcional
6     const Izquierdo({super.key});
7
8     // Método obligatorio que crea el estado del widget
9     @override
10    State< StatefulWidget> createState() {
11        // Retorna una instancia de la clase body que manejará el estado
12        return body();
13    }
14}
15 // Clase body que extiende State<Izquierdo> para manejar el estado
16 class body extends State<Izquierdo>{
17     // Método build que construye la interfaz de usuario
18     @override
19     Widget build(BuildContext context) {
20         // Retorna un widget Center que centra a su hijo
21         return Center(
22             // Widget hijo del Center - un Text con contenido 'Izquierdo'
23             child: Text('Izquierdo'),
24         ); // Center
25     }
}
```

Izquierdo.dart

```
1 // Importa el paquete widgets.dart de Flutter (contiene widgets básicos)
2 import 'package:flutter/widgets.dart';
3 // Define Derecho como un StatefulWidget (widget con estado)
4 class Derecho extends StatefulWidget{
5     // Constructor constante que recibe una key opcional
6     const Derecho({super.key});
7
8     // Método obligatorio que crea el estado del widget
9     @override
10    State< StatefulWidget> createState() {
11        // Retorna una instancia de la clase body que manejará el estado
12        return body();
13    }
14}
15 // Clase body que extiende State<Derecho> para manejar el estado
16 class body extends State<Derecho>{
17     // Método build que construye la interfaz de usuario
18     @override
19     Widget build(BuildContext context) {
20         // Retorna un widget Center que centra a su hijo en la pantalla
21         return Center(
22             // Widget hijo del Center - un Text con contenido 'Derecho'
23             child: Text('Derecho'),
24         ); // Center
25     }
}
```

Derecho.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa el paquete widgets.dart de Flutter (contiene widgets básicos)
2 import 'package:flutter/widgets.dart';
3 // Define Centro como un StatefulWidget (widget con estado)
4 class Centro extends StatefulWidget{
5     // Constructor constante que recibe una key opcional
6     const Centro({super.key});
7     // Método obligatorio que crea el estado del widget
8     @override
9     State< StatefulWidget> createState() {
10         // Retorna una instancia de la clase body que manejará el estado
11         return body();
12     }
13 }
14 // Clase body que extiende State<Centro> para manejar el estado
15 class body extends State<Centro>{
16     // Método build que construye la interfaz de usuario
17     @override
18     Widget build(BuildContext context) {
19         // Retorna un widget Center que centra a su hijo en la pantalla
20         return Center(
21             // Widget hijo del Center - un Text con contenido 'Centro'
22             child: Text('Centro'),
23         ); // Center
24     }
25 }
```

Centro.dart



EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

Programa 3 Parcial II

```
1 // Importa el archivo que contiene la definición de la pantalla Pagina1
2 import 'package:app_4/widget/pagina1.dart';
3 // Importa las utilidades principales de Flutter para construir la UI (widgets, temas, etc.)
4 import 'package:flutter/material.dart';
5 // Punto de entrada de la aplicación Flutter
6 Run | Debug | Profile
7 void main() {
8     // runApp inicia la aplicación y recibe el widget raíz
9     runApp(const MyApp());
10 }
11 // Definición de un widget sin estado (inmutable) llamado MyApp
12 class MyApp extends StatelessWidget {
13     // Constructor constante que permite optimizaciones en tiempo de compilación
14     const MyApp({super.key});
15     // Comentario: Este widget es la raíz de la aplicación.
16     @override
17     Widget build(BuildContext context) {
18         // MaterialApp es un widget que configura la app con Material Design
19         return MaterialApp(
20             // home define la pantalla inicial; aquí se instancia Pagina1()
21             home: Pagina1(),
22         ); // MaterialApp
23     }
24 }
```

Main.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
25 @override
26 Widget build(BuildContext context) {
27     return Scaffold( // Estructura básica de la página con AppBar, cuerpo y barra inferior
28         // Barra superior de la aplicación
29         appBar: AppBar(
30             // Título del AppBar
31             title: Text(
32                 'Operaciones Básicas',
33                 style: TextStyle(
34                     color: const Color.fromARGB(255, 0, 0, 0) // Color del texto del título
35                 ), // TextStyle
36             ), // Text
37             backgroundColor: const Color.fromARGB(255, 227, 142, 233), // Color de fondo del AppBar
38         ), // AppBar
39         // Cuerpo de la página, cambia según el valor de _index
40         body: _index == 0 ? Suma() : // Si _index es 0, muestra el widget Suma
41             _index == 1 ? Resta() : // Si _index es 1, muestra el widget Resta
42             _index == 2 ? Multiplicacion() : // Si _index es 2, muestra el widget Multiplicación
43             Division(), // Si _index es 3, muestra el widget División
44         // Barra de navegación inferior
45         bottomNavigationBar: BottomNavigationBar(
46             type: BottomNavigationBarType.fixed, // Tipo de barra fija (no se mueve al seleccionar)
47             backgroundColor: const Color.fromARGB(255, 227, 142, 233), // Color de fondo de la barra
48             items: [ // Lista de botones de la barra
49             ] // Detalles de los botones de la barra
50         )
51     );
52 }
```

```
1 // Importa el widget de la operación división desde la carpeta widget de tu proyecto
2 import 'package:app_4/widget/division.dart';
3 // Importa el widget de la operación multiplicación
4 import 'package:app_4/widget/multiplicacion.dart';
5 // Importa el widget de la operación resta
6 import 'package:app_4/widget/resta.dart';
7 // Importa el widget de la operación suma
8 import 'package:app_4/widget/suma.dart';
9 // Importa el paquete principal de Flutter para usar Material Design y widgets
10 import 'package:flutter/material.dart';
11 // Clase principal de la página, que puede cambiar de estado
12 class Pagina1 extends StatefulWidget{
13     // Constructor de la clase que permite usar 'super.key' para identificar el widget
14     const Pagina1({super.key});
15     // Método que crea el estado de este widget
16     @override
17     State<StatefulWidget> createState() {
18         return Disenio(); // Retorna la instancia de la clase de estado
19     }
20 }
21 // Clase que maneja el estado de Pagina1
22 class Disenio extends State<Pagina1>{
23     int _index = 0; // Variable privada para guardar la pestaña seleccionada (0=Suma, 1=Resta...)
24     // Método que construye la interfaz visual
25     @override
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
48     items: [ // Lista de botones de la barra
49       BottomNavigationBarItem(
50         icon: Icon(Icons.add), // Icono del botón
51         label: 'Suma', // Texto debajo del ícono
52       ), // BottomNavigationBarItem
53       BottomNavigationBarItem(
54         icon: Icon(Icons.remove),
55         label: 'Resta',
56       ), // BottomNavigationBarItem
57       BottomNavigationBarItem(
58         icon: Icon(Icons.close),
59         label: 'Multiplicación',
60       ), // BottomNavigationBarItem
61       BottomNavigationBarItem(
62         icon: Icon(Icons.percent),
63         label: 'División',
64       ), // BottomNavigationBarItem
65     ],
66     currentIndex: _index, // Indica cuál botón está seleccionado
67     selectedItemColor: const Color.fromRGBO(255, 22, 8, 24), // Color del botón seleccionado
68     // Función que se ejecuta al pulsar un botón
69     onTap: (index){
```

```
65   ],
66   currentIndex: _index, // Indica cuál botón está seleccionado
67   selectedItemColor: const Color.fromRGBO(255, 22, 8, 24), // Color del botón seleccionado
68   // Función que se ejecuta al pulsar un botón
69   onTap: (index){
70     setState(() { // setState actualiza la interfaz
71       _index = index; // Cambia el índice de la pestaña seleccionada
72     });
73   },
74 }, // BottomNavigationBar
75 ); // Scaffold
76 }
77 }
78 |
```

Pagina1.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
26 |     title: Text('Suma de dos números'),
27 |     backgroundColor: Colors.blueGrey,
28 |   ), // AppBar
29 |   body: Center(
30 |     // Center centra su contenido en la pantalla
31 |     child: Padding(
32 |       // Padding agrega espacio alrededor del contenido
33 |       padding: const EdgeInsets.all(15),
34 |       child: Card(
35 |         // Card: tarjeta visual con sombra y bordes redondeados
36 |         elevation: 18, // Profundidad de sombra
37 |         shape: RoundedRectangleBorder(
38 |           borderRadius: BorderRadius.circular(5), // Bordes redondeados
39 |         ), // RoundedRectangleBorder
40 |         child: SingleChildScrollView(
41 |           // Permite desplazamiento si el contenido sobrepasa el espacio disponible
42 |           child: Padding(
43 |             padding: const EdgeInsets.all(15.0), // Espaciado interno
44 |             child: Column(
45 |               // Coloca los widgets uno debajo del otro
46 |               mainAxisAlignment: MainAxisAlignment.min,
47 |               children: [
```

```
1 | // Importa los paquetes principales de Flutter necesarios para la interfaz
2 | import 'package:flutter/material.dart';
3 | // Importa utilidades para controlar la entrada de texto (por ejemplo, limitar a solo números)
4 | import 'package:flutter/services.dart';
5 | // Importa los widgets base de Flutter (opcional, pero suele venir con Material)
6 | import 'package:flutter/widgets.dart';
7 | // Se define una clase "Suma" que será un widget con estado (StatefulWidget)
8 | class Suma extends StatefulWidget {
9 |   const Suma({super.key}); // Constructor con clave opcional (para optimizaciones)
10 |   @override
11 |   State<StatefulWidget> createState() {
12 |     // Retorna una instancia del estado que controlará el comportamiento del widget
13 |     return body();
14 |   }
15 | }
16 | // Esta clase maneja el estado del widget "Suma"
17 | class body extends State<Suma> {
18 |   // Controladores para obtener el texto de los campos de entrada (TextFields)
19 |   final n1 = TextEditingController();
20 |   final n2 = TextEditingController();
21 |   @override
22 |   Widget build(BuildContext context) {
23 |     // Scaffold: estructura visual básica (AppBar, Body, etc.)
24 |     return Scaffold(
```

EVELYN ESMERALDA LOMELI VENEGAS

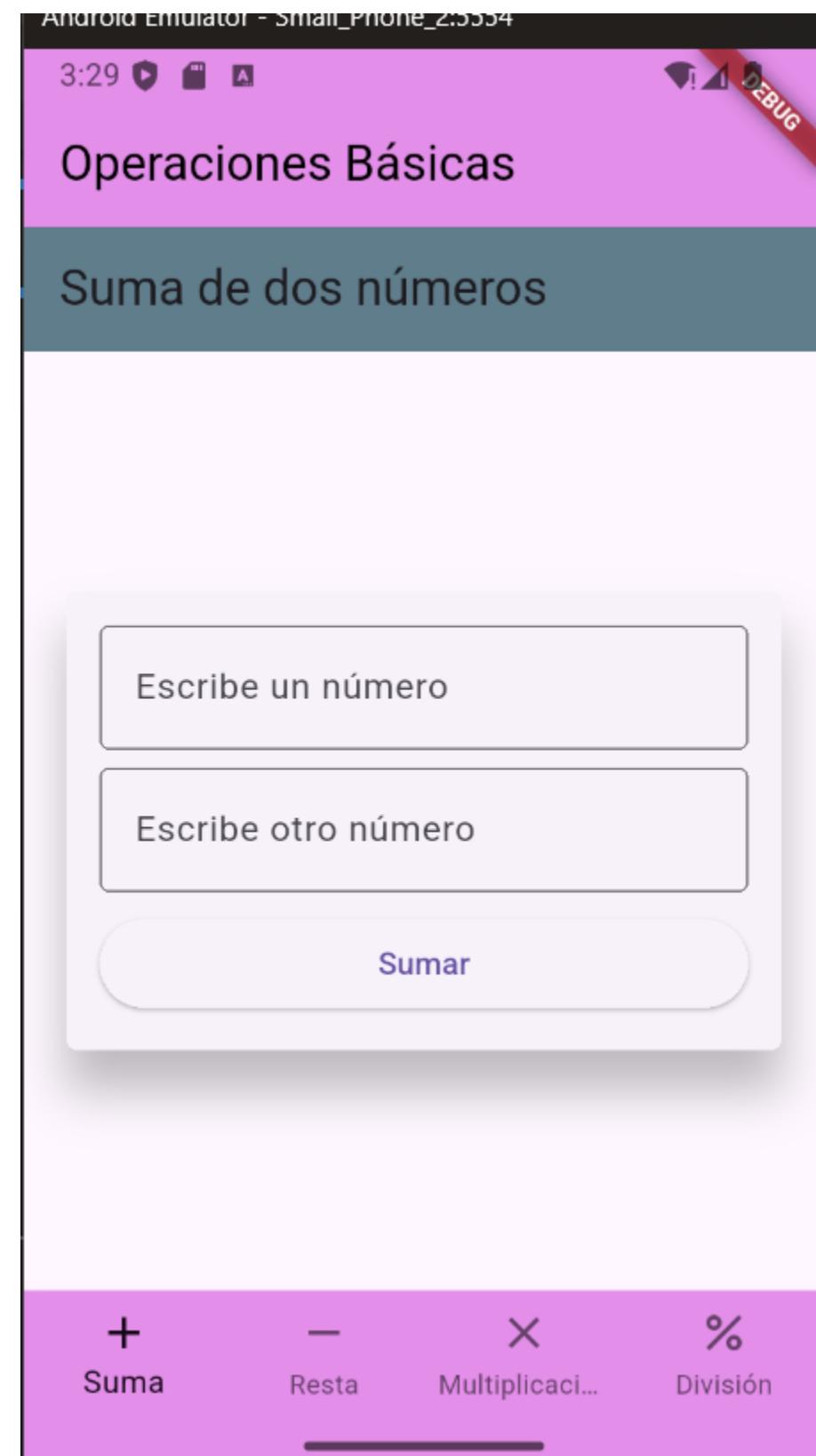
MANUAL DE PROGRAMADOR

```
SizedBox(
    width: double.infinity, // Ocupa todo el ancho disponible
), // SizedBox
// Primer campo de texto para el primer número
SizedBox(
    width: double.infinity,
    child: TextField(
        controller: n1, // Controlador del primer campo
        keyboardType: TextInputType.number, // Muestra teclado numérico
        inputFormatters: [
            // Permite solo números del 0 al 9
            FilteringTextInputFormatter.allow(RegExp(r'[0-9]'))
        ],
        decoration: InputDecoration(
            labelText: 'Escribe un número', // Texto de ayuda
            border: OutlineInputBorder(), // Borde visible
        ), // InputDecoration
    ), // TextField
), // SizedBox
SizedBox(height: 8), // Espacio vertical
// Segundo campo de texto para el segundo número
SizedBox(
    width: double.infinity,
```

```
91     ), // SizedBox
92     ],
93     ), // Column
94     ), // Padding
95     ), // SingleChildScrollView
96     ), // Card
97     ), // Padding
98     ), // Center
99     ); // Scaffold
100 }
101
102 // Función que realiza la suma
103 void sumar() {
104     // Aquí irá la lógica para sumar los valores de los dos campos
105     final num1 = int.tryParse(n1.text) ?? 0;
106     final num2 = int.tryParse(n2.text) ?? 0;
107     final resultado = num1 + num2;
108 }
109 }
110 |
```

Suma.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
app_4 > lib > widget > resta.dart > body
1 // Importa los paquetes necesarios para usar los widgets de Flutter
2 import 'package:flutter/material.dart';
3 import 'package:flutter/services.dart';
4 import 'package:flutter/widgets.dart';
5 // Clase principal del widget "Resta", que será con estado ( StatefulWidget )
6 class Resta extends StatefulWidget {
7   const Resta({super.key}); // Constructor con clave opcional
8   @override
9   State< StatefulWidget > createState() {
10    // Retorna una instancia del estado "body"
11    return body();
12  }
13}
14 // Clase que contiene el estado del widget (donde está la lógica y los datos)
15 class body extends State< Resta > {
16   // Controladores para los TextField, para obtener los valores escritos
17   final n1 = TextEditingController();
18   final n2 = TextEditingController();
19   @override
20   Widget build(BuildContext context) {
21     // Scaffold proporciona la estructura base (AppBar, Body, etc.)
22     return Scaffold(
23       appBar: AppBar(
24         title: Text('Resta de dos números'),
25         backgroundColor: Colors.blueGrey,
```

```
26     ),
27     body: Center(
28       // Centra el contenido en la pantalla
29       child: Padding(
30         padding: const EdgeInsets.all(15), // Espaciado exterior
31         child: Card(
32           elevation: 18, // Sombra de la tarjeta
33           shape: RoundedRectangleBorder(
34             borderRadius: BorderRadius.circular(5), // Bordes redondeados
35           ),
36           child: SingleChildScrollView(
37             // Permite desplazarse si el contenido es largo
38             child: Padding(
39               padding: const EdgeInsets.all(15.0), // Espaciado interno
40               child: Column(
41                 mainAxisSize: MainAxisSize.min,
42                 children: [
43                   // Primer campo de texto
44                   TextField(
45                     controller: n1, // Controlador del primer número
46                     keyboardType: TextInputType.number, // Teclado numérico
47                     inputFormatters: [
48                       FilteringTextInputFormatter.allow(RegExp(r'[0-9]')),
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
49 ],
50   decoration: InputDecoration(
51     labelText: 'Escribe un número',
52     border: OutlineInputBorder(),
53   ), // InputDecoration
54   ), // TextField
55   const SizedBox(height: 10), // Espacio entre campos
56
57   // Segundo campo de texto
58   TextField(
59     controller: n2, // Controlador del segundo número
60     keyboardType: TextInputType.number,
61     inputFormatters: [
62       FilteringTextInputFormatter.allow(RegExp(r'[0-9]')),
63     ],
64     decoration: InputDecoration(
65       labelText: 'Escribe otro número',
66       border: OutlineInputBorder(),
67     ), // InputDecoration
68   ), // TextField
69   const SizedBox(height: 15),
70
71   // Botón que ejecuta la función de resta
```

```
70   ),
71   ),
72   ),
73   ),
74   ),
75   ),
76   ),
77   ),
78   ),
79   ),
80   ),
81   ),
82   ),
83   ),
84   );
85 }
86
87 // Método que realiza la resta y muestra el resultado
88 void resta() {
89   // Convierte los textos a enteros
90   final num1 = int.tryParse(n1.text) ?? 0;
91   final num2 = int.tryParse(n2.text) ?? 0;
```

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

```
88 // Método que realiza la resta y muestra el resultado
89 void resta() {
90     // Convierte los textos a enteros
91     final num1 = int.tryParse(n1.text) ?? 0;
92     final num2 = int.tryParse(n2.text) ?? 0;
93     // Realiza la resta
94     final resultado = num1 - num2;
95     // Muestra el resultado en un AlertDialog
96     showDialog(
97         context: context,
98         builder: (BuildContext context) {
99             return AlertDialog(
100                 title: const Text("Resultado de la Resta"),
101                 content: Text("El resultado de $num1 - $num2 es: $resultado"),
102                 actions: [
103                     TextButton(
104                         onPressed: () => Navigator.of(context).pop(), // Cierra el diálogo
105                         child: const Text("Cerrar"),
106                     ), // TextButton
107                 ],
108             ); // AlertDialog
109         },
110     );
111 }
```

Resta.dart



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa la librería principal de Flutter para usar Material Design y widgets
2 import 'package:flutter/material.dart';
3 // Importa servicios de Flutter, necesario para usar inputFormatters
4 import 'package:flutter/services.dart';
5 // Importa widgets básicos de Flutter (opcional aquí porque ya está incluido en material.dart)
6 import 'package:flutter/widgets.dart';
7 // Widget de estado que representa la pantalla de multiplicación
8 class Multiplicacion extends StatefulWidget{
9     // Constructor de la clase, permite usar la clave del widget
10    const Multiplicacion({super.key});
11    // Método que crea el estado del widget
12    @override
13    State<StatefulWidget> createState() {
14        return body(); // Retorna la instancia de la clase de estado
15    }
16    // Función vacía (no hace nada aún, se puede eliminar o implementar después)
17    void multiplicacion(){
18    }
19    // Clase que maneja el estado de Multiplicacion
20    class body extends State<Multiplicacion>{
21        @override
22        Widget build(BuildContext context) {
23            return Scaffold( // Estructura básica de la pantalla
24                appBar: AppBar(
25                    title: Text('Multiplicación de dos números'),
```

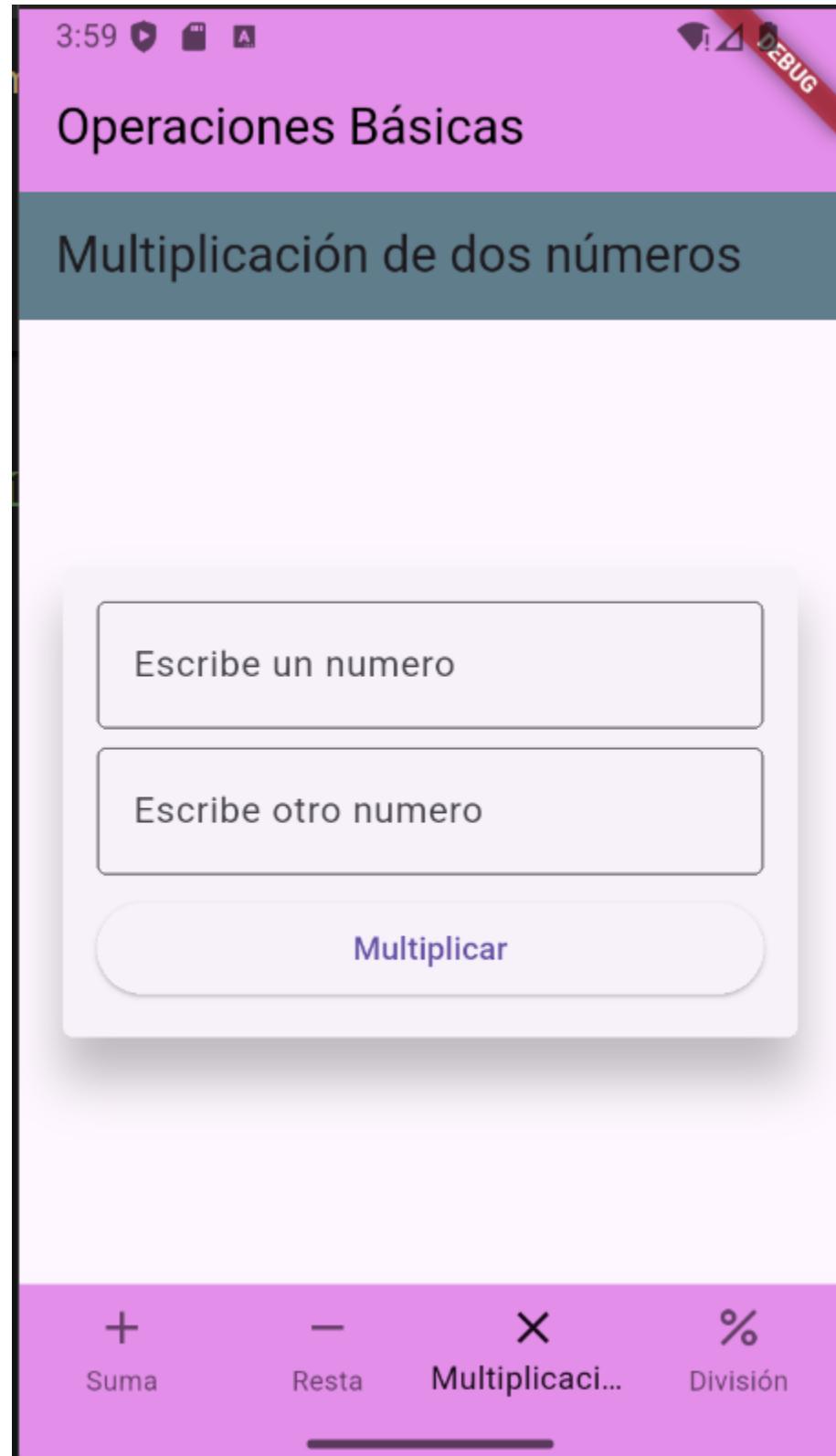
```
26   backgroundColor: Colors.blueGrey,
27 ),
28 body: Center( // Centra el contenido en la pantalla
29   child: Padding(
30     padding: EdgeInsetsGeometry.all(15), // Agrega un espacio de 15 pixeles a todos lados (nota: Edi
31     child: Card( // Tarjeta con sombra y borde redondeado
32       elevation: 18, // Sombra de la tarjeta
33       shape: RoundedRectangleBorder(
34         borderRadius: BorderRadiusGeometry.circular(5) // Borde redondeado de 5 pixeles (nota: Borde
35       ), // RoundedRectangleBorder
36       child: SingleChildScrollView( // Permite que el contenido sea desplazable si excede la pantalla
37         child: Padding(
38           padding: const EdgeInsets.all(15.0), // Padding interno de 15 pixeles
39           child: Column( // Organiza los elementos en vertical
40             mainAxisAlignment: MainAxisAlignment.min, // Ocupa solo el espacio necesario
41             children: [
42               SizedBox(
43                 width: double.infinity, // Ocupa todo el ancho disponible (en este caso no hace nada
44               ), // SizedBox
45               SizedBox(
46                 width: double.infinity, // El TextField ocupa todo el ancho
47                 child: TextField(
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
48 |         keyboardType: TextInputType.number, // Teclado numérico
49 |         inputFormatters: [
50 |             FilteringTextInputFormatter.allow(RegExp(r'[0-9]')) // Solo permite números
51 |         ],
52 |         decoration: InputDecoration(
53 |             labelText: 'Escribe un numero', // Texto de etiqueta
54 |             border: OutlineInputBorder() // Borde del campo de texto
55 |         ), // InputDecoration
56 |         ), // TextField
57 |     ], // SizedBox
58 |     SizedBox(
59 |         height: 8, // Espacio vertical entre los campos
60 |     ), // SizedBox
61 |     SizedBox(
62 |         width: double.infinity,
63 |         child: TextField(
64 |             keyboardType: TextInputType.number,
65 |             inputFormatters: [
66 |                 FilteringTextInputFormatter.allow(RegExp(r'[0-9]'))
67 |             ],
68 |             decoration: InputDecoration(
69 |                 labelText: 'Escribe otro numero',
```

```
70 |                     decoration: InputDecoration(
71 |                         labelText: 'Escribe otro numero',
72 |                         border: OutlineInputBorder()
73 |                     ), // InputDecoration
74 |                     ), // TextField
75 |                 ), // SizedBox
76 |                 SizedBox(
77 |                     height: 8, // Espacio vertical antes del botón
78 |                 ), // SizedBox
79 |                 SizedBox(
80 |                     width: double.infinity, // Botón ocupa todo el ancho
81 |                     child: ElevatedButton(
82 |                         onPressed: () {}, // Acción vacía al presionar (aquí pondrías la lógica de multi
83 |                         child: Text('Multiplicar') // Texto del botón
84 |                     ), // ElevatedButton
85 |                 ) // SizedBox
86 |             ],
87 |         ), // Column
88 |     ), // Padding
89 |     ), // SingleChildScrollView
90 | ), // Card
91 | ), // Padding
92 | ) // Center
```



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa la librería principal de Flutter para usar Material Design y widgets
2 import 'package:flutter/material.dart';
3 // Importa servicios de Flutter, necesario para usar inputFormatters
4 import 'package:flutter/services.dart';
5 // Importa widgets básicos de Flutter (opcional aquí porque ya está incluido en material.dart)
6 import 'package:flutter/widgets.dart';
7 // Widget de estado que representa la pantalla de división
8 class Division extends StatefulWidget{
9     // Constructor del widget, permite usar la clave del widget
10    const Division({super.key});
11    // Método que crea el estado del widget
12    @override
13    State<StatefulWidget> createState() {
14        return body(); // Retorna la instancia de la clase de estado
15    }
16 }
17 // Función vacía (puede ser usada para lógica de división, actualmente no hace nada)
18 void division(){
19 }
20 // Clase que maneja el estado de Division
21 class body extends State<Division>{
22    @override
23    Widget build(BuildContext context) {
24        return Scaffold( // Estructura básica de la pantalla
25            appBar: AppBar(
```

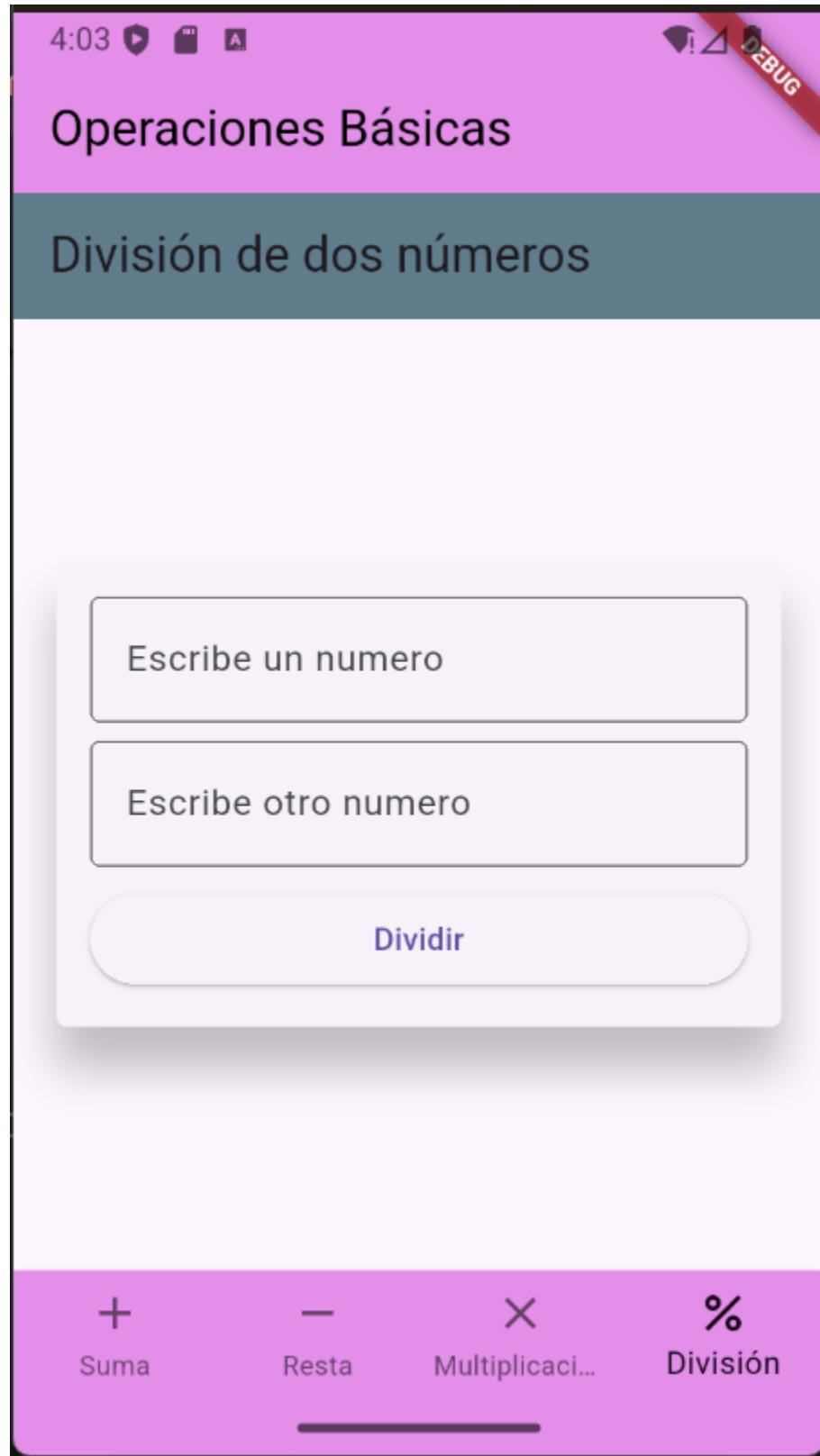
```
26 title: Text('División de dos números'),
27 backgroundColor: Colors.blueGrey,
28 ), // AppBar
29 body: Center( // Centra el contenido en la pantalla
30 child: Padding(
31 padding: EdgeInsetsGeometry.all(15), // Espacio de 15 píxeles a todos lados (nota: usar EdgeInsets)
32 child: Card( // Tarjeta con sombra y borde redondeado
33 elevation: 18, // Sombra de la tarjeta
34 shape: RoundedRectangleBorder(
35 borderRadius: BorderRadiusGeometry.circular(5) // Borde redondeado de 5 pixeles (nota: usar BorderRadius)
36 ), // RoundedRectangleBorder
37 child: SingleChildScrollView( // Permite desplazamiento si el contenido excede la pantalla
38 child: Padding(
39 padding: const EdgeInsets.all(15.0), // Padding interno de la tarjeta
40 child: Column( // Organiza los elementos en vertical
41 mainAxisAlignment: MainAxisAlignment.min, // Ocupa solo el espacio necesario
42 children: [
43 SizedBox(
44 width: double.infinity, // Ocupa todo el ancho (no tiene efecto visual aquí)
45 ), // SizedBox
46 SizedBox(
47 width: double.infinity, // Campo de texto ocupa todo el ancho
48 child: TextField(
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
49 |         keyboardType: TextInputType.number, // Teclado numérico
50 |         inputFormatters: [
51 |             FilteringTextInputFormatter.allow(RegExp(r'[0-9]')) // Solo permite ingresar números
52 |         ],
53 |         decoration: InputDecoration(
54 |             labelText: 'Escribe un numero', // Texto de la etiqueta
55 |             border: OutlineInputBorder() // Borde del campo
56 |         ), // InputDecoration
57 |         ), // TextField
58 |     ), // SizedBox
59 |     SizedBox(
60 |         height: 8, // Espacio vertical entre los campos
61 |     ), // SizedBox
62 |     SizedBox(
63 |         width: double.infinity, // Segundo campo de texto ocupa todo el ancho
64 |         child: TextField(
65 |             keyboardType: TextInputType.number,
66 |             inputFormatters: [
67 |                 FilteringTextInputFormatter.allow(RegExp(r'[0-9]')) // Solo números
68 |             ],
69 |             decoration: InputDecoration(
70 |                 labelText: 'Escribe otro numero', // Etiqueta del segundo campo
71 |                 border: OutlineInputBorder()
```

```
71 |             ),
72 |             ), // InputDecoration
73 |             ), // TextField
74 |         ), // SizedBox
75 |         SizedBox(
76 |             height: 8, // Espacio vertical antes del botón
77 |         ), // SizedBox
78 |         SizedBox(
79 |             width: double.infinity, // Botón ocupa todo el ancho
80 |             child: ElevatedButton(
81 |                 onPressed: () {}, // Acción vacía al presionar (aquí iría la lógica de división)
82 |                 child: Text('Dividir') // Texto del botón
83 |             ), // ElevatedButton
84 |         ), // SizedBox
85 |     ],
86 |     ), // Column
87 |     ), // Padding
88 |     ), // SingleChildScrollView
89 |     ), // Card
90 |     ), // Padding
91 |     ), // Center
92 | ); // Scaffold
93 | }
```



EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

Practica 5 Parcial II

```
app5 > lib > main.dart > ...
1 // Importa el widget "Maine" desde la carpeta widget de tu proyecto
2 import 'package:app5/widget/Maine.dart';
3 // Importa Flutter Material, necesario para usar widgets de Material Design
4 import 'package:flutter/material.dart';
5
6 // Función principal de la aplicación, punto de entrada
Run | Debug | Profile
7 void main() {
8   runApp(const MyApp()); // Ejecuta la aplicación y carga el widget MyApp
9 }
10
11 // Clase principal de la aplicación, es un widget sin estado
12 class MyApp extends StatelessWidget {
13   // Constructor del widget MyApp, permite usar la clave del widget
14   const MyApp({super.key});
15
16   // Método que construye la interfaz visual de este widget
17   @override
18   Widget build(BuildContext context) {
19     return MaterialApp( // Widget que configura la app como Material App
20       home: Maine(), // Página principal que se mostrará al iniciar la app
21     ); // MaterialApp
22   }
23 }
24
```

Main.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 | // Importa las páginas que se mostrarán desde la carpeta paginas
2 | import 'package:app5/paginas/datos.dart';
3 | import 'package:app5/paginas/home.dart';
4 | import 'package:app5/paginas/lista.dart';
5 | // Importa Flutter Material, necesario para usar widgets de Material Design
6 | import 'package:flutter/material.dart';
7 | // Widget de estado que representa la pantalla principal con Drawer
8 | class Maine extends StatefulWidget{
9 |   const Maine({super.key});
10 |   // Crea el estado del widget
11 |   @override
12 |   State<StatefulWidget> createState() {
13 |     return Base(); // Retorna la clase que maneja el estado
14 |   }
15 |   // Clase que maneja el estado de Maine
16 |   class Base extends State<Maine>{
17 |     int _index = 0; // Índice de la página actualmente seleccionada
18 |     // Lista de páginas que se mostrarán según el índice
19 |     final List<Widget> _paginas =[
20 |       Home(), // Página Home
21 |       Lista(), // Página Lista
22 |       Datos() // Página Datos
23 |     ];
24 |     // Función para cambiar el índice y actualizar la pantalla
25 |     void _ClickIndex(int i){
```

```
26 |       setState(() {
27 |         _index = i; // Cambia la página seleccionada
28 |       });
29 |     }
30 |     @override
31 |     Widget build(BuildContext context) {
32 |       return Scaffold(
33 |         // Barra superior
34 |         appBar: AppBar(
35 |           title: Text(
36 |             'Navigator Drawer',
37 |             style: TextStyle(
38 |               color: Colors.white // Color del texto
39 |             ), // TextStyle
40 |             ), // Text
41 |             backgroundColor: const Color.fromARGB(255, 123, 46, 154), // Color de fondo del AppBar
42 |           ), // AppBar
43 |           // Drawer lateral
44 |           drawer: Drawer(
45 |             child: ListView(
46 |               padding: EdgeInsets.all(5), // Espaciado interno
47 |               children: [
48 |                 // Encabezado del Drawer
49 |                 DrawerHeader(
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
49 |   decoration: BoxDecoration(
50 |     color: const Color.fromRGBO(255, 46, 4, 65) // Color de fondo del header
51 |   ), // BoxDecoration
52 |   child: Row(
53 |     children: [
54 |       CircleAvatar(
55 |         radius: 30, // Tamaño del avatar
56 |         backgroundImage: AssetImage('assets/homa.png'), // Imagen del avatar
57 |       ), // CircleAvatar
58 |       Padding(
59 |         padding: const EdgeInsets.only(left:15, right: 20), // Espaciado lateral
60 |         child: Column(
61 |           mainAxisAlignment: MainAxisAlignment.center,
62 |           children: [
63 |             Text(
64 |               'Aplicacion ',
65 |               style: TextStyle(
66 |                 color: Colors.white,
67 |                 fontSize: 20,
68 |                 fontFamily: "Verdana",
69 |                 fontStyle: FontStyle.italic
70 |               ), // TextStyle
71 |             ), // Text

```

```
72 |             Text(
73 |               'Desarrollada por',
74 |               style: TextStyle(
75 |                 color: Colors.white,
76 |               ), // TextStyle
77 |             ), // Text
78 |             Text(
79 |               'TSJ Zapotlanejo',
80 |               style: TextStyle(
81 |                 color: Colors.white,
82 |                 fontSize: 20,
83 |                 fontFamily: "Verdana",
84 |                 fontStyle: FontStyle.italic
85 |               ), // TextStyle
86 |             ), // Text
87 |           ],
88 |         ), // Column
89 |       ) // Padding
90 |     ],
91 |   ) // Row
92 | ), // DrawerHeader
93 | // Opción Home en el Drawer
94 | ListTile(
```

EVELYN ESMERALDA LOMELI VENEGAS

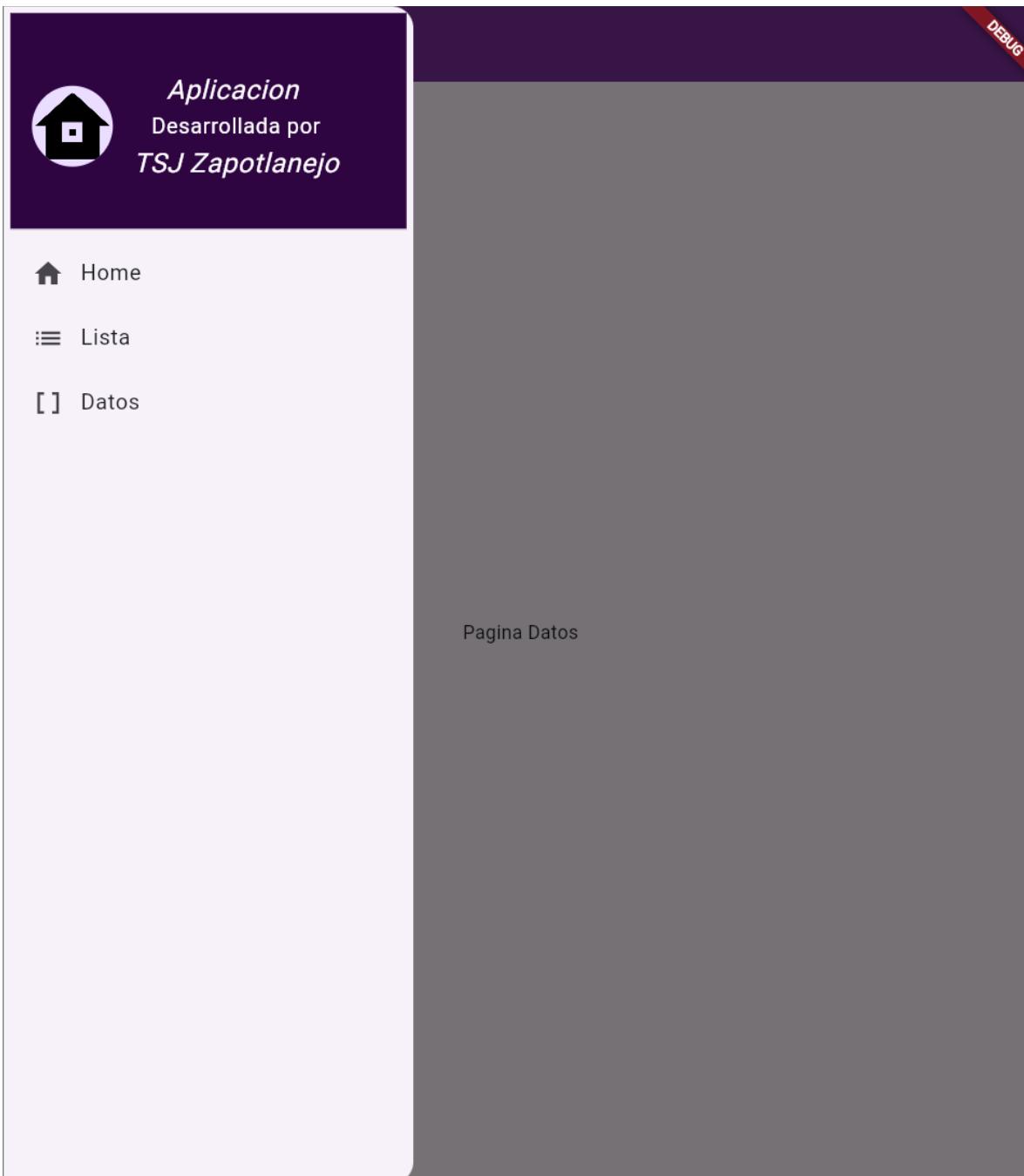
MANUAL DE PROGRAMADOR

```
95 |         leading: Icon(Icons.home), // Ícono de la opción
96 |         title: Text('Home'), // Texto de la opción
97 |         onTap: () { // Acción al pulsar
98 |             _ClickIndex(0); // Cambia a la página Home
99 |             Navigator.pop(context); // Cierra el Drawer
100|         },
101|     ), // ListTile
102|     // Opción Lista en el Drawer
103|     ListTile(
104|         leading: Icon(Icons.list),
105|         title: Text('Lista'),
106|         onTap: () {
107|             _ClickIndex(1); // Cambia a la página Lista
108|             Navigator.pop(context);
109|         },
110|     ), // ListTile
111|     // Opción Datos en el Drawer
112|     ListTile(
113|         leading: Icon(Icons.data_array),
114|         title: Text('Datos'),
115|         onTap: () {
116|             _ClickIndex(2); // Cambia a la página Datos
117|             Navigator.pop(context);
118|         },
119|     ) // ListTile
120| ],
121| ), // ListView
122| ), // Drawer
123| // Cuerpo de la página, se muestra según el índice seleccionado
124| body: _paginas[_index],
125| ); // Scaffold
126| }
127| }
128| }
```

```
116|         _ClickIndex(2); // Cambia a la página Datos
117|         Navigator.pop(context);
118|     },
119|     ) // ListTile
120| ],
121| ), // ListView
122| ), // Drawer
123| // Cuerpo de la página, se muestra según el índice seleccionado
124| body: _paginas[_index],
125| ); // Scaffold
126| }
127| }
128| }
```

Maine.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
app5 > lib > paginas > lista.dart > Disenio
1 // Importa Flutter Material, necesario para usar widgets de Material Design
2 import 'package:flutter/material.dart';
3 // Widget de estado que representa la pantalla de Lista
4 class Lista extends StatefulWidget{
5   @override
6     State<StatefulWidget> createState() {
7       return Disenio(); // Retorna la clase que maneja el estado de la pantalla
8     }
9   }
10 // Clase que maneja el estado de Lista
11 class Disenio extends State<Lista>{
12   @override
13     Widget build(BuildContext context) {
14       return Scaffold( // Estructura básica de la pantalla
15         body: Center( // Centra el contenido en la pantalla
16           child: ListView( // Lista desplazable vertical
17             scrollDirection: Axis.vertical, // Dirección de desplazamiento (vertical por defecto)
18             children: [ // Elementos dentro de la lista
19               SizedBox(
20                 height: 100, // Altura del primer elemento
21                 width: 100, // Ancho del primer elemento
22                 child: Image.asset('assets/pa1.jpg') // Imagen del primer elemento
23               ), // SizedBox
24               Image.asset('assets/pa2.jpeg'), // Segunda imagen, sin SizedBox, toma el tamaño natural
25               Image.asset('assets/pa3.jpg') // Tercera imagen
26             ]
27           )
28         )
29       );
30     }
31   }
32 }
```

```
app5 > lib > paginas > lista.dart > Disenio
1 // Importa Flutter Material, necesario para usar widgets de Material Design
2 import 'package:flutter/material.dart';
3 // Widget de estado que representa la pantalla de Lista
4 class Lista extends StatefulWidget{
5   @override
6     State<StatefulWidget> createState() {
7       return Disenio(); // Retorna la clase que maneja el estado de la pantalla
8     }
9   }
10 // Clase que maneja el estado de Lista
11 class Disenio extends State<Lista>{
12   @override
13     Widget build(BuildContext context) {
14       return Scaffold( // Estructura básica de la pantalla
15         body: Center( // Centra el contenido en la pantalla
16           child: ListView( // Lista desplazable vertical
17             scrollDirection: Axis.vertical, // Dirección de desplazamiento (vertical por defecto)
18             children: [ // Elementos dentro de la lista
19               SizedBox(
20                 height: 100, // Altura del primer elemento
21                 width: 100, // Ancho del primer elemento
22                 child: Image.asset('assets/pa1.jpg') // Imagen del primer elemento
23               ), // SizedBox
24               Image.asset('assets/pa2.jpeg'), // Segunda imagen, sin SizedBox, toma el tamaño natural
25               Image.asset('assets/pa3.jpg') // Tercera imagen
26             ]
27           )
28         )
29       );
30     }
31   }
32 }
```

Lista.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

≡ Navigator Drawer

Open navigation menu



DEBUG



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
app5 > lib > paginas > home.dart > Disenio > build
1 // Importa Flutter Material, necesario para usar widgets de Material Design
2 import 'package:flutter/material.dart';
3 // Widget de estado que representa la pantalla principal (Home)
4 class Home extends StatefulWidget{
5   @override
6     State<StatefulWidget> createState() {
7       return Disenio(); // Retorna la clase que maneja el estado
8     }
9   }
10 // Clase que maneja el estado de Home
11 class Disenio extends State<Home>{
12   @override
13   Widget build(BuildContext context) {
14     return Scaffold( // Estructura básica de la pantalla
15       body: Padding(
16         padding: const EdgeInsets.all(8.0), // Espaciado interno de 8 píxeles
17         child: Column( // Organiza los elementos en vertical
18           children: [
19             Expanded(
20               flex: 1, // Ocupa 1 parte proporcional del espacio disponible
21               child: Center( // Centra el contenido dentro del espacio asignado
22                 child: Text('Pagina Principal'), // Texto principal
23               ), // Center
24             ), // Expanded
```

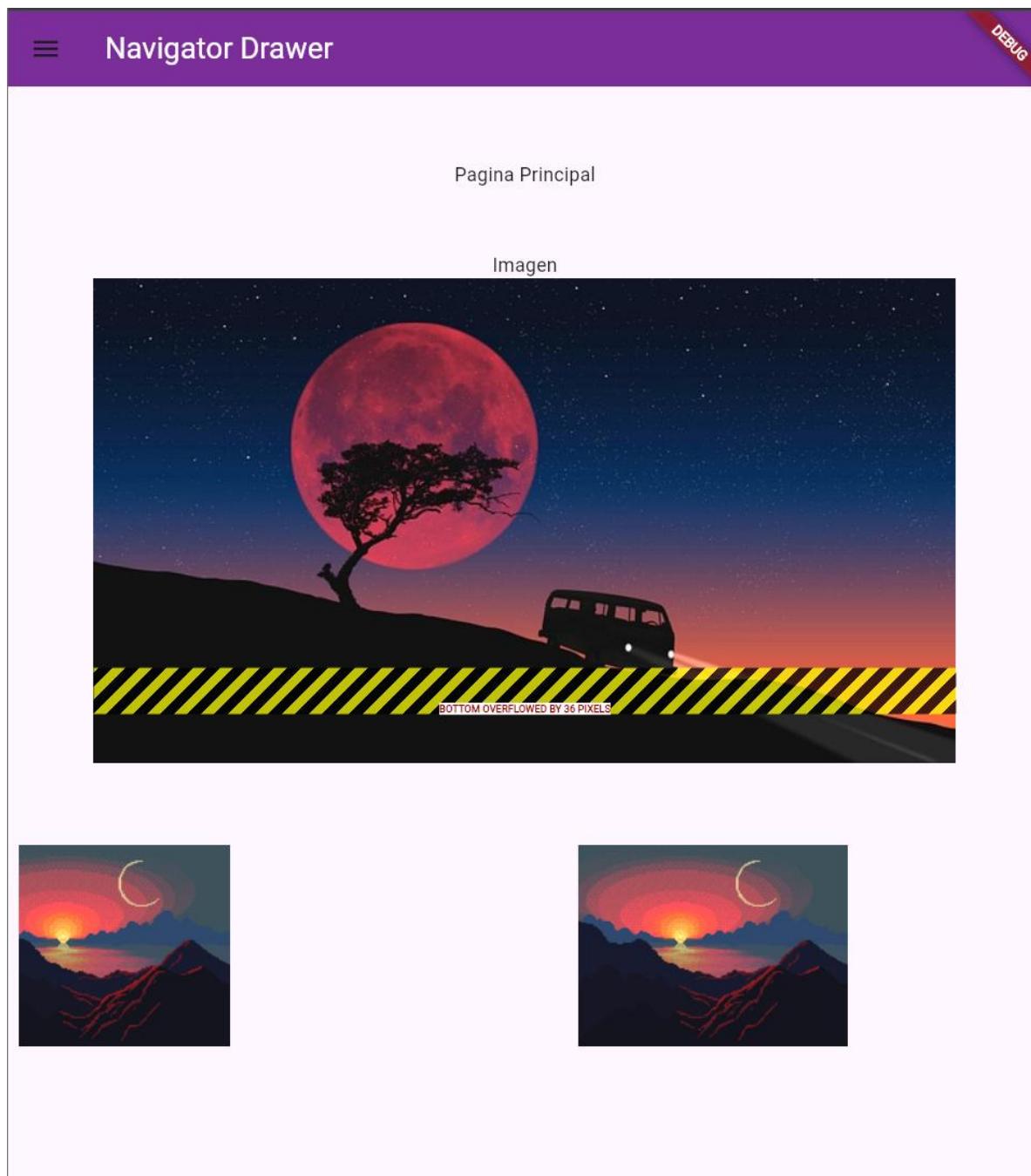
```
25           ),
26           flex: 3, // Ocupa 3 partes proporcionales
27           child: Column( // Columna para texto e imagen
28             mainAxisAlignment: MainAxisAlignment.center, // Centra verticalmente los elementos
29             children: [
30               Text('Imagen'), // Texto indicador
31               Image.network('https://cdn.pixabay.com/photo/2023/03/16/08/42/camping-7856198_640.jpg')
32             ],
33           ), // Column
34         ), // Expanded
35         Expanded(
36           flex: 3, // Ocupa 3 partes proporcionales
37           child: ListView(
38             scrollDirection: Axis.horizontal, // Desplazamiento horizontal
39             children: [ // Imágenes dentro del scroll horizontal
40               Image.network('https://i.redd.it/6ihl9vurecz01.png'),
41               Image.network('https://i.redd.it/6ihl9vurecz01.png'),
42               Image.network('https://i.redd.it/6ihl9vurecz01.png'),
43               Image.network('https://i.redd.it/6ihl9vurecz01.png')
44             ],
45           ), // ListView
46         ), // Expanded
```

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

```
34      ), // Expanded
35      Expanded(
36          flex: 3, // Ocupa 3 partes proporcionales
37          child: ListView(
38              scrollDirection: Axis.horizontal, // Desplazamiento horizontal
39              children: [ // Imágenes dentro del scroll horizontal
40                  Image.network('https://i.redd.it/6ihl9vurecz01.png'),
41                  Image.network('https://i.redd.it/6ihl9vurecz01.png'),
42                  Image.network('https://i.redd.it/6ihl9vurecz01.png'),
43                  Image.network('https://i.redd.it/6ihl9vurecz01.png')
44              ],
45          ), // ListView
46      ), // Expanded
47      ],
48  ), // Column
49  ), // Padding
50 ); // Scaffold
51 }
52 }
53 |
```

Home.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa Flutter Material, necesario para usar widgets de Material Design
2 import 'package:flutter/material.dart';
3 // Widget de estado que representa la pantalla de Datos
4 class Datos extends StatefulWidget {
5   @override
6   State<StatefulWidget> createState() {
7     return Disenio(); // Retorna la clase que maneja el estado
8   }
9   // Clase que maneja el estado de Datos
10  class Disenio extends State<Datos> {
11    @override
12    Widget build(BuildContext context) {
13      return Scaffold(
14        // Mantener AppBar con el mismo color que usas en las otras páginas
15        appBar: AppBar(
16          title: const Text('Datos'),
17          backgroundColor: Colors.purpleAccent,
18        ), // AppBar
19        // Fondo de la página similar a las imágenes
20        body: Center(
21          child: SingleChildScrollView(
22            padding: const EdgeInsets.all(16.0),
23            child: Card(
24              elevation: 12,
25              shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),
```

```
25            child: Padding(
26              padding: const EdgeInsets.symmetric(vertical: 28.0, horizontal: 22.0),
27              child: Column(
28                mainAxisAlignment: MainAxisAlignment.min,
29                children: [
30                  SizedBox(height: 18),
31                  // Nombre (titulo) - estilo acorde a la paleta
32                  Text(
33                    'Evelyn Esmeralda Lomeli Venegas',
34                    style: TextStyle(
35                      fontSize: 22,
36                      fontWeight: FontWeight.w700,
37                      color: const Color.fromARGB(255, 20, 4, 49),
38                      fontFamily: 'Verdana',
39                      fontStyle: FontStyle.italic,
40                    ), // TextStyle
41                    textAlign: TextAlign.center,
42                  ), // Text
43                  const SizedBox(height: 8),
44                  // Carrera
45                  Row(
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
47         mainAxisAlignment: MainAxisAlignment.center,
48         children: [
49             const SizedBox(width: 8),
50             Text(
51                 'Ingenieria en Informatica', // <- completa con tu carrera exacta
52                 style: TextStyle(fontSize: 16, color: Colors.black),
53             ), // Text
54         ],
55     ), // Row
56     SizedBox(height: 10),
57     // Semestre
58     Row(
59         mainAxisAlignment: MainAxisAlignment.center,
60         children: [
61             const SizedBox(width: 8),
62             Text(
63                 '5º Semestre', // Semestre
64                 style: TextStyle(fontSize: 16, color: Colors.black),
65             ), // Text
66         ],
67     ), // Row
```

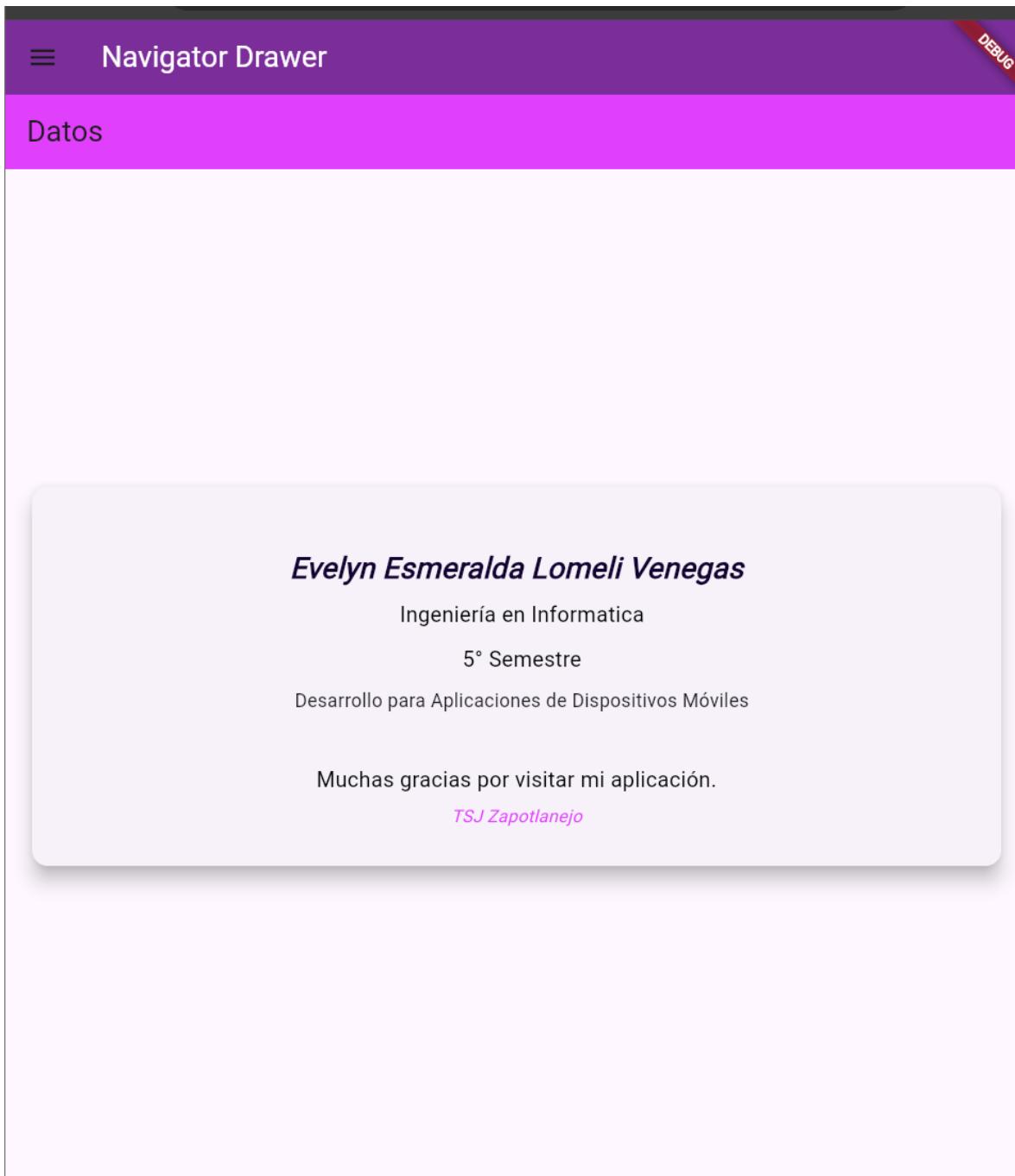
```
68     SizedBox(height: 10),
69     // Materia
70     Row(
71         mainAxisAlignment: MainAxisAlignment.center,
72         children: [
73             const SizedBox(width: 8),
74             Flexible(
75                 child: Text(
76                     'Desarrollo para Aplicaciones de Dispositivos Móviles',
77                     textAlign: TextAlign.center,
78                 ), // Text
79             ), // Flexible
80         ],
81     ), // Row
82     SizedBox(height: 20),
83
84     SizedBox(height: 18),
85     // Texto de agradecimiento con estilo ligero
86     Text(
87         'Muchas gracias por visitar mi aplicación.',
88         style: TextStyle(
89             fontSize: 16
```

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

```
90           color: Colors.black,
91           ), // TextStyle
92           textAlign: TextAlign.center,
93           ), // Text
94           SizedBox(height: 6),
95           // Pie: pequeño texto descriptivo o contacto (opcional)
96           Text(
97             'TSJ Zapotlanejo',
98             style: TextStyle(fontSize: 13, color: Colors.purpleAccent, fontStyle: FontStyle.italic),
99             ), // Text
100           ],
101           ), // Column
102           ), // Padding
103           ), // Card
104           ), // SingleChildScrollView
105           ), // Center
106           ); // Scaffold
107         }
108       }
109     |
110   }
```

Datos.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR



EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

Practica 6 Parcial II

```
1 // Importa el archivo 'pagina1.dart' desde la carpeta 'widget' dentro del proyecto 'app_6'.
2 // Este archivo contiene la definición de la primera pantalla o widget principal de la app.
3 import 'package:app_6/widget/pagina1.dart';
4 // Importa el paquete principal de Flutter que contiene todos los widgets y herramientas
5 // necesarias para crear interfaces gráficas con Material Design.
6 import 'package:flutter/material.dart';
7 // Función principal (punto de entrada del programa).
8 // Es lo primero que se ejecuta cuando se inicia la aplicación Flutter.
9 void main() {
10    // runApp() lanza la aplicación y toma como parámetro el widget raíz de la app.
11    // En este caso, se llama a MyApp, que es el widget principal.
12    runApp(const MyApp());
13 }
14 // Se define la clase MyApp, que es el widget raíz de toda la aplicación.
15 // Extiende de StatelessWidget porque su contenido no cambia mientras la app se ejecuta.
16 class MyApp extends StatelessWidget {
17    // Constructor constante de la clase MyApp.
18    // 'super.key' permite identificar este widget de forma única dentro del árbol de widgets.
19    const MyApp({super.key});
20    // Sobrescribimos el método build(), que construye la interfaz gráfica (UI) de este widget.
21    @override
22    Widget build(BuildContext context) {
23        // Retorna un widget MaterialApp, que es el contenedor principal de toda app Flutter
24        // basada en Material Design (provee tema, navegación, rutas, etc.).
25        return MaterialApp(
```

```
21    @override
22    Widget build(BuildContext context) {
23        // Retorna un widget MaterialApp, que es el contenedor principal de toda app Flutter
24        // basada en Material Design (provee tema, navegación, rutas, etc.).
25        return MaterialApp(
26            // Título de la aplicación, visible en algunos sistemas o durante la multitarea.
27            title: 'Flutter Demo',
28            // Propiedad 'home' indica cuál será la pantalla principal al iniciar la app.
29            // En este caso, se muestra el widget Pagina1, importado al inicio del archivo.
30            home: Pagina1(),
31        ); // MaterialApp
32    }
33 }
34 }
```

Main.dart

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
1 // Importa el paquete principal de Flutter con todos los widgets visuales disponibles.
2 import 'package:flutter/material.dart';
3 // Define la clase principal de la pantalla llamada "Pagina1".
4 // Es un StatefulWidget porque su estado puede cambiar (por ejemplo, los textos ingresados).
5 class Pagina1 extends StatefulWidget {
6     // Constructor constante, que llama al constructor padre con una clave opcional.
7     const Pagina1({super.key});
8     // Crea el estado asociado a este widget, devolviendo una instancia de la clase Disenio.
9     @override
10    State< StatefulWidget > createState() {
11        return Disenio();
12    }
13}
14 // Clase que maneja el estado y la lógica de "Pagina1".
15 class Disenio extends State< Pagina1 > {
16     // Controladores de texto para obtener los valores ingresados en los TextField.
17     final TextEditingController usuario = TextEditingController();
18     final TextEditingController password = TextEditingController();
19     // Función que valida los datos ingresados por el usuario.
20     void validar() {
21         // Se obtienen los valores escritos en los campos de texto.
22         String u = usuario.text;
23         String p = password.text;
24         // Verifica que los campos no estén vacíos.
25         if (u.isNotEmpty || p.isNotEmpty) {
```

```
26             // Si el usuario y la contraseña son correctos, no hace nada (puede añadirse navegación aquí).
27             if (u == 'admin' && p == '12345') {
28                 // Aquí podrías agregar navegación o mensaje de éxito.
29             } else {
30                 // Si son incorrectos, muestra un mensaje de error con SnackBar.
31                 ScaffoldMessenger.of(context).showSnackBar(
32                     SnackBar(content: Text('Error usuario incorrecto')),
33                 );
34             }
35         }
36     }
37     // Método que construye la interfaz visual de esta pantalla.
38     @override
39     Widget build(BuildContext context) {
40         // Scaffold proporciona una estructura visual básica (barra superior, cuerpo, etc.)
41         return Scaffold(
42             // Barra superior (AppBar) de la aplicación.
43             appBar: AppBar(
44                 // Color de fondo personalizado de la AppBar.
45                 backgroundColor: const Color.fromRGBO(255, 223, 149, 241),
46                 // Título que se muestra en la parte superior.
47                 title: Text('Repaso Parcial 2'),
48             ), // AppBar
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
49 | // Cuerpo principal de la pantalla.
50 | body: Padding(
51 |   // Margen interno general de 14 píxeles alrededor del contenido.
52 |   padding: const EdgeInsets.all(14.0),
53 |   // Card crea una tarjeta con bordes redondeados y sombra.
54 |   child: Card(
55 |     // Color de la sombra de la tarjeta.
56 |     shadowColor: const Color.fromRGBO(255, 143, 106, 92),
57 |     // Altura de la sombra (profundidad visual).
58 |     elevation: 50,
59 |     // Margen externo de la tarjeta.
60 |     margin: EdgeInsets.all(10),
61 |     // SingleChildScrollView permite que el contenido sea desplazable
62 |     // si excede el tamaño de la pantalla.
63 |     child: SingleChildScrollView(
64 |       // Columna que organiza los widgets en forma vertical.
65 |       child: Column(
66 |         children: [
67 |           // Contenedor con tamaño fijo para mostrar una imagen circular (avatar).
68 |           SizedBox(
69 |             width: 250,
70 |             height: 250,
```

```
71 |           // CircleAvatar muestra una imagen dentro de un círculo.
72 |           child: CircleAvatar(
73 |             // Carga una imagen desde los assets del proyecto.
74 |             child: Image.asset('assets/us.png'),
75 |             radius: 50, // Radio del círculo.
76 |           ), // CircleAvatar
77 |         ), // SizedBox
78 |         // Campo de texto para ingresar el usuario.
79 |         Padding(
80 |           padding: const EdgeInsets.only(left: 15, right: 15),
81 |           child: TextField(
82 |             // Controlador que obtiene el valor ingresado.
83 |             controller: usuario,
84 |             // Estilo del texto dentro del campo.
85 |             style: TextStyle(
86 |               fontSize: 18,
87 |               fontFamily: 'Verdana',
88 |             ), // TextStyle
89 |             ), // TextField
90 |           ), // Padding
91 |           // Campo de texto para ingresar la contraseña.
92 |           Padding(
93 |             padding: const EdgeInsets.only(left: 15, right: 15),
```

EVELYN ESMERALDA LOMELI VENEGAS

MANUAL DE PROGRAMADOR

```
94 |         child: TextField(  
95 |             // Controlador que obtiene la contraseña escrita.  
96 |             controller: password,  
97 |             // Estilo del texto.  
98 |             style: TextStyle(  
99 |                 fontSize: 18,  
100 |                 fontFamily: 'Verdana',  
101 |                     ), // TextStyle  
102 |             ), // TextField  
103 |         ), // Padding  
104 |         // Espaciado vertical entre los campos y el botón.  
105 |         SizedBox(  
106 |             width: 15,  
107 |             height: 15,  
108 |         ), // SizedBox  
109 |         // Botón de tipo "ElevatedButton" que ejecuta la función validar().  
110 |         SizedBox(  
111 |             width: double.infinity, // Ocupa todo el ancho disponible.  
112 |             child: ElevatedButton(  
113 |                 onPressed: validar, // Llama a la función de validación.  
114 |                 child: Text('Aceptar'), // Texto mostrado en el botón.  
115 |             ), // ElevatedButton  
116 |         ), // SizedBox
```

```
109 |         // Botón de tipo "ElevatedButton" que ejecuta la función validar().  
110 |         SizedBox(  
111 |             width: double.infinity, // Ocupa todo el ancho disponible.  
112 |             child: ElevatedButton(  
113 |                 onPressed: validar, // Llama a la función de validación.  
114 |                 child: Text('Aceptar'), // Texto mostrado en el botón.  
115 |             ), // ElevatedButton  
116 |         ), // SizedBox  
117 |     ],  
118 |     ), // Column  
119 |     ), // SingleChildScrollView  
120 |     ), // Card  
121 |     ), // Padding  
122 | ); // Scaffold  
123 }  
124 }  
125 |
```

Pagina1.dart

EVELYN ESMERALDA LOMELI VENEGAS
MANUAL DE PROGRAMADOR

Repasso Parcial 2

DEBUG



Aceptar