

DOCUMENTAÇÃO DE DESENVOLVIMENTO DE APLICATIVOS MÓVEIS

com FlutterFLow

projeto: Sistema de Venda de sucos
Davi de Mello Diniz Gomes

Dezembro/2025

Índice

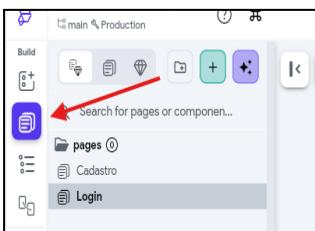
- A - Tela de login**
- B - Tela de Cadastro**
- C - Tela Home**
- D - Tela de Endereço do Usuário**
- E - Tela de Dados do Usuário**
- F - API CALL**

A - Tela de login

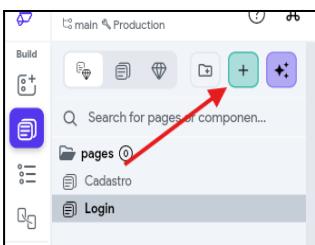
Resultado final:



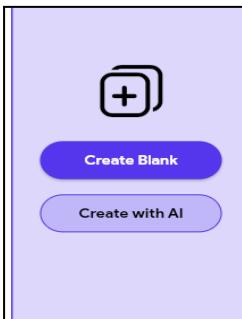
1. Criar uma página chamado “Login”



1.1 Para criar uma página no *flutterflow*, selecione, no canto esquerdo, o ícone chamado *Page Selector*.

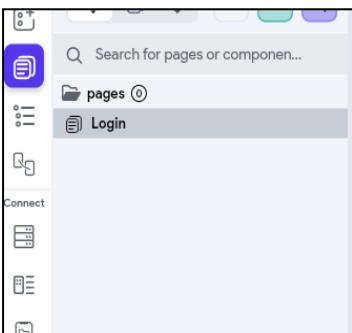


1.2 Logo, selecione *Add Page, Component, or Flow*



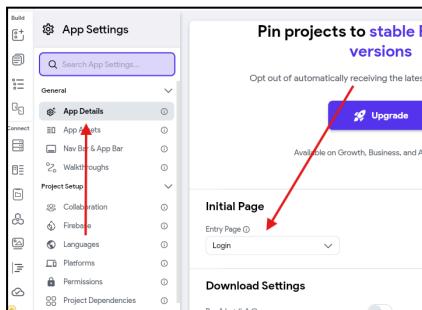
1.3 Selecione *Create Blank*, após isso, escreva um nome (em *page name*) chamado **Login** e selecione a opção *Create Page*.

obs.: caso não apareça assim:



Repita o processo **com atenção**, se aparecer, parabéns você seguiu o processo corretamente e então selecione a página Login e arraste para a pasta “pages”.

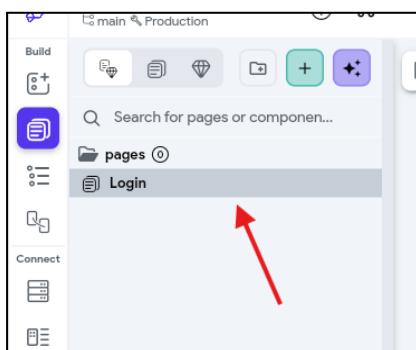
2. Definir página Login como página inicial



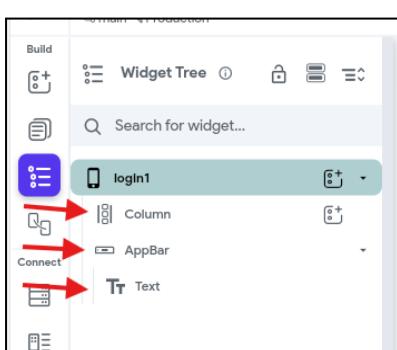
2.1 Para criar uma página no *flutterflow*, selecione, no canto esquerdo, o ícone chamado **Settings and Integrations**.

2.2 Logo, selecione **app details**, procure por *Initial Page*, e em *Entry Page* selecione a página Login.

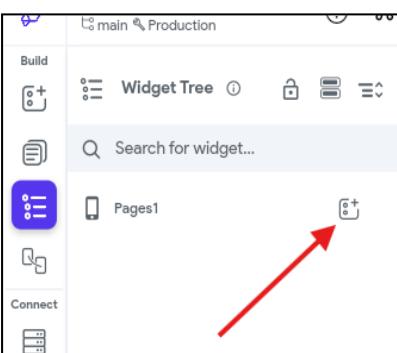
3. Editar a página de “Login”



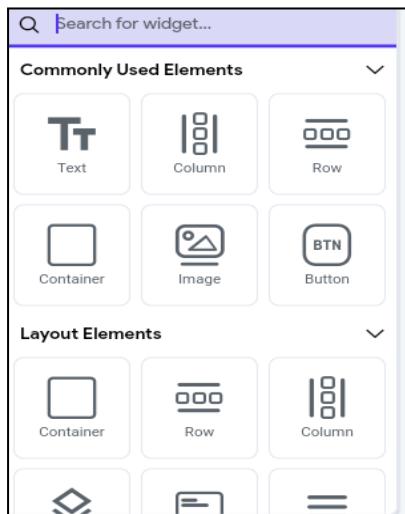
3.1 (Antes de editar, selecione a página Login e arraste para a pasta pages) Para editar a página Login, de Selecione a página Login, e após isso, selecione o ícone chamado **widget tree** , no canto esquerdo, abaixo **Page Selector**.



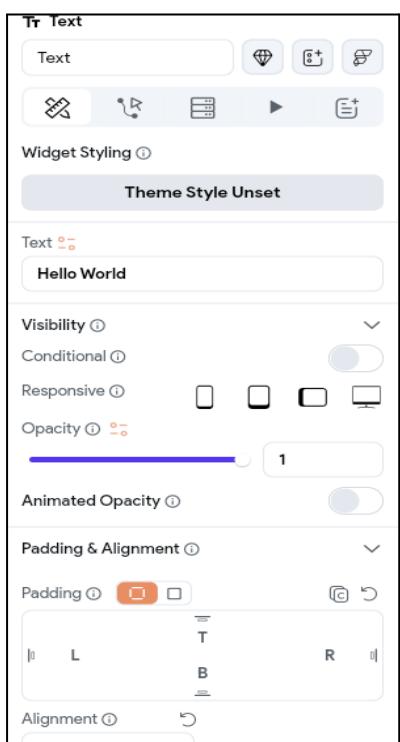
3.2 Apague todos os **widgets** no **widget tree** do Login, para melhor organização.



3.3 Para criar um widget, selecione o ícone chamado **Add a child to this widget**.



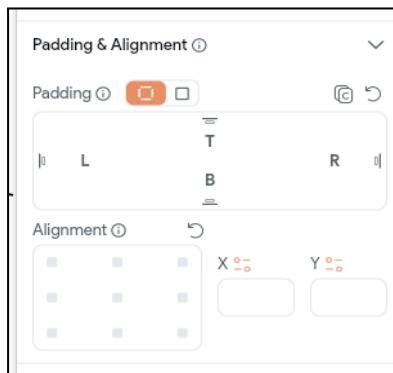
3.4 Escolha os widgets (como “text” para adicionar textos, “column” para organizar os widgets e colunas, “row” para sobrepor, textField para o usuário escrever seu email, senha e pesquisas,) para customizar sua página.



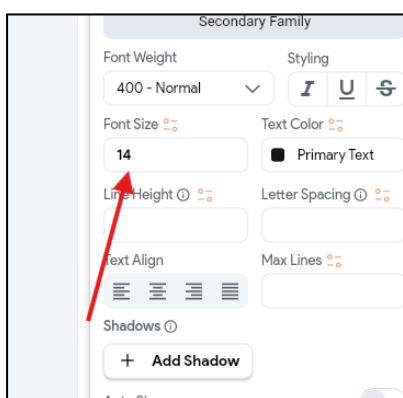
3.5 Para configurar seu widgets, selecione widget e vá no canto direito,

3.6 Lá (no canto direito) você terá como configurar:

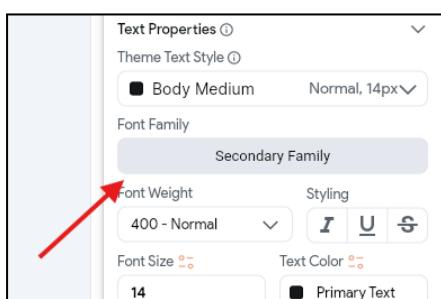
a posição:



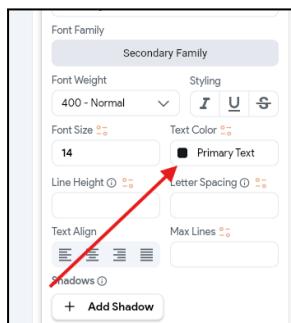
tamanho:



fonte:

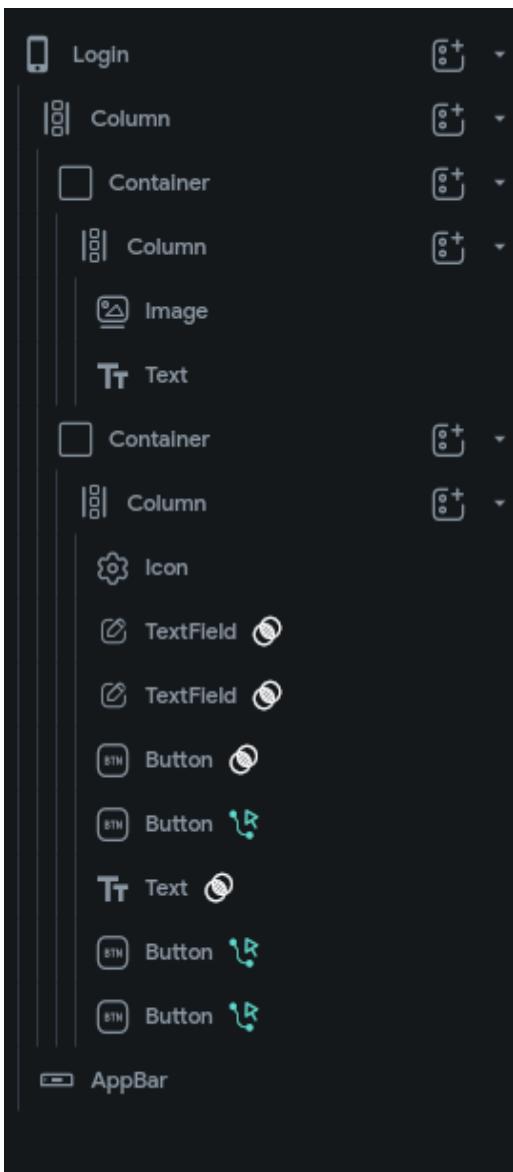


cor:



e comportamento, estilos e etc...

obs.: configuração(widgets) usada para fazer o resultado final



Em login é necessário: a solicitação de email e senha

B - Tela de Cadastro

Resultado final:



Bem-vindo ao Limão do Vale!



Preencha tudo para o seu Cadastro.

Usuário —
Crie seu Usuário

Email —
Insira seu email

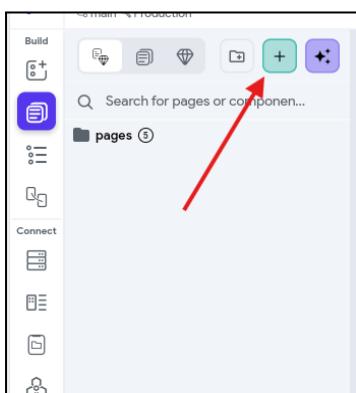
Senha —
Crie uma senha

Comfirme sua senha

Cadastrar-se 

Já possui login?

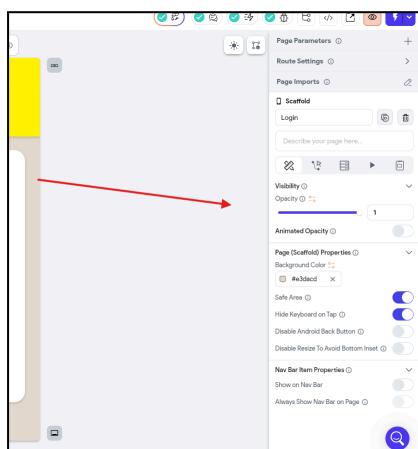
1. Criar uma página chamado “Cadastro”



1.1 Para criar uma página no *flutterflow*, repita o mesmo processo demonstrado na criação do login (selecione, no canto esquerdo, o ícone chamado *Page Selector*, selecione *Add Page*, *Component*, or *Flow*, Selecione *Create Blank*, após isso, escreva um nome (em *page name*) chamado **Cadastro** e selecione a opção *Create Page*.)

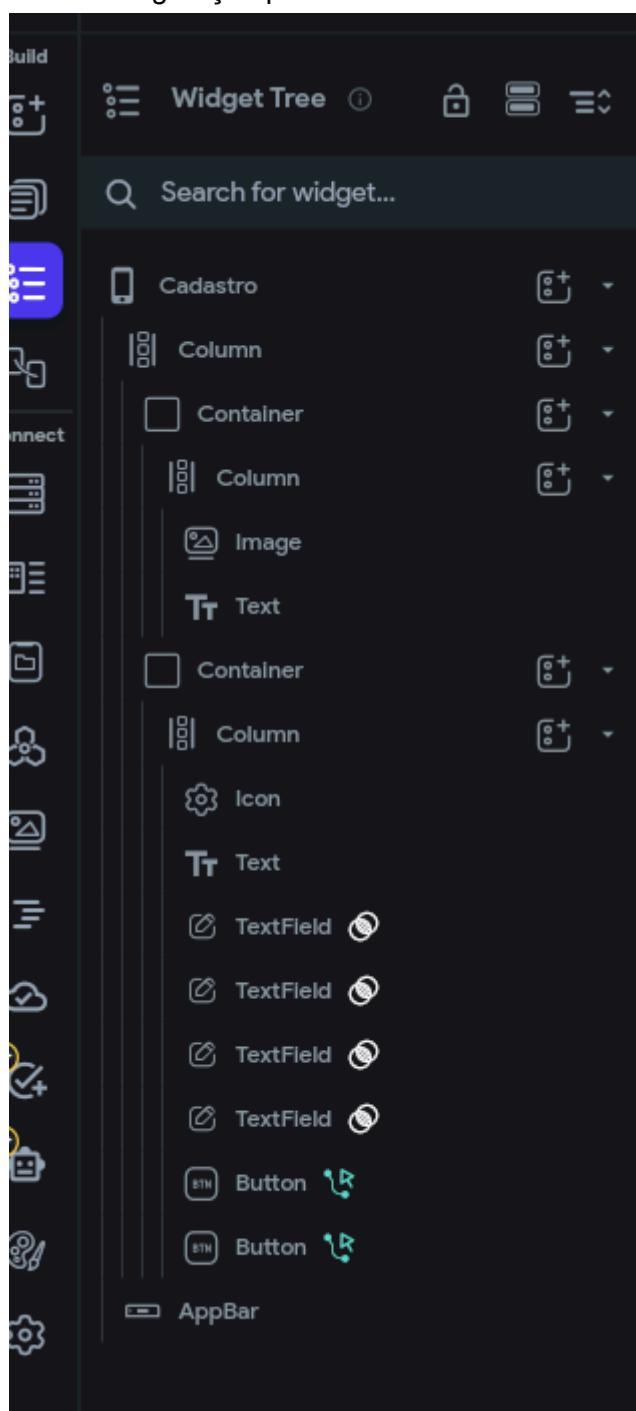
2. Editar a página de “Cadastro”

2.1 Para editar a página Cadastro, repita o mesmo processo na edição de login(Seleciona a página cadastro, e após isso, selecione o ícone chamado *widget tree* , no canto esquerdo, abaixo *Page Selector*. E depois apague todos widgets presentes para melhor organização



2.2 E configure seus widgets, selecione widget e vá no canto direito.

Obs.: Configuração para fazer o resultado final:



Em cadastro é necessário: a possibilidade de criar um usuário, colocar email e senha.
Em caso de não possuir uma conta

C - Tela de Home

Resultado Final:



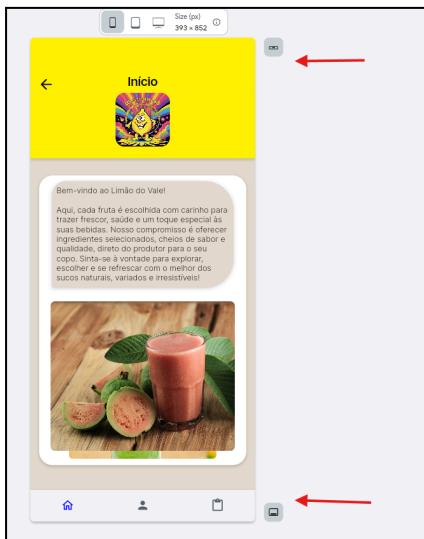
1. Criar uma página chamado “Home”

1.1 Para criar uma página no *flutterflow*, repita o mesmo processo demonstrado na criação do login (selecione, no canto esquerdo, o ícone  chamado *Page Selector*, selecione  *Add Page, Component, or Flow*, Selecione *Create Blank*, após isso, escreva um nome (em *page name*) chamado **Home** e selecione a opção *Create Page*).

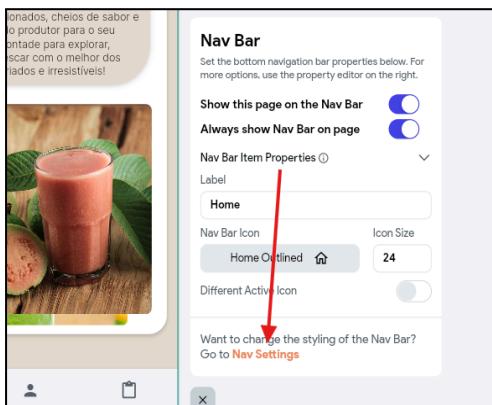
2. Editar a página de “Home”

2.1 Apague todos os widgets presentes para melhor organização. E depois para editar a página Home, repita o mesmo processo na edição de login(Selecione a página, e após isso, selecione o ícone  chamado *widget tree* , no canto esquerdo, abaixo  *Page Selector*.)

3. Adicionar “header” e “footer” ao “Home”

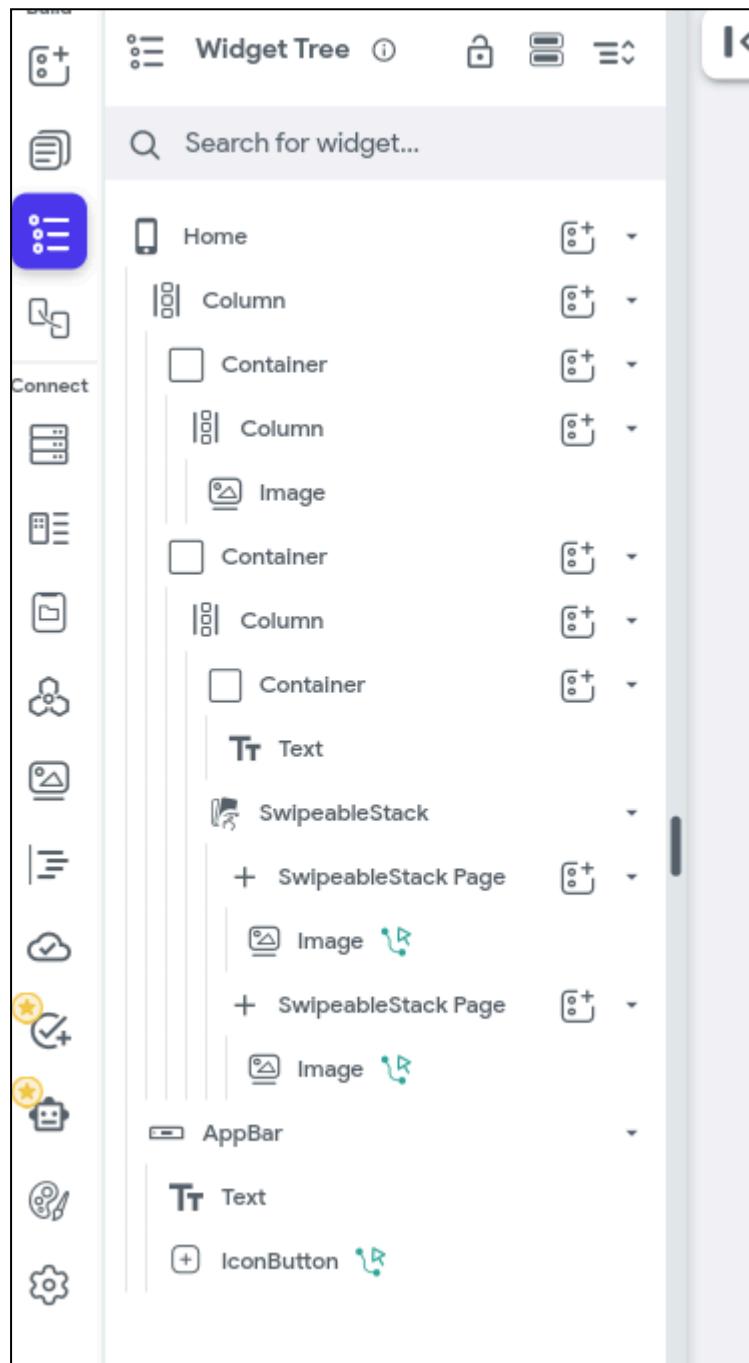


3.1 Para adicionar o *Header* selecione o ícone na parte superior direito da tela:  , após isso selecione um dos modelos e customize-os como os *widgets*.



3.2 Para adicionar o *Footer* selecione o ícone na parte inferior da tela:  , após isso selecione a opção “*Nav Settings*” destacado de laranja. Após isso, customize.

Obs.: Configuração para fazer o resultado final



Em home é necessário: a possibilidade de criar um usuário, colocar email e senha.
Em caso de não possuir uma conta

D - Tela de Endereço do Usuário

Resultado Final:



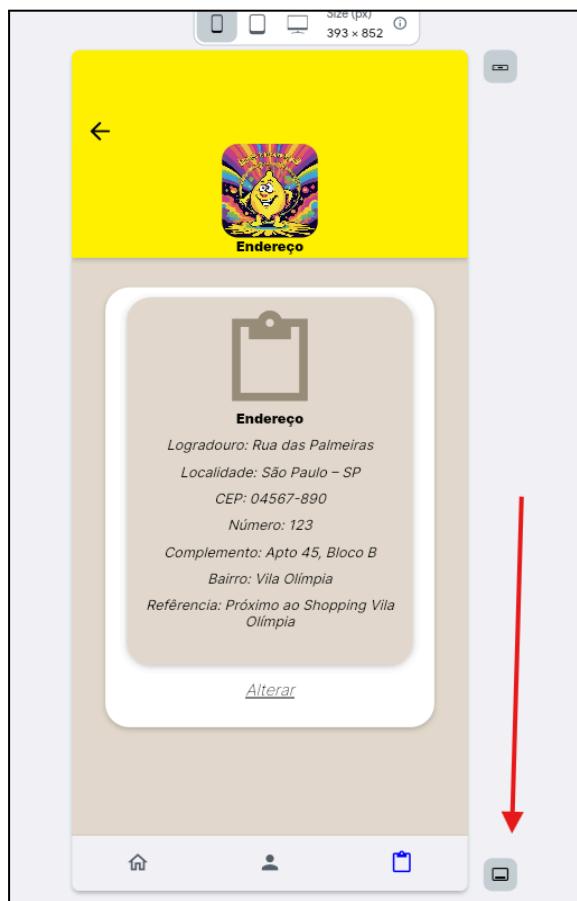
1. Criar uma página chamado “Endereco”

1.1 Para criar uma página no *flutterflow*, repita o mesmo processo demonstrado na criação do login (selecione, no canto esquerdo, o ícone  chamado *Page Selector*, selecione  *Add Page, Component, or Flow*, Selecione *Create Blank*, após isso, escreva um nome (em *page name*) chamado **Endereco** e selecione a opção *Create Page*).

2. Editar a página de “Endereco”

2.1 Para editar a página Endereco, apague todos widgets presentes para melhor organização *E depois* repita o mesmo processo na edição de login (Selecione a página, e após isso, selecione o ícone  chamado *widget tree* , no canto esquerdo, abaixo *Page Selector*.)

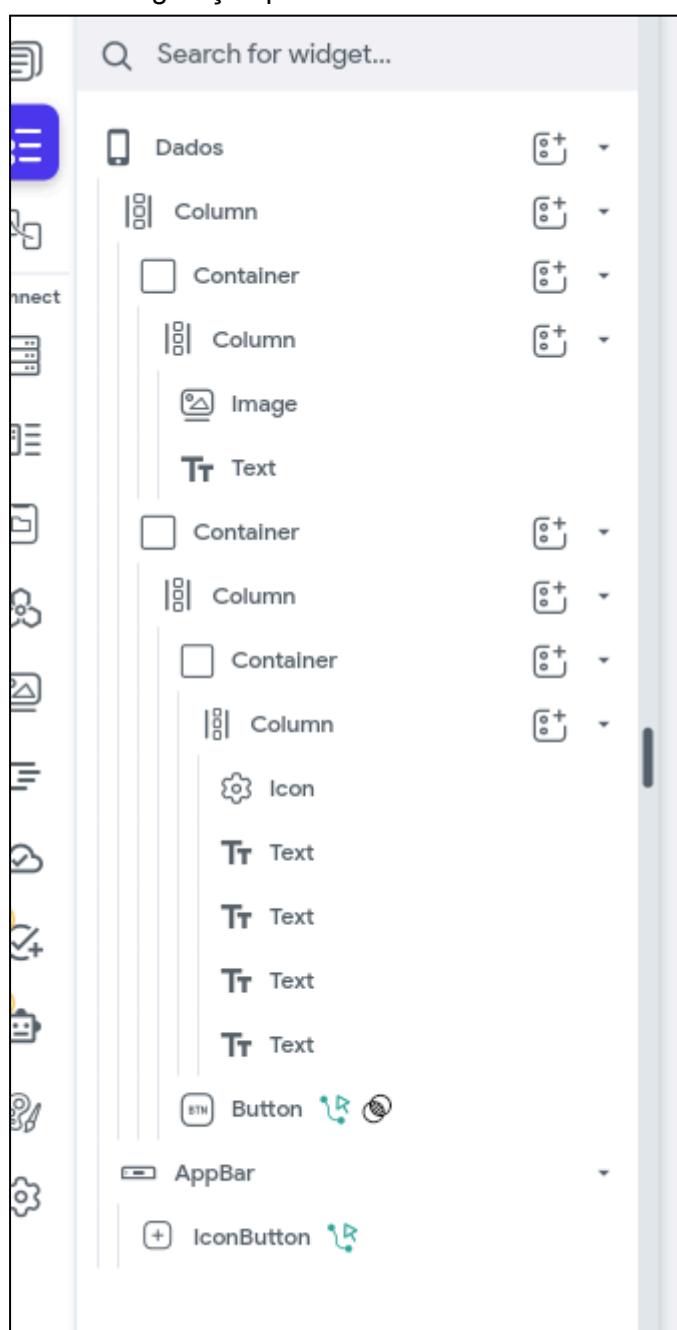
3. Adicionar ““footer” ao “Endereco”



3.1 Adicione o *Footer* selecionando o ícone na parte inferior da tela:  , após isso selecione a opção “Nav Settings” destacado de laranja. Após isso, customize.

Obs.: o Footer se “conectam” automaticamente no Flutter Flow.

Obs.: Configuração para fazer o resultado final



Lembrando que o Endereço deve possuir: logradouro, localidade, cep, número, complemento, bairro, referência e botão para salvar.

E - Tela de Dados do Usuário

Resultado Final:



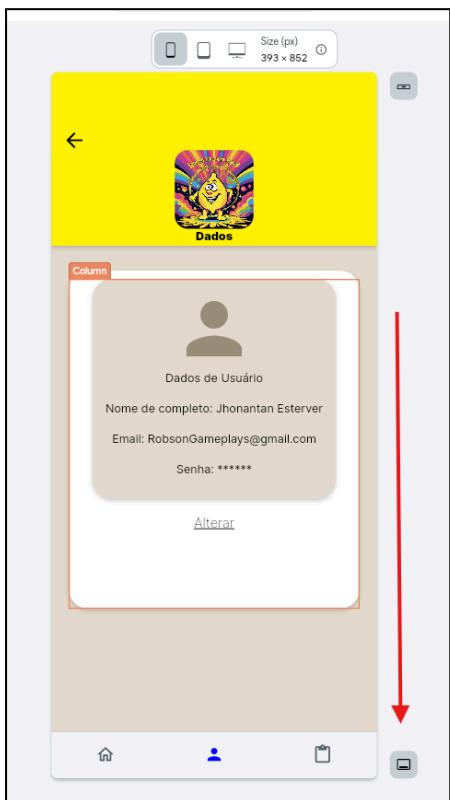
1. Criar uma página chamado “dados”

1.1 Para criar uma página no *flutterflow*, repita o mesmo processo demonstrado na criação do login (selecione, no canto esquerdo, o ícone  chamado *Page Selector*, selecione  *Add Page, Component, or Flow*, Selecione *Create Blank*, após isso, escreva um nome (em *page name*) chamado **Dados** e selecione a opção *Create Page*.).

2. Editar a página de “Dados”

2.1 Para editar a página Dados, repita o mesmo processo na edição de login(Selecione a página cadastro, e após isso, selecione o ícone  chamado *widget tree* , no canto esquerdo, abaixo  *Page Selector*. E depois apague todos widgets presentes para melhor organização.

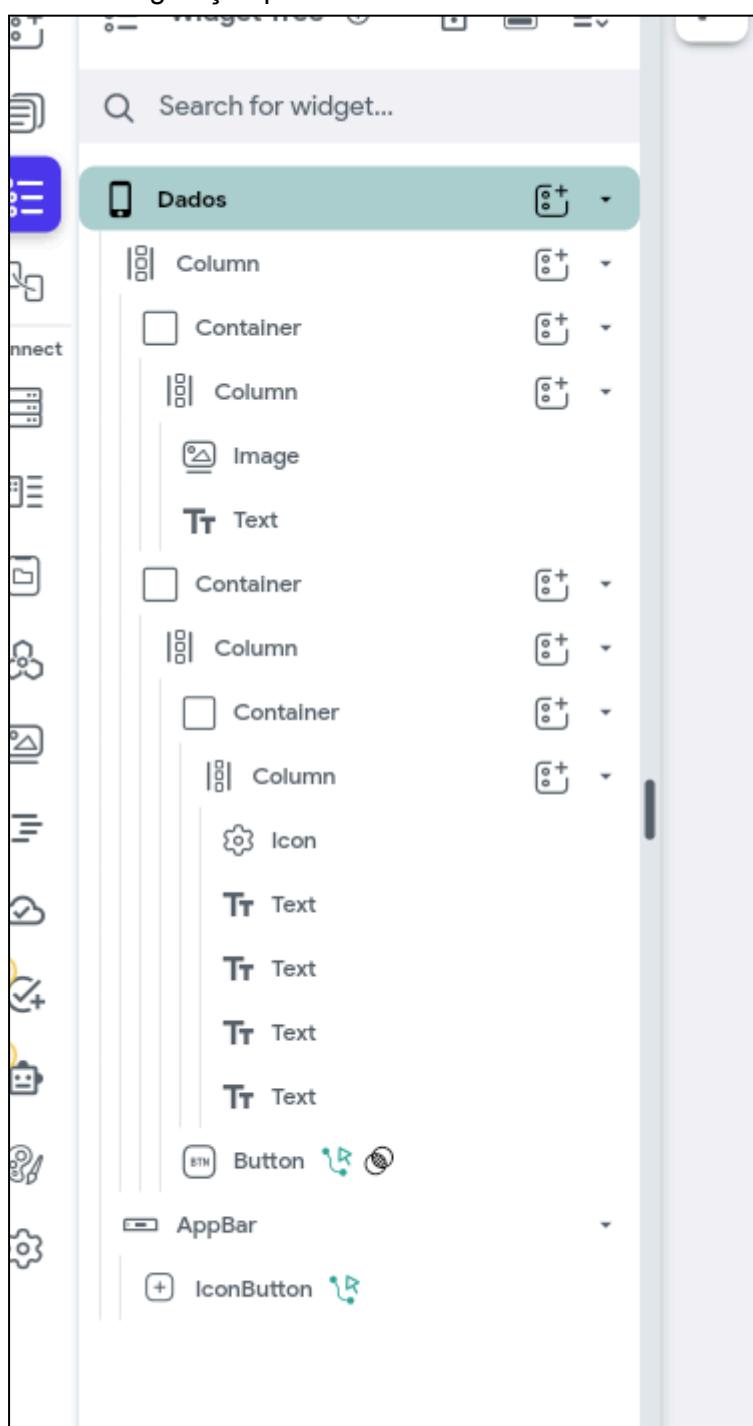
3. Adicionar ““footer” ao “Dados”



3.1 Adicione o *Footer* selecionando o ícone na parte inferior da tela:  , após isso selecione a opção “Nav Settings” destacado de laranja. Após isso, customize.

Obs.: o Footer se “conectam” automaticamente no Flutter Flow.

Obs.: Configuração para fazer o resultado final



Os Dados devem conter o nome, o email, senha e etc.

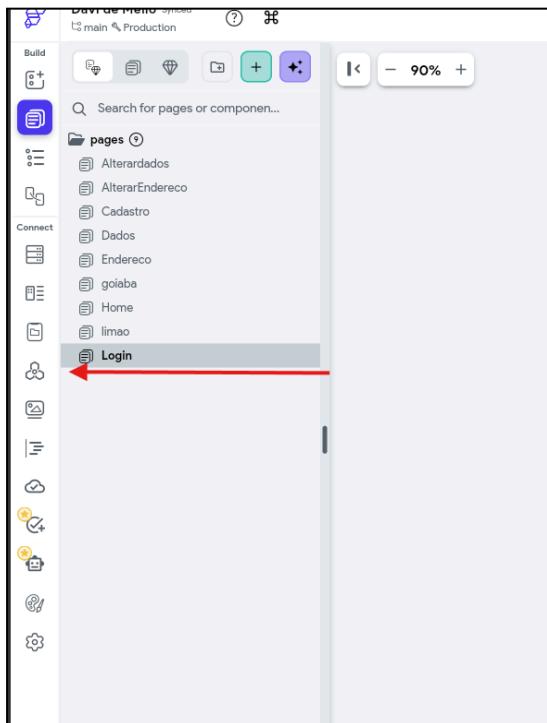
F - API CALL

O que é?

Uma **API Call** é quando um aplicativo envia uma solicitação a um servidor para obter dados ou executar uma ação e recebe uma resposta de volta.

Como Criar uma API call?

Antes de criarmos as API Calls na aplicação, é necessário configurar e publicar a API no **Render**.



Passos:

1. Para criar uma API Call, selecione, no canto esquerdo, o ícone ;
2. Após isso, em API calls selecione + Add;
3. Depois “Create API CALL”. Assim criando uma API Call no seu Flutter flow.

Resultado:

A screenshot of the 'Create API Call' dialog box. It has tabs for 'Call Definition' and 'Response & Test', with 'Call Definition' selected. The 'API Call Name' field is empty. Under 'Method Type', 'GET' is selected. The 'API URL' field contains 'https://api.example.com/posts/{postId}'. The 'Headers' tab is active, showing a 'Headers' section with a '+ Add Header' button. At the bottom right are 'Cancel' and 'Add Call' buttons.

E para salvar a API Call, selecione o botão **Add Call**.

1. API CALL para cadastro de usuário

The screenshot shows the 'Define API Call' interface. The 'Call Definition' tab is selected. The 'API Call Name' field contains 'Cadastrar'. The 'Method Type' is set to 'POST' and the 'API URL' is 'https://thelastdance.onrender.com/usuario/cadastrar?confSenha=string'. The 'Body' tab is active, showing a JSON structure with variables:

```
1 {
2   "nomeCompleto": "nomeCompleto",
3   "email": "email",
4   "senha": "senha",
5   "cpf": "cpf",
6   "dataNascimento": "dataNascimento",
7 }
```

Esta API Call realiza o cadastro de um novo usuário no sistema. A chamada utiliza o método **POST** e envia um corpo JSON contendo os dados necessários para criação do usuário. E utiliza o endpoint:

`https://thelastdance.onrender.com/usuario/cadastrar?confSenha=string`

The screenshot shows the 'Response & Test' interface. The 'Test API Call' button is visible at the top right. The 'Preview and Test' section shows variables and their values:

Name	Value	Include
nomeCompleto	Jhonatan Estervers da Silva	<input checked="" type="checkbox"/>
email	Radaudh@gmail.com	<input checked="" type="checkbox"/>
senha	6543211	<input checked="" type="checkbox"/>
cpf	8285775	<input checked="" type="checkbox"/>
dataNascimento	1999-03-03	<input checked="" type="checkbox"/>
confSenha	6543211	<input checked="" type="checkbox"/>

The 'Test Response' tab is selected, showing the response body as 'null' and a status of 'Status: 200 (Success)'. Below this, the 'Response Type' and 'Data Type' sections are shown.

Para saber se está funcionando basta ir em *Response & Test*, e após selecione o **Test API Call**, e espere o *status* aparecer, caso apareça **200 (success)** a API Call está funcionando e caso apareça diferente a API não está funcionando.

(Obs.: Em Test Response deveria aparecer os valores)

Print da API CALL funcionando no APP:



2. API CALL para Cadastro de endereço

The screenshot shows the "Create API Call" interface. The "Call Definition" tab is active. The "API Call Name" field contains "Cadastro de endereco". The "Method Type" is set to "POST" and the "API URL" is "https://thelastdance.onrender.com/cliente/salvar". The "Body" tab is selected, showing a JSON structure with variables:

```
1 {
2   "logradouro": "logradouro",
3   "localidade": "localidade",
4   "cep": "cep",
5   "numero": "numero",
6   "complemento": "complemento",
7   "bairro": "bairro",
8   "referencia": "referencia"
9 }
```

Esta API Call realiza o cadastro de um endereço associado ao usuário no sistema. A chamada utiliza o método **POST** e envia um corpo JSON contendo as informações completas do endereço (como rua, número, CEP, cidade e estado).
Ela utiliza o endpoint:

<https://thelastdance.onrender.com/cliente/salvar>

The screenshot shows the "Response & Test" interface. The "Test API Call" button is visible. The "Preview and Test" section shows variables and their values:

Name	Value	Include
logradouro	Rua das Palmeiras	<input checked="" type="checkbox"/>
localidade	São Paulo	<input checked="" type="checkbox"/>
cep	04567890	<input checked="" type="checkbox"/>
numero	123	<input checked="" type="checkbox"/>
complemento	Bloco B	<input checked="" type="checkbox"/>
bairro	Vila Olímpia	<input checked="" type="checkbox"/>
referencia	Proximo ao Shopping Vila Olímpia	<input checked="" type="checkbox"/>

The "Test Response" tab is selected, showing the JSON response body:

```
1 {
2   "id": 11,
3   "logradouro": "Rua das Palmeiras",
4   "localidade": "São Paulo",
5   "cep": 4567890,
6   "numero": 123,
7   "complemento": "Bloco B",
8   "bairro": "Vila Olímpia",
9   "referencia": "Proximo ao Shopping Vila Olímpia"
10 }
```

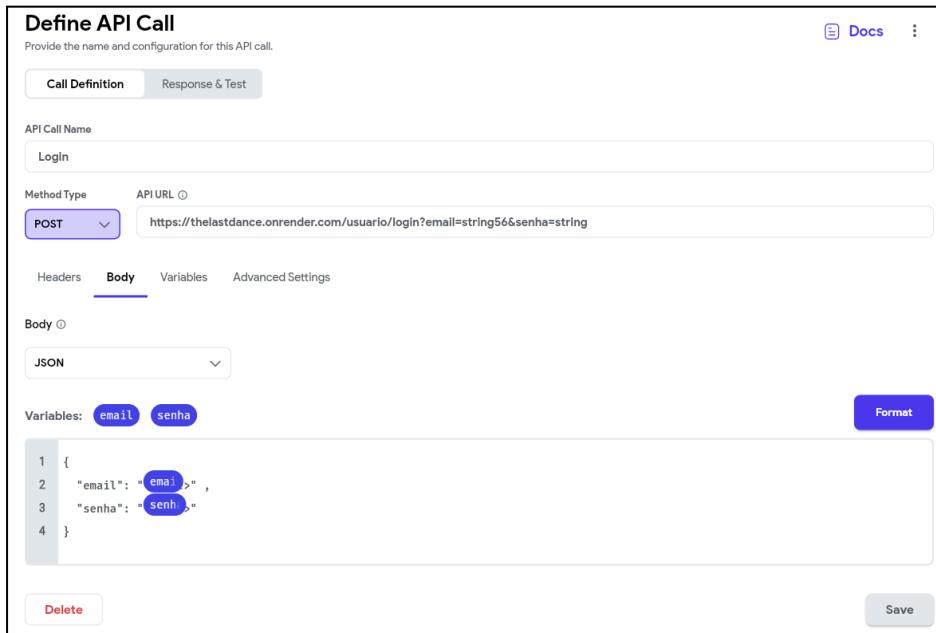
Status: 200 (Success)

Para saber se está funcionando basta ir em *Response & Test*, e após selecione o **Test API Call**, e espere o *status* aparecer, caso apareça **200 (success)** a API Call está funcionando e caso apareça diferente a API não está funcionando ou possui algum problema.

Print da API CALL funcionando no APP:



3. API CALL para Login



The screenshot shows the 'Define API Call' interface. The 'Call Definition' tab is active. The 'API Call Name' is set to 'Login'. The 'Method Type' is 'POST' and the 'API URL' is 'https://thelastdance.onrender.com/usuario/login?email=string56&senha=string'. The 'Body' tab is selected, showing a JSON structure with variables 'email' and 'senha':

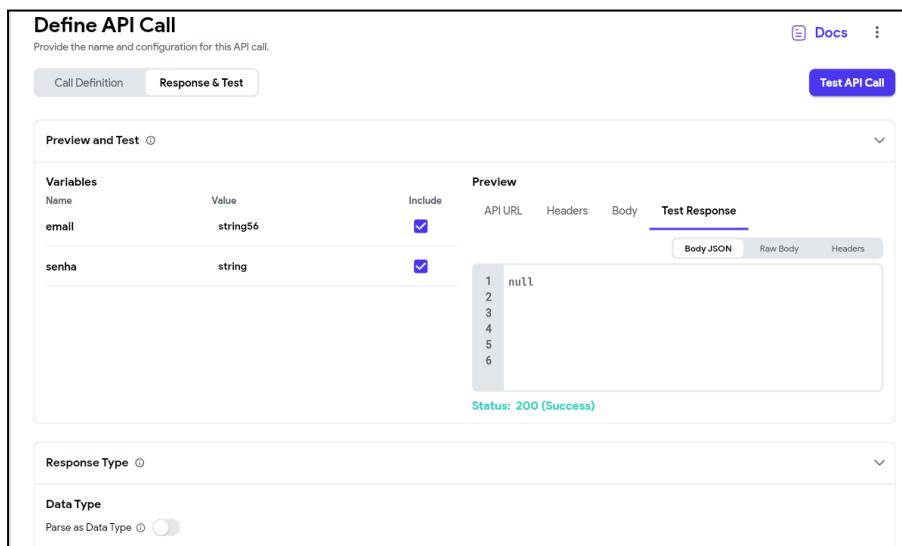
```
1 {
2   "email": "email",
3   "senha": "senha"
4 }
```

Below the body editor are 'Delete' and 'Save' buttons.

Esta API Call realiza o processo de autenticação do usuário no sistema. A chamada utiliza o método **POST** e envia um corpo JSON contendo as credenciais de acesso (como e-mail e senha), que serão validadas pelo backend.

Ela utiliza o endpoint:

```
https://thelastdance.onrender.com/usuario/login?email=string56&senha=string
```



The screenshot shows the 'Response & Test' interface. The 'Test API Call' button is visible. The 'Preview' section shows the response body is 'null'. The status is 'Status: 200 (Success)'.

E para saber se está funcionando basta ir em *Response & Test*, e após selecione o **Test API Call**, e espere o *status* aparecer, caso apareça **200 (success)** a API Call está funcionando e caso apareça diferente a API não está funcionando.

Print da API CALL funcionando no APP:



4. API CALL para Dados do Usuário

Create API Call

Provide the name and configuration for this API call.

Call Definition Response & Test

API Call Name
Dados

Method Type GET API URL https://thelastdance.onrender.com/cliente/listar

Headers Query Parameters Variables Advanced Settings

Headers + Add Header

Cancel Add Call

Esta API Call recupera os dados completos de um usuário no sistema.
A chamada utiliza o método **GET** e retorna um conjunto de informações referentes ao perfil e dados associados do usuário.
E utiliza o endpoint:

https://thelastdance.onrender.com/cliente/listar

Preview and test

Variables

Name	Value

Preview

Include API URL Headers Test Response

Body JSON Raw Body Headers

```
10     "referencia": null
11 },
12 {
13   "id": 2,
14   "logradouro": null,
15   "localidade": null,
16   "cep": null,
17   "numero": null,
18   "complemento": null,
19   "bairro": null,
20   "referencia": null
21 },
22 {
23   "id": 3,
24   "logradouro": null,
25   "localidade": null,
26   "cep": null,
27   "numero": null,
28   "complemento": null,
29   "bairro": null,
30   "referencia": null
31 },
32 {
33   "id": 4,
34   "logradouro": null,
35   "localidade": null,
36   "cep": null,
37   "numero": null,
38   "referencia": null
39 }
```

Status: 200 (Success)

Print da API CALL funcionando no APP:

