

3. Widgets no Flutter

Explicação Teórica - Flutter

1. StatefulWidget e StatelessWidget

No Flutter, temos dois tipos principais de widgets: StatelessWidget e StatefulWidget.

StatelessWidget

O StatelessWidget é um widget que não muda seu estado durante o ciclo de vida do aplicativo. Ou seja, ele representa algo que não precisa ser modificado após ser exibido. Uma vez construído, ele não sofre alterações, a menos que receba novos parâmetros. Exemplo: Um texto estático, ícones ou botões que não precisam ser atualizados.

StatefulWidget

O StatefulWidget é utilizado quando você precisa de um widget que muda seu estado ao longo do tempo, ou seja, que precisa ser atualizado conforme interações do usuário ou mudanças no sistema. Quando o estado do widget é alterado, a interface é reconstruída automaticamente, refletindo as mudanças.

Exemplo: Um formulário onde o usuário insere dados que precisam ser validados ou armazenados.

2. Layouts (Row, Column, Stack)

Flutter permite criar layouts complexos usando widgets como Row, Column e Stack, que organizam outros widgets na tela.

Row

O Row organiza seus widgets filhos em uma linha horizontal. Todos os widgets são dispostos de forma horizontal, e você pode ajustar o alinhamento e o espaçamento entre eles.

Exemplo: Colocar ícones ou botões lado a lado.

Column

O Column é semelhante ao Row, mas organiza os widgets em uma coluna vertical. Ele é útil para empilhar widgets um em cima do outro, como em formulários.

Stack

O Stack permite sobrepor widgets uns sobre os outros, em vez de organizá-los horizontalmente ou verticalmente. Isso é útil para criar layouts onde um widget aparece sobre outro, como uma imagem com um texto sobreposto.

3. Navegação entre páginas (Navigator)

O Flutter utiliza o widget Navigator para fazer a navegação entre diferentes telas. A navegação é baseada em uma pilha, onde você pode "empurrar" (push) uma nova página para o topo da pilha ou "puxar" (pop) para removê-la e voltar para a anterior.

Navigator.push()

Navega para uma nova página.

Navigator.pop()

Retorna à página anterior, removendo a atual da pilha de navegação.

4. Formulários e Validação com Gerenciamento de Estado

No Flutter, você pode criar formulários usando o widget Form, que agrupa campos de entrada de dados (como TextFormField) e facilita a validação.

TextFormField

O TextFormField é um campo de texto que pode ser validado e gerenciado dentro de um formulário. Ele é muito útil quando você precisa capturar e validar dados do usuário.

GlobalKey<FormState>

O GlobalKey<FormState> permite associar o formulário a uma chave única, possibilitando validar ou salvar os dados preenchidos.

Validação

Você pode definir funções de validação para os campos do formulário. Essas funções são executadas quando o formulário é enviado, verificando se os dados inseridos estão corretos.

Exemplos práticos

1. StatefulWidget e StatelessWidget

Imagem 1

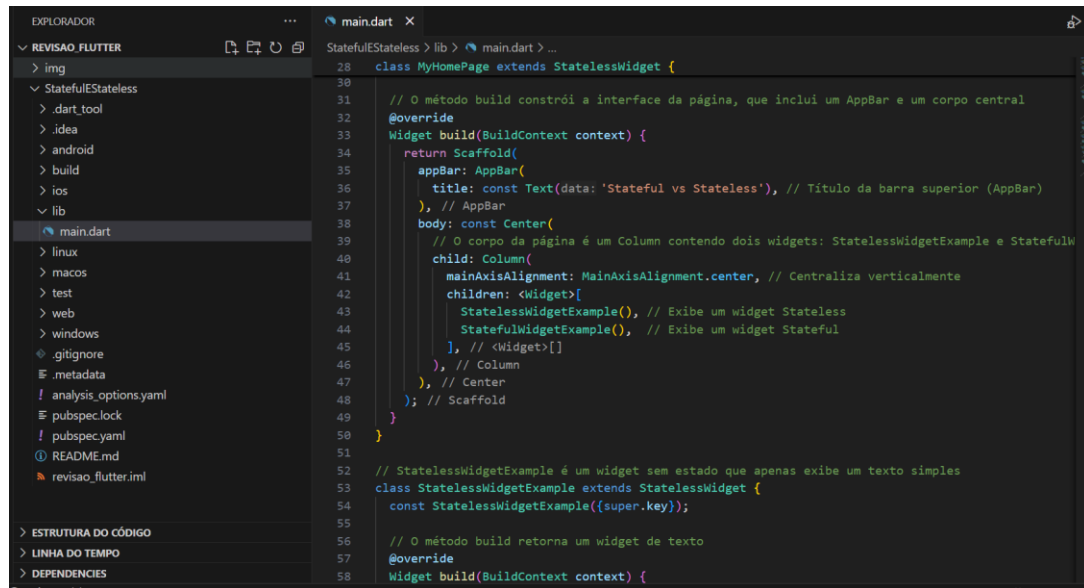


Imagem 2

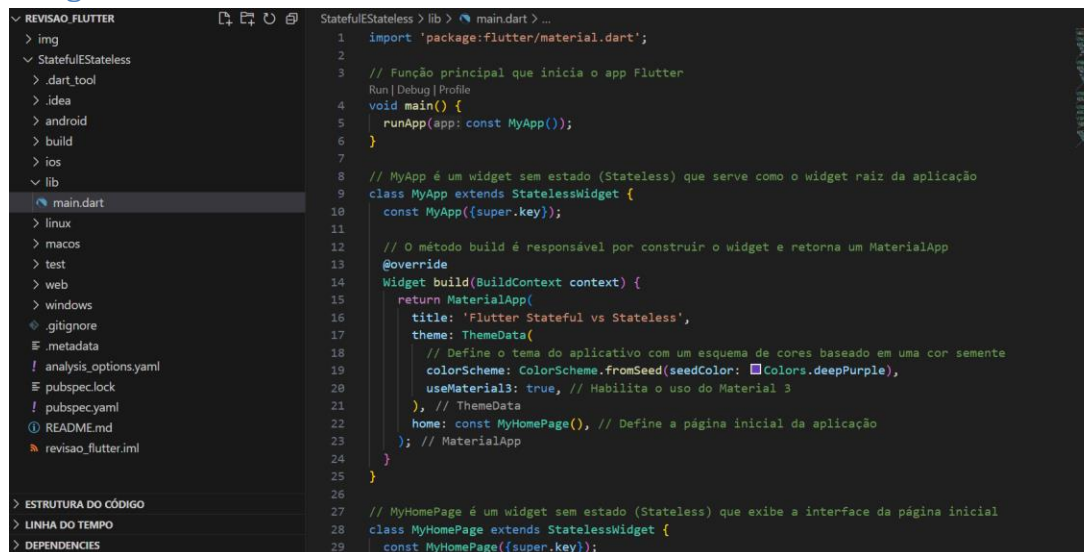
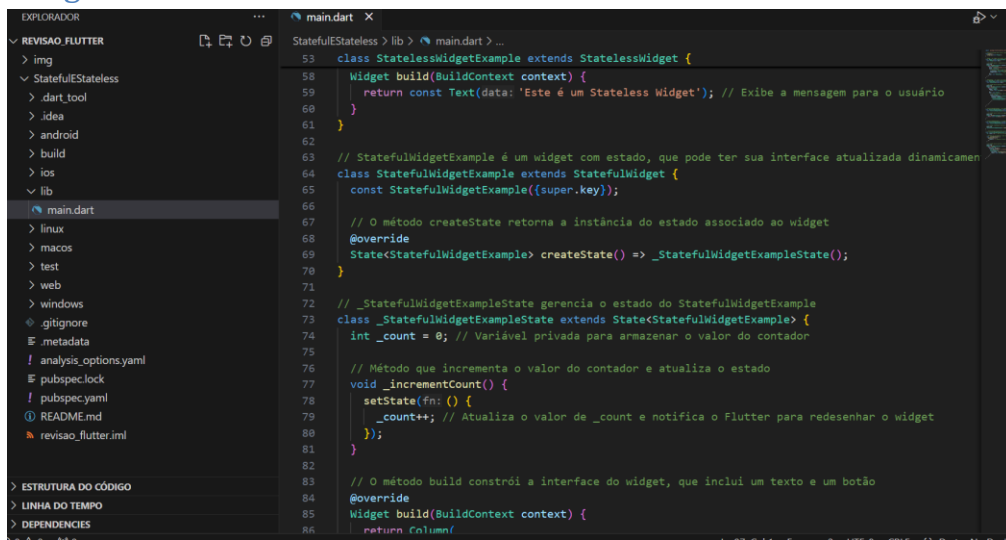


Imagem 3



```
EXPLORADOR
REVISAO_FLUTTER
  img
  StatefulStateless
  .dart_tool
  .idea
  .android
  .build
  .ios
  .lib
  main.dart
  linux
  macos
  test
  web
  windows
  .gitignore
  .metadata
  analysis_options.yaml
  pubspec.lock
  pubspec.yaml
  README.md
  revisao_flutter.iml
ESTRUTURA DO CÓDIGO
LINHA DO TEMPO
DEPENDENCIES

StatefulStateless > lib > main.dart > ...
53 class StatelessWidgetExample extends StatelessWidget {
54   Widget build(BuildContext context) {
55     return const Text(data: 'Este é um Stateless Widget'); // Exibe a mensagem para o usuário
56   }
57 }
58
59 // StatefulWidgetExample é um widget com estado, que pode ter sua interface atualizada dinamicamen
60 class StatefulWidgetExample extends StatefulWidget {
61   const StatefulWidgetExample({super.key});
62
63   // O método createState retorna a instância do estado associado ao widget
64   @override
65   State<StatefulWidgetExample> createState() => _StatefulWidgetExampleState();
66 }
67
68 // _StatefulWidgetExampleState gerencia o estado do StatefulWidgetExample
69 class _StatefulWidgetExampleState extends State<StatefulWidgetExample> {
70   int _count = 0; // Variável privada para armazenar o valor do contador
71
72   // Método que incrementa o valor do contador e atualiza o estado
73   void _incrementCount() {
74     setState(fn: () {
75       _count++; // Atualiza o valor de _count e notifica o Flutter para redesenhar o widget
76     });
77   }
78
79   // O método build constrói a interface do widget, que inclui um texto e um botão
80   @override
81   Widget build(BuildContext context) {
82     return Column(
83
```

Imagem 4



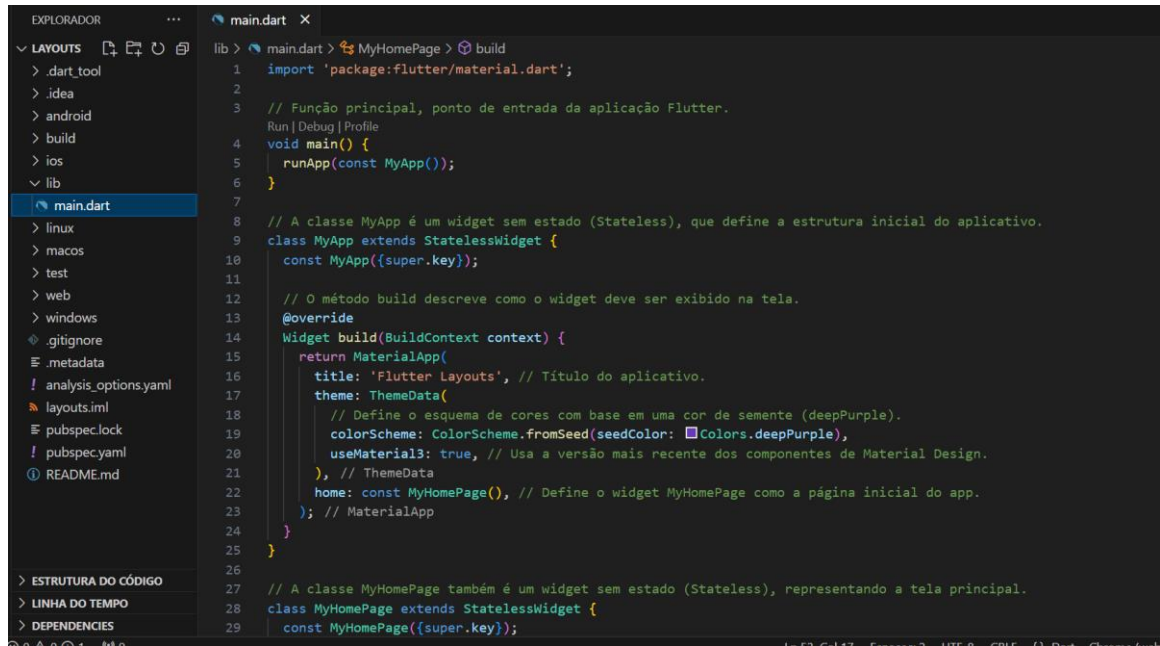
```
87     children: <Widget>[
88       Text(data: 'Este é um Stateful Widget: $_count'), // Exibe o valor atual do contador
89       ElevatedButton(
90         onPressed: _incrementCount, // Chama o método para incrementar o contador
91         child: const Text(data: 'Incrementar'), // Texto do botão
92       ), // ElevatedButton
93     ],
94   ); // Column
95 }
96 }
97
```

Imagem 5



2. Layouts (Row, Column, Stack)

Imagem 1

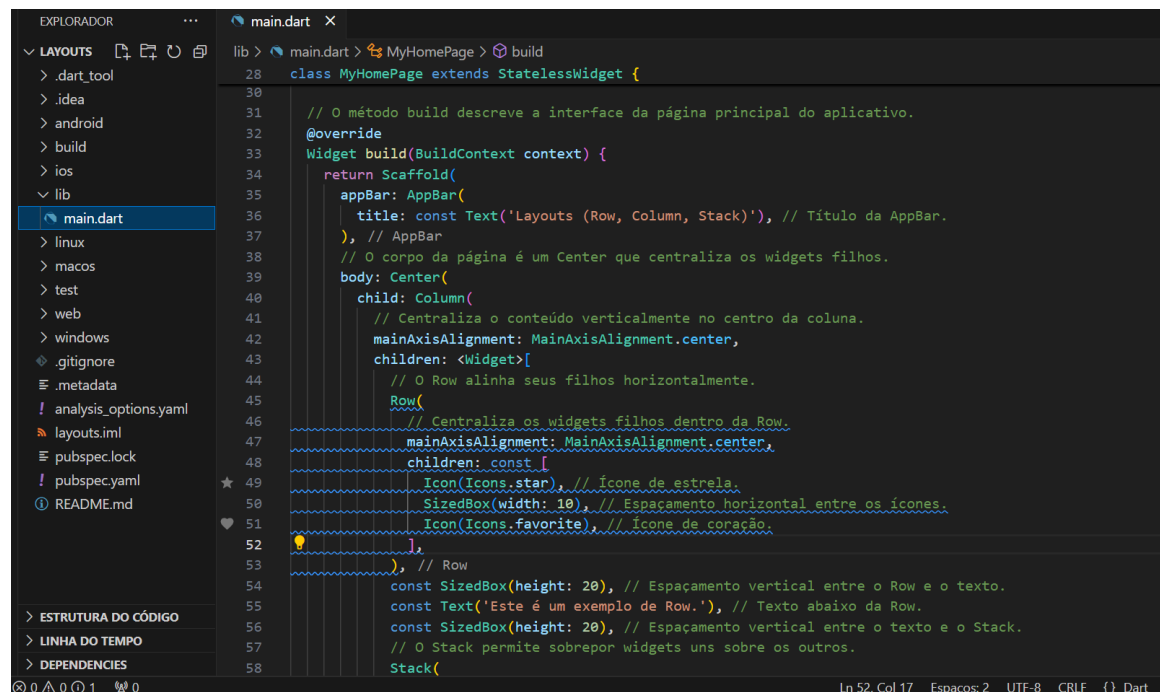


```
EXPLORADOR  ...  main.dart X
LAYOUTS  ...  lib > main.dart > MyHomePage > build
  > .dart_tool
  > .idea
  > android
  > build
  > ios
  > lib
  main.dart
  > linux
  > macos
  > test
  > web
  > windows
  > .gitignore
  > .metadata
  > analysis_options.yaml
  > layouts.iml
  > pubspec.lock
  > pubspec.yaml
  > README.md

ESTRUTURA DO CÓDIGO
LINHA DO TEMPO
DEPENDENCIES

1  import 'package:flutter/material.dart';
2
3  // Função principal, ponto de entrada da aplicação Flutter.
4  void main() {
5    runApp(const MyApp());
6  }
7
8  // A classe MyApp é um widget sem estado (Stateless), que define a estrutura inicial do aplicativo.
9  class MyApp extends StatelessWidget {
10    const MyApp({super.key});
11
12    // O método build descreve como o widget deve ser exibido na tela.
13    @override
14    Widget build(BuildContext context) {
15      return MaterialApp(
16        title: 'Flutter Layouts', // Título do aplicativo.
17        theme: ThemeData(
18          // Define o esquema de cores com base em uma cor de semente (deepPurple).
19          colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
20          useMaterial3: true, // Usa a versão mais recente dos componentes de Material Design.
21        ), // ThemeData
22        home: const MyHomePage(), // Define o widget MyHomePage como a página inicial do app.
23      ); // MaterialApp
24    }
25  }
26
27  // A classe MyHomePage também é um widget sem estado (Stateless), representando a tela principal.
28  class MyHomePage extends StatelessWidget {
29    const MyHomePage({super.key});
```

Imagem 2

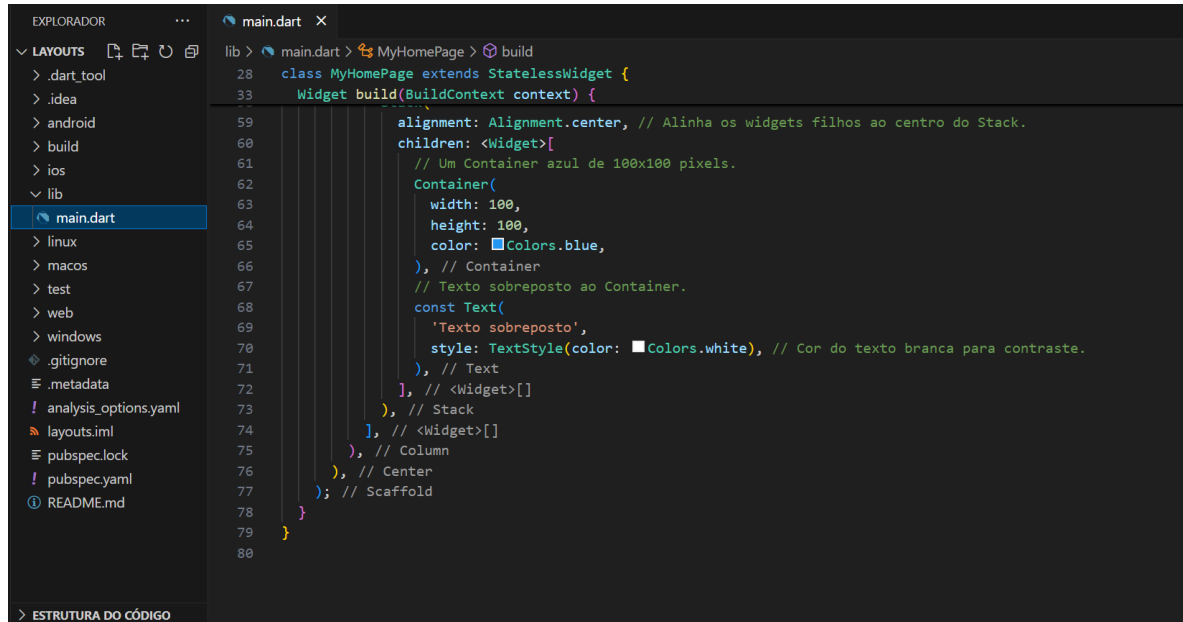


```
EXPLORADOR  ...  main.dart X
LAYOUTS  ...  lib > main.dart > MyHomePage > build
  > .dart_tool
  > .idea
  > android
  > build
  > ios
  > lib
  main.dart
  > linux
  > macos
  > test
  > web
  > windows
  > .gitignore
  > .metadata
  > analysis_options.yaml
  > layouts.iml
  > pubspec.lock
  > pubspec.yaml
  > README.md

ESTRUTURA DO CÓDIGO
LINHA DO TEMPO
DEPENDENCIES

28  class MyHomePage extends StatelessWidget {
29
30    // O método build descreve a interface da página principal do aplicativo.
31    @override
32    Widget build(BuildContext context) {
33      return Scaffold(
34        appBar: AppBar(
35          title: const Text('Layouts (Row, Column, Stack)'), // Título da AppBar.
36        ), // AppBar
37        // O corpo da página é um Center que centraliza os widgets filhos.
38        body: Center(
39          child: Column(
40            // Centraliza o conteúdo verticalmente no centro da coluna.
41            mainAxisAlignment: MainAxisAlignment.center,
42            children: <Widget>[
43              // O Row alinha seus filhos horizontalmente.
44              Row(
45                // Centraliza os widgets filhos dentro da Row.
46                mainAxisAlignment: MainAxisAlignment.center,
47                children: const [
48                  Icon(Icons.star), // Ícone de estrela.
49                  SizedBox(width: 10), // Espaçamento horizontal entre os ícones.
50                  Icon(Icons.favorite), // Ícone de coração.
51                ], // Row
52              const SizedBox(height: 20), // Espaçamento vertical entre o Row e o texto.
53              const Text('Este é um exemplo de Row.'), // Texto abaixo da Row.
54              const SizedBox(height: 20), // Espaçamento vertical entre o texto e o Stack.
55              // O Stack permite sobrepor widgets uns sobre os outros.
56              Stack(
```

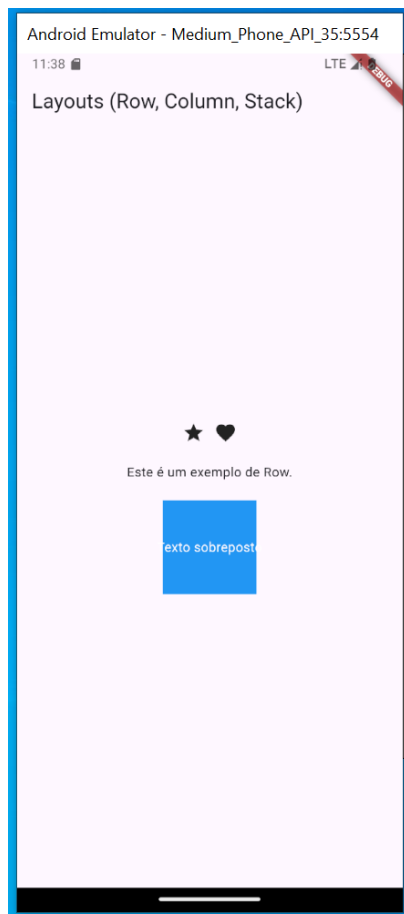
Imagem 4



```
EXPLORADOR
main.dart x
lib > main.dart > MyHomePage > build
28 class MyHomePage extends StatelessWidget {
33   Widget build(BuildContext context) {
59     alignment: Alignment.center, // Alinha os widgets filhos ao centro do Stack.
60     children: <Widget>[
61       // Um Container azul de 100x100 pixels.
62       Container(
63         width: 100,
64         height: 100,
65         color: Colors.blue,
66       ), // Container
67       // Texto sobreposto ao Container.
68       const Text(
69         'Texto sobreposto',
70         style: TextStyle(color: Colors.white), // Cor do texto branca para contraste.
71       ), // Text
72     ], // <Widget>[]
73   ), // Stack
74 ], // <Widget>[]
75 ), // Column
76 ), // Center
77 ); // Scaffold
78 }
79 }
80 }

> ESTRUTURA DO CÓDIGO
```

Imagem 5



3. Navegação entre páginas (Navigator)

Imagem 1

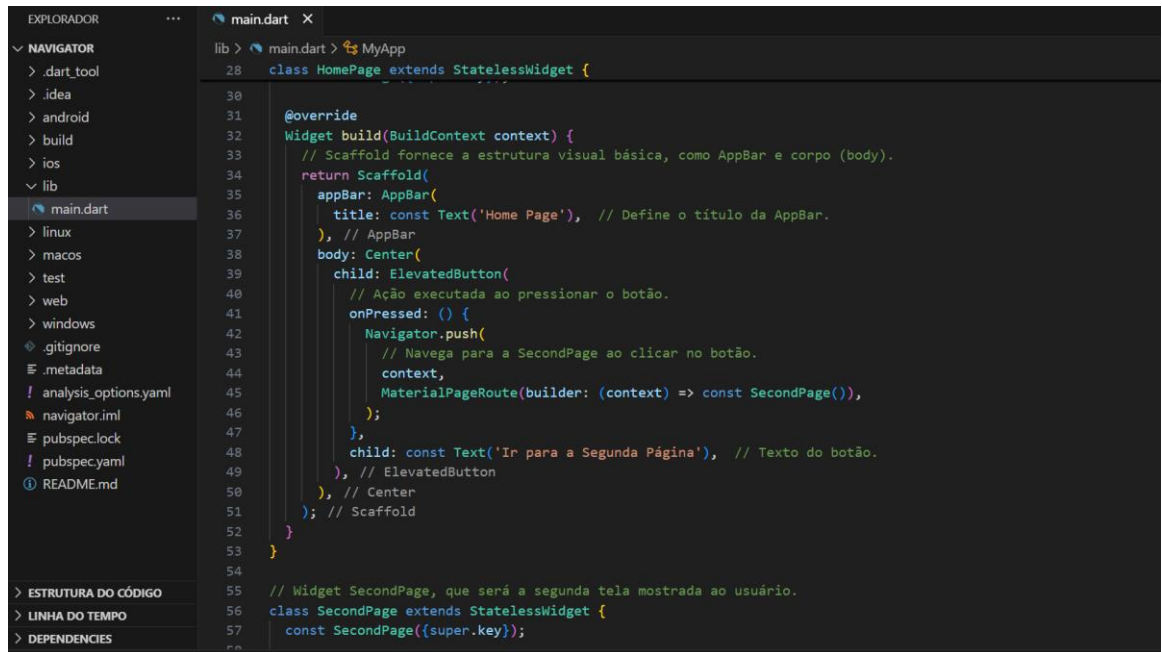


Imagem 2

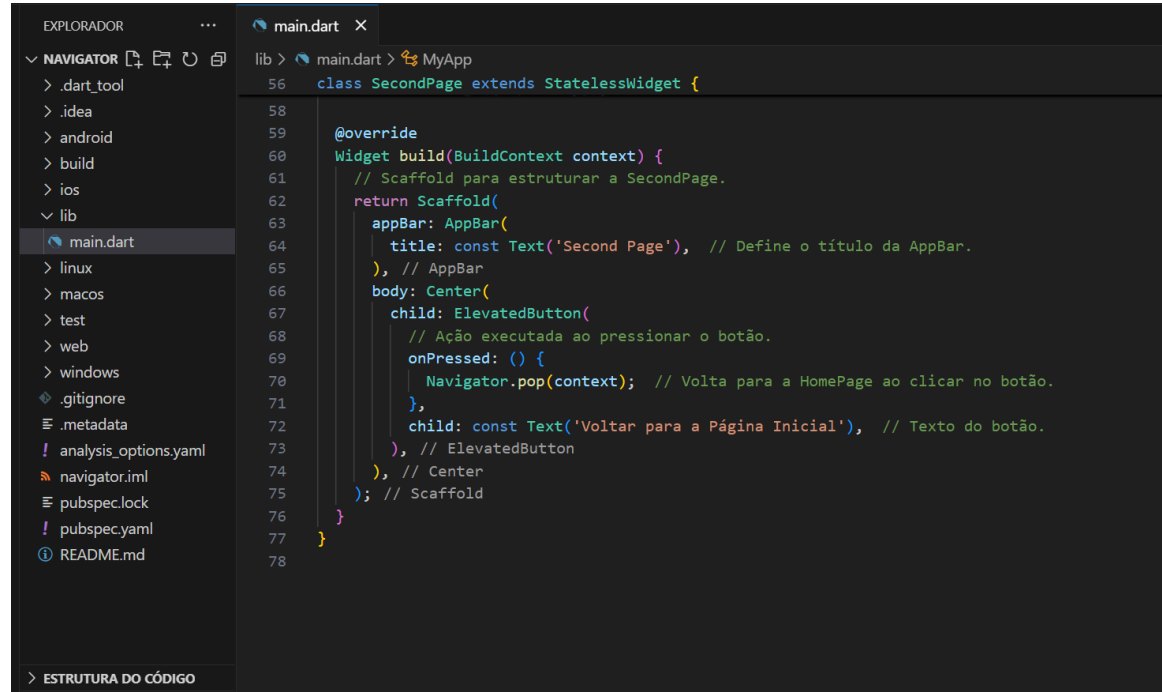


Imagem3

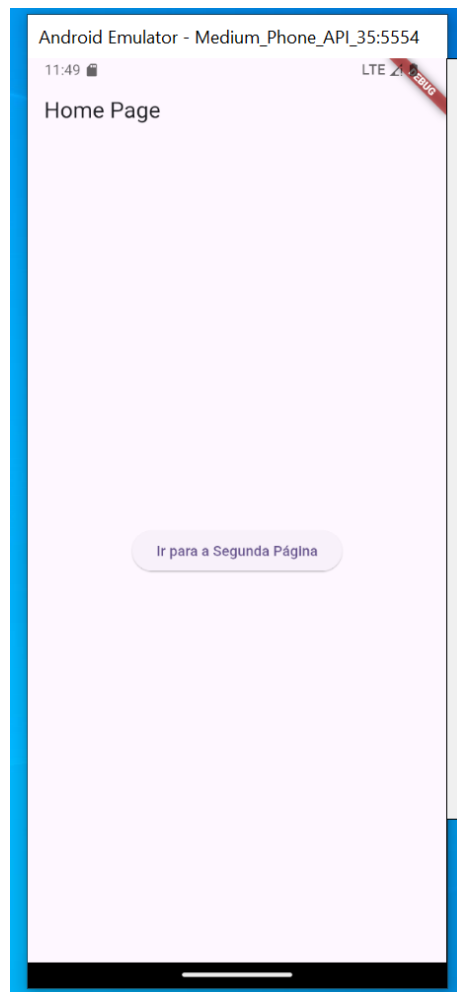
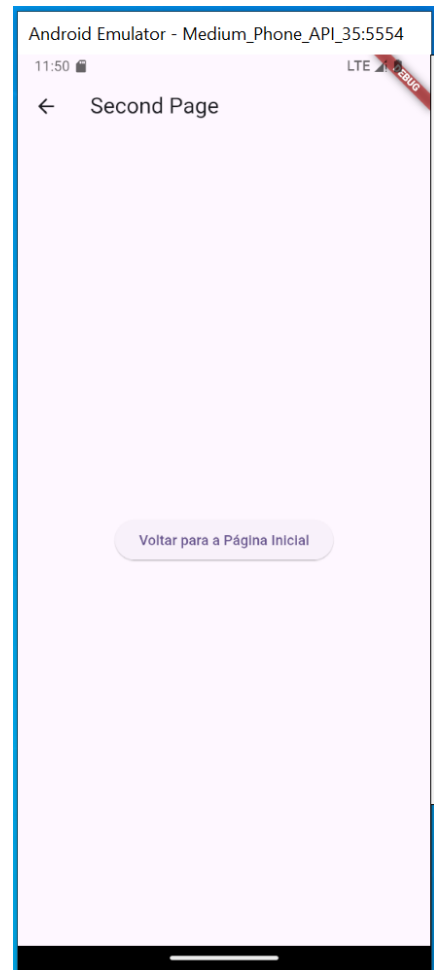


Imagem 4



4. Formulários e Validação

Imagem 1

```
main.dart x
lib > main.dart > ...
1 import 'package:flutter/material.dart';
2 import 'package:shared_preferences/shared_preferences.dart';
3
4 void main() {
5   runApp(const MyApp());
6 }
7
8 // Widget principal que inicializa o app e define a página inicial (HomePage)
9 class MyApp extends StatelessWidget {
10   const MyApp({super.key});
11
12   @override
13   Widget build(BuildContext context) {
14     return MaterialApp(
15       title: 'Cadastro e Login',
16       theme: ThemeData(
17         colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
18         useMaterial3: true,
19       ), // ThemeData
20       home: const HomePage(), // Define HomePage como a tela inicial
21     ); // MaterialApp
22   }
23 }
24
25 // Página inicial que oferece as opções de cadastro e login
26 class HomePage extends StatelessWidget {
27   const HomePage({super.key});
28
29   @override
```

Imagem 2

```
main.dart x
lib > main.dart > ...
26 class HomePage extends StatelessWidget {
27
28   Widget build(BuildContext context) {
29     return Scaffold(
30       appBar: AppBar(
31         title: const Text('Bem-vindo'),
32       ), // AppBar
33       body: Center(
34         child: Column(
35           mainAxisAlignment: MainAxisAlignment.center,
36           children: <Widget>[
37             // Botão para navegar para a página de cadastro
38             ElevatedButton(
39               onPressed: () {
40                 Navigator.of(context).push(
41                   MaterialPageRoute(
42                     builder: (context) => const RegistrationPage(),
43                   ), // MaterialPageRoute
44                 );
45               },
46               child: const Text('Cadastrar E-mail'),
47             ), // ElevatedButton
48             const SizedBox(height: 20),
49             // Botão para navegar para a página de login
50             ElevatedButton(
51               onPressed: () {
52                 Navigator.of(context).push(
53                   MaterialPageRoute(
54                     builder: (context) => const LoginPage(),
55                   ), // MaterialPageRoute
```

Imagem 3

```
lib > main.dart > ...
26 class HomePage extends StatelessWidget {
30   Widget build(BuildContext context) {
58     );
59   },
60   child: const Text('Login'),
61 ), // ElevatedButton
62 ], // <Widget>[]
63 ), // Column
64 ), // Center
65 ); // Scaffold
66 }
67 }

// Página de cadastro de e-mail e senha
70 class RegistrationPage extends StatefulWidget {
71   const RegistrationPage({super.key});
72
73   @override
74   State<RegistrationPage> createState() => _RegistrationPageState();
75 }
76
77 class _RegistrationPageState extends State<RegistrationPage> {
78   final _formKey = GlobalKey<FormState>();
79   String? _email;
80   String? _password;
81
82   // Função para salvar as credenciais usando shared_preferences
83   Future<void> _saveCredentials(String email, String password) async {
84     final prefs = await SharedPreferences.getInstance();
```

Imagem 4

```

main.dart X
lib > main.dart > ...
77 class _RegistrationPageState extends State<RegistrationPage> {
78   Future<void> _saveCredentials(String email, String password) async {
79     await prefs.setString('email', email); // Salva o e-mail localmente
80     await prefs.setString('password', password); // Salva a senha localmente
81   }
82
83   @override
84   Widget build(BuildContext context) {
85     return Scaffold(
86       appBar: AppBar(
87         title: const Text('Cadastro'),
88       ), // AppBar
89       body: Padding(
90         padding: const EdgeInsets.all(16.0),
91         child: Form(
92           key: _formKey, // Chave global para manipular o formulário
93           child: Column(
94             mainAxisAlignment: MainAxisAlignment.center,
95             children: <Widget>[
96               // Campo para inserir o e-mail
97               TextFormField(
98                 decoration: const InputDecoration(labelText: 'Digite seu e-mail'),
99                 validator: (value) {
100                   // Validação para garantir que o e-mail seja válido
101                   if (value == null || value.isEmpty) {
102                     return 'Por favor, insira seu e-mail';
103                   }
104                   if (!RegExp(r'^[a-zA-Z0-9+].[a-zA-Z0-9+]').hasMatch(value)) {
105                     return 'Insira um e-mail válido';
106                   }
107                 },
108               ),
109             ],
110           ),
111         ),
112       ),
113     );
114   }
115 }

```

Imagem 5

```
main.dart X
lib > main.dart > ...
77 class _RegistrationPageState extends State<RegistrationPage> {
90   Widget build(BuildContext context) {
112     },
113     return null;
114   },
115   onSave: (value) {
116     _email = value; // Salva o e-mail inserido
117   },
118 }, // TextFormField
119 const SizedBox(height: 20),
120 // Campo para inserir a senha
121 TextFormField(
122   decoration: const InputDecoration(labelText: 'Digite sua senha'),
123   obscureText: true, // Oculta o texto da senha
124   validator: (value) {
125     // Validação para garantir que a senha não esteja vazia
126     if (value == null || value.isEmpty) {
127       return 'Por favor, insira sua senha';
128     }
129     return null;
130   },
131   onSave: (value) {
132     _password = value; // Salva a senha inserida
133   },
134 }, // TextFormField
135 const SizedBox(height: 20),
136 // Botão de ação para cadastrar o e-mail e senha
137 ElevatedButton(
138   onPressed: () async {
```

Imagem 6

```
main.dart X
lib > main.dart > ...
77 class _RegistrationPageState extends State<RegistrationPage> {
90   Widget build(BuildContext context) {
139     if (_formKey.currentState!.validate()) {
140       _formKey.currentState!.save();
141       // Salvar o e-mail e a senha localmente
142       await _saveCredentials(_email!, _password!);
143       Navigator.of(context).pop(); // Retorna à tela inicial
144     }
145   },
146   child: const Text('Cadastrar E-mail'),
147 }, // ElevatedButton
148 ], // <Widget>[]
149 ), // Column
150 ), // Form
151 ), // Padding
152 ); // Scaffold
153 }
154 }
155
156 // Página de login, onde o usuário insere e-mail e senha
157 class LoginPage extends StatefulWidget {
158   const LoginPage({super.key});
159
160   @override
161   State<LoginPage> createState() => _LoginPageState();
162 }
163
164 class _LoginPageState extends State<LoginPage> {
165   final _loginFormKey = GlobalKey<FormState>();
```

Imagem 7

```
main.dart X
lib > main.dart > ...
164 class _LoginPageState extends State<LoginPage> {
166   String? _loginEmail;
167   String? _loginPassword;
168
169   // Função para carregar as credenciais salvas
170   Future<void> _loadCredentials() async {
171     final prefs = await SharedPreferences.getInstance();
172     setState(() {
173       // Carrega o e-mail e senha salvos
174       UserSession.email = prefs.getString('email');
175       UserSession.password = prefs.getString('password');
176     });
177   }
178
179   @override
180   void initState() {
181     super.initState();
182     _loadCredentials(); // Carrega as credenciais ao iniciar a página de login
183   }
184
185   @override
186   Widget build(BuildContext context) {
187     return Scaffold(
188       appBar: AppBar(
189         title: const Text('Login'),
190       ), // AppBar
191       body: Padding(
192         padding: const EdgeInsets.all(16.0),
193         child: Form(
194           key: loginFormKey, // Chave global para manipular o formulário
```

Imagem 8

```
main.dart X
lib > main.dart > ...
164 class _LoginPageState extends State<LoginPage> {
186   Widget build(BuildContext context) {
195     child: Column(
196       mainAxisAlignment: MainAxisAlignment.center,
197       children: <Widget>[
198         // Campo para inserir o e-mail no login
199         TextFormField(
200           decoration: const InputDecoration(labelText: 'Digite seu e-mail'),
201           validator: (value) {
202             // Validação para garantir que o e-mail seja válido
203             if (value == null || value.isEmpty) {
204               return 'Por favor, insira seu e-mail';
205             }
206             if (!RegExp(r'^[a-zA-Z0-9+@[\\]]+\\.([a-zA-Z0-9+@[\\]]+)$').hasMatch(value)) {
207               return 'Insira um e-mail válido';
208             }
209             return null;
210           },
211           onSave: (value) {
212             _loginEmail = value; // Salva o e-mail inserido no login
213           },
214         ), // TextFormField
215         const SizedBox(height: 20),
216         // Campo para inserir a senha no login
217         TextFormField(
218           decoration: const InputDecoration(labelText: 'Digite sua senha'),
219           obscureText: true, // Oculta o texto da senha
220           validator: (value) {
221             // Validação para garantir que a senha não esteja vazia
```

Imagem 9

```
main.dart X
lib > main.dart > ...
164 class _LoginPageState extends State<LoginPage> {
186   Widget build(BuildContext context) {
222     if (value == null || value.isEmpty) {
223       return 'Por favor, insira sua senha';
224     }
225     return null;
226   },
227   onSave: (value) {
228     _loginPassword = value; // Salva a senha inserida no login
229   },
230   ), // TextFormField
231   const SizedBox(height: 20),
232   // Botão para fazer o login
233   ElevatedButton(
234     onPressed: () {
235       if (_loginFormKey.currentState!.validate()) {
236         _loginFormKey.currentState!.save();
237         // Verifica se o e-mail e senha correspondem às credenciais salvas
238         if (_loginEmail == UserSession.email &&
239             _loginPassword == UserSession.password) {
240           Navigator.of(context).pushReplacement(
241             MaterialPageRoute(
242               builder: (context) => UserArea(email: UserSession.email!),
243             ), // MaterialPageRoute
244           );
245         } else {
246           // Exibe uma mensagem de erro se o login falhar
247           ScaffoldMessenger.of(context).showSnackBar(
248             const SnackBar(content: Text('E-mail ou senha incorretos.')),
```

Imagem 10

```
main.dart X
lib > main.dart > ...
164 class _LoginPageState extends State<LoginPage> {
186   Widget build(BuildContext context) {
250     }
251   }
252 },
253 child: const Text('Entrar'),
254 ), // ElevatedButton
255 ], // <Widget>[]
256 ), // Column
257 ), // Form
258 ), // Padding
259 ); // Scaffold
260 }
261 }
262
263 // Página de área do usuário após o login bem-sucedido
264 class UserArea extends StatelessWidget {
265   final String email;
266
267   const UserArea({super.key, required this.email});
268
269   @override
270   Widget build(BuildContext context) {
271     return Scaffold(
272       appBar: AppBar(
273         title: const Text('Área do Usuário'),
274         actions: [
275           // Botão para fazer logout e voltar para a página inicial
276           IconButton(
```

Imagem 11

```
main.dart X
lib > main.dart > ...
264 class UserArea extends StatelessWidget {
270   Widget build(BuildContext context) {
277     icon: const Icon(Icons.logout),
278     onPressed: () {
279       Navigator.of(context).pushReplacement(
280         MaterialPageRoute(builder: (context) => const HomePage()),
281       );
282     },
283   ), // IconButton
284 ],
285 ), // AppBar
286 body: Center(
287   child: Text('Bem-vindo, $email!', style: const TextStyle(fontSize: 24)),
288 ), // Center
289 ); // Scaffold
290 }
291 }
292
293 // Classe para manter o estado do usuário (sessão temporária)
294 class UserSession {
295   static String? email; // E-mail do usuário
296   static String? password; // Senha do usuário
297 }
298
```

Imagem 12 – Tela Inicial

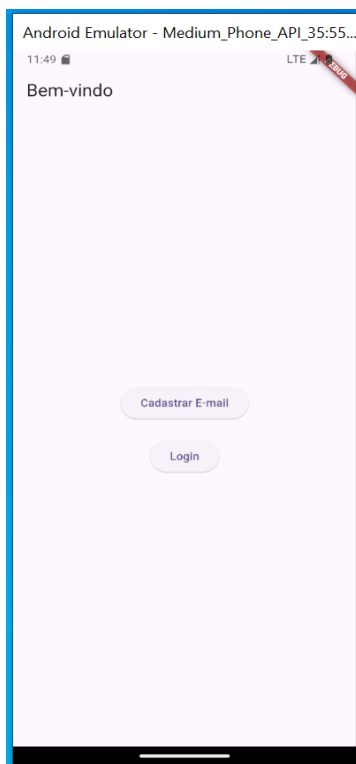


Imagem 13 – Tela Cadastro

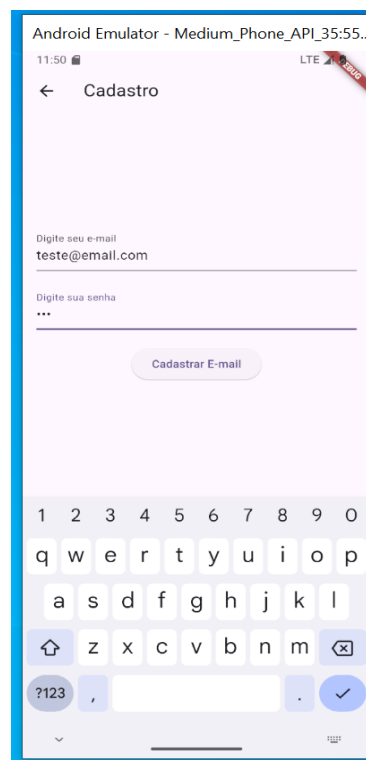


Imagem 14 - Tela Login

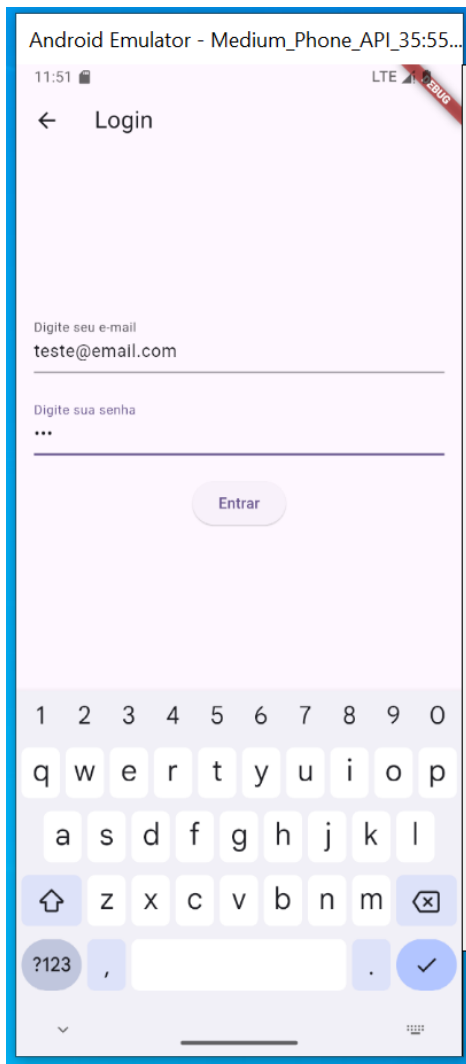


Imagem 15 - Tela Usuário

