

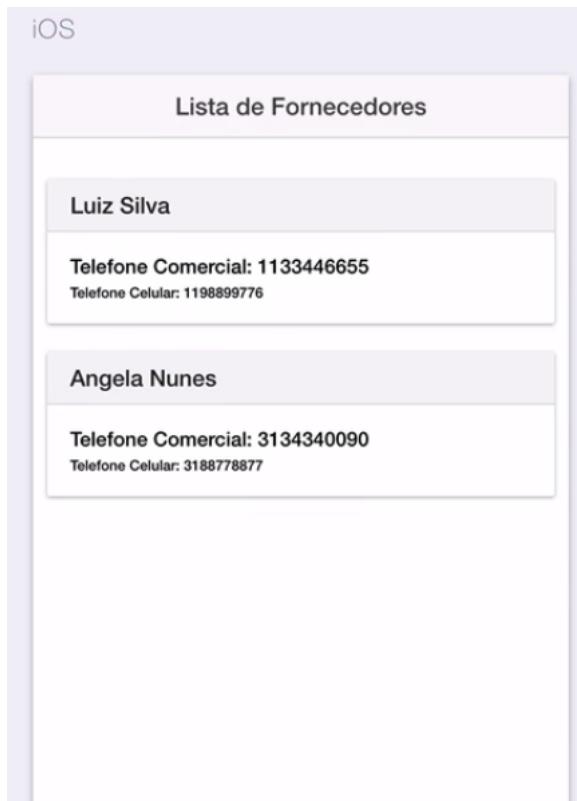
01

## Desenvolvendo nossa aplicação pelo Ionic Creator

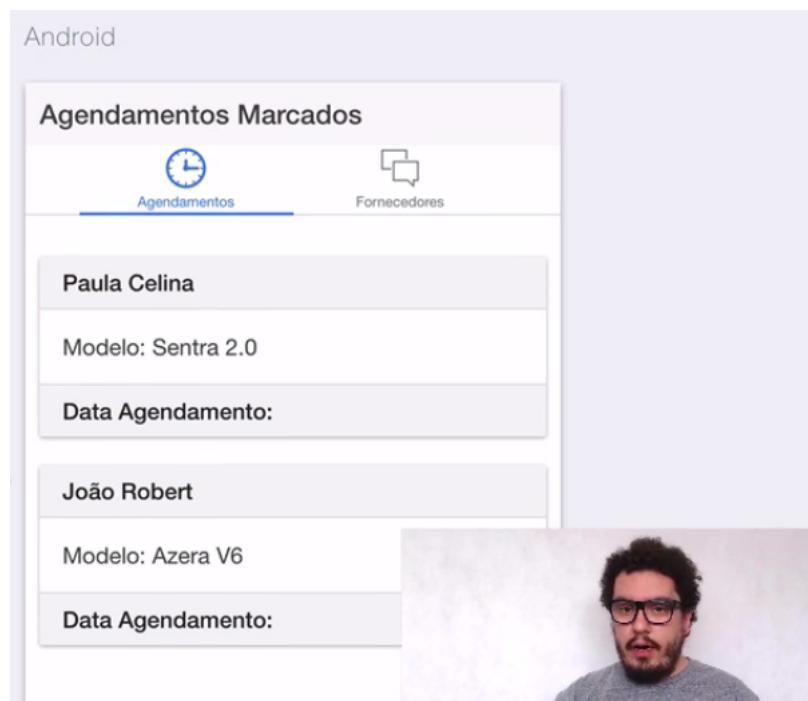
### Transcrição

Nós criamos a app para o usuário. Agora, o cliente quer uma segunda aplicação, que será utilizada pelos vendedores e será mais simples. Ela terá um outro layout, usando abas.

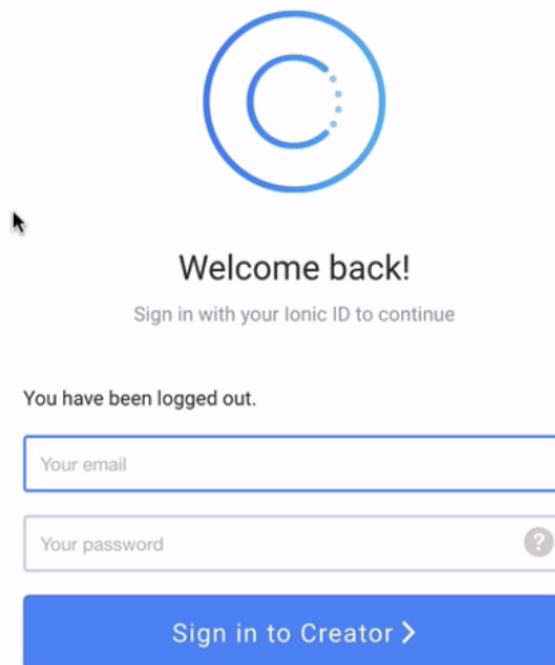
No iOS, as abas ficarão na parte inferior:



Enquanto no Android, as abas ficarão na parte de cima.

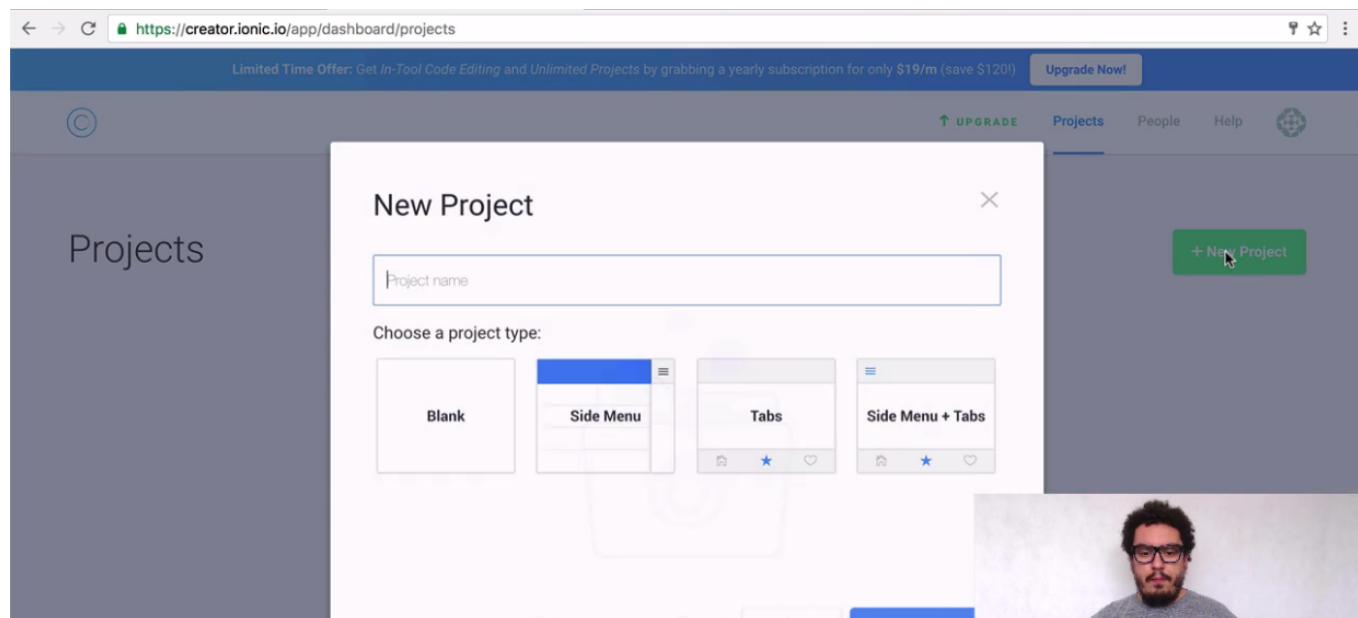


Apenas utilizando as tags do Ionic, conseguimos criar o layout para diferentes plataformas. Nós criaremos a segunda aplicação do zero, mas não pela linha de comando como fizemos anteriormente. Desta vez, usaremos o [Ionic Creator](https://creator.ionic.io/app/login) (<https://creator.ionic.io/app/login>).



O site é dos mesmos criadores do Ionic, para você prototipar as suas aplicações. Atualmente, ele possui uma versão gratuita, em que você consegue montar um projeto - o que nos atende. Mas se você quiser criar mais projetos, você terá que pagar. Mas a maior parte das funcionalidades é gratuita.

Após criarmos a nossa conta, cairemos na tela que é o dashboard principal.



Vamos criar um novo projeto, que chamaremos de `Alura-Gerenciador`. Depois, poderemos selecionar qual template queremos usar. Nós escolheremos o "Tab", que contém abas. Clicaremos no botão `Create Project` e ele nos dará um template pronto, que poderemos customizar.

The screenshot shows the Ionic Creator interface. On the left, there's a sidebar with 'Pages' (containing 'Camera Tab Default Page', 'Cart Tab Default Page', 'Cloud Tab Default Page', and 'Tabs Controller') and 'Components' (containing 'Hanging' and 'Paragraph'). The main area shows a preview of an iPhone device with a 'TABS CONTROLLER' component. On the right, there's a 'Page' configuration panel with fields for 'Title' (set to 'Tabs Controller'), 'Routing URL' (set to '/page1'), and 'Route Parameters' (with an 'Add' button). Below these are fields for 'Background' and 'Color' (set to 'Default'), and a 'Upload Image...' button.

Está configurado para iOS, mas conseguimos mudar para Android.

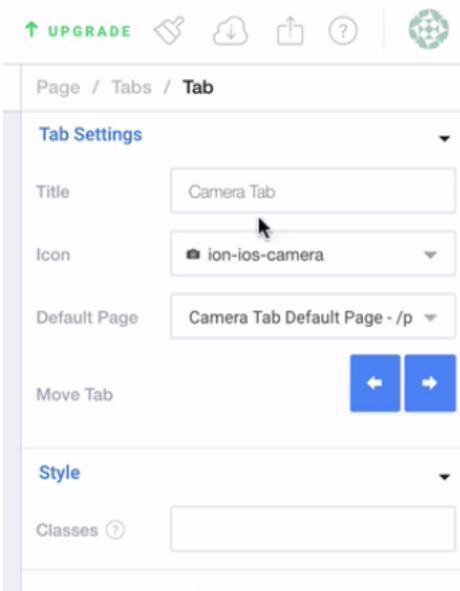
The screenshot shows the 'Device' dropdown menu in the Ionic Creator interface. The 'iPhone' option is selected and highlighted with a red box. Other options like 'Android' and 'Web' are also visible in the dropdown.

E então, as abas já surgirão no topo do aplicativo. À esquerda, temos uma parte em que podemos escolher as páginas e os componentes.

The screenshot shows the 'Pages' section in the Ionic Creator interface. The 'Tabs Controller' page is expanded, showing its 'tabs' component and three tabs: 'Camera Tab', 'Cart Tab', and 'Cloud Tab'. The 'Cloud Tab' tab is selected.

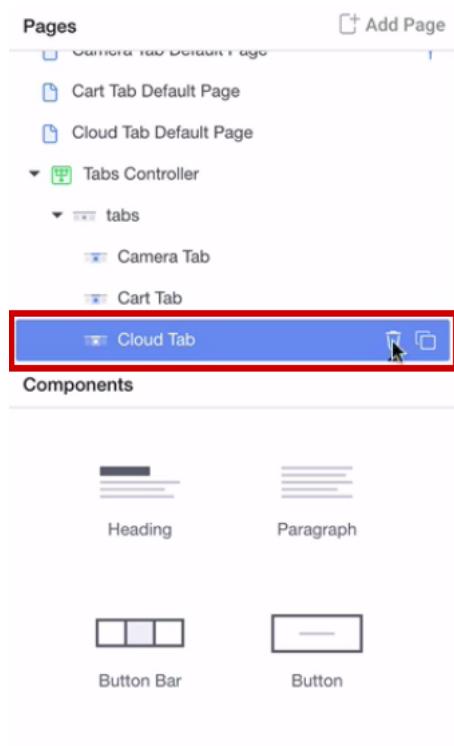
Nos componentes, basta clicar e arrastar para o aplicativo. Então, é interessante você já ter um conhecimento prévio do Ionic para utilizar o Ionic Creator. Por exemplo, no aplicativo anterior, nós construimos um card manualmente. Agora, poderemos adicionar o componente do card.

À direita, podemos configurar as abas.



Temos a opção de selecionar os componentes da *Tab*.

Vamos começar a customizar a app. Atualmente, temos três abas, mas queremos apenas dois. Vamos remover um na parte de páginas. Seleccionaremos a aba que não queremos mais e clicaremos na lixeira.



Agora teremos apenas duas abas. Na seção de abas, configuraremos que a primeira se chamará "Agendamentos" e o ícone será "Clock".

Device: Android Phone

100%

Page / Tabs / Tab

Tab Settings

Title: Agendamentos

Icon: ion-clock

Default Page: Camera Tab Default Page - /p

Move Tab: [left, right]

Style: [dropdown]

Classes: [input]

A segunda, receberá o nome de "Fornecedores" e o ícone será "ion-android-chat".

Device: Android Phone

100%

Page / Tabs / Tab

Tab Settings

Title: Fornecedores

Icon: ion-android-chat

Default Page: Cart Tab Default Page - /page

Move Tab: [left, right]

Style: [dropdown]

Classes: [input]

Vamos excluir uma página que era referente à Tab que já apagamos. Vamos ajustar os títulos das outras páginas para "Agendamentos" e a URL para `/agendamentos`. Renomearemos a segunda página também, no caso para "Fornecedores" e a rota para `"/fornecedores"`.

https://creator.ionic.io/app/designer/6f46a562f938

Alura-Gerenciador Saving...

Pages

Agendamentos

Fornecedores

Tabs Controller

Device: Android Phone

100%

Page

Title: Fornecedores

Routing URL: /fornecedores

State (sref): tabsController.fornecedores

Route Parameters: [PRO]

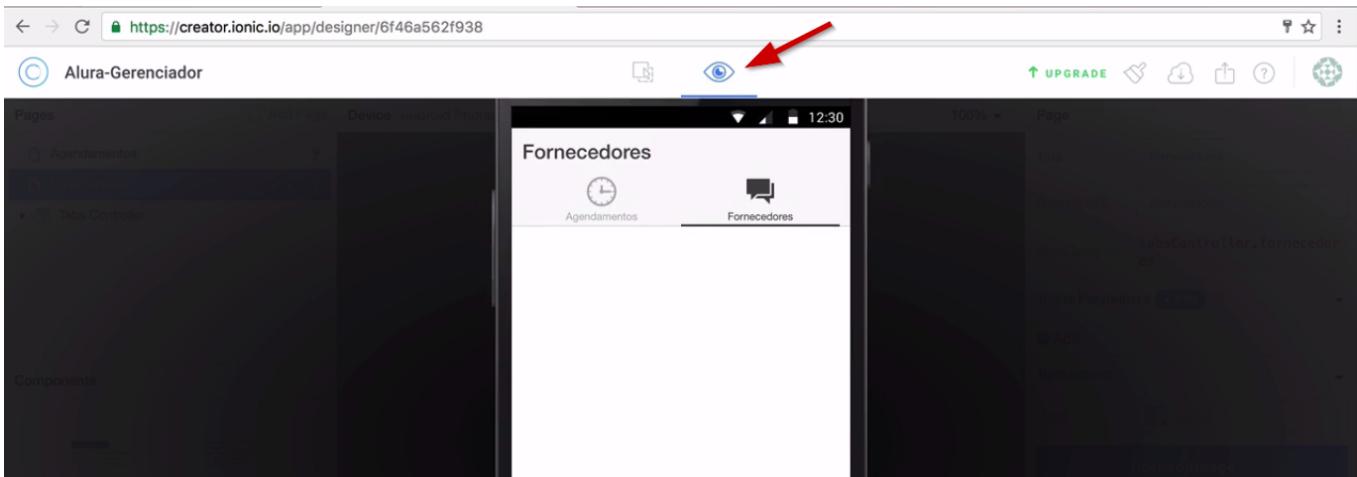
Add

Background

Color: Default

Upload Image...

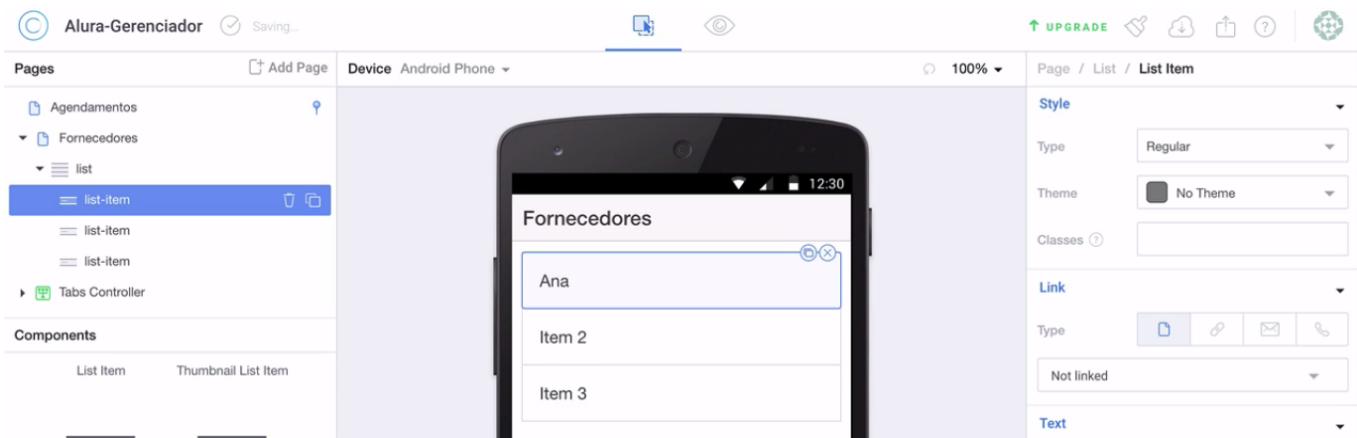
E temos a opção de ver como está ficando o nosso aplicativo.



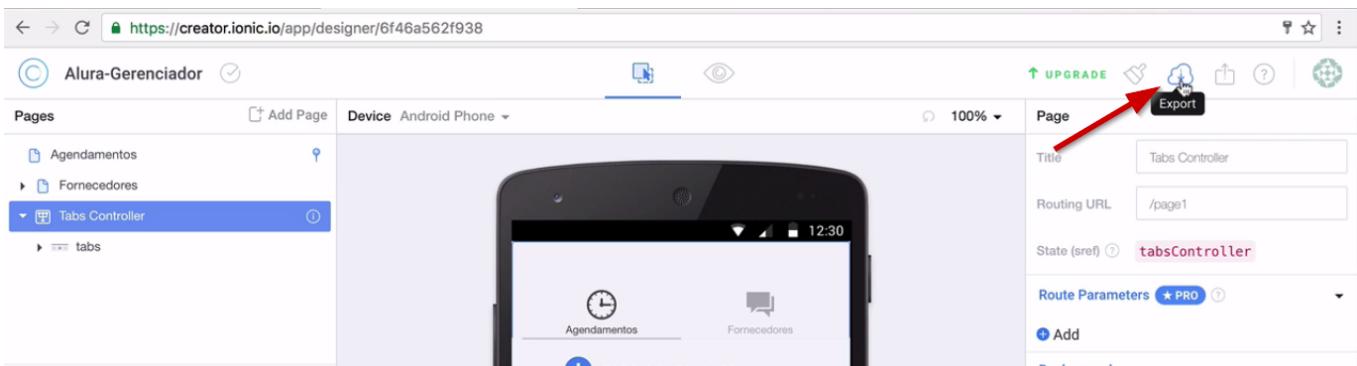
Podemos selecionar o dispositivo, por exemplo, iPhone ou um iPad.

A seguir, adicionaremos componente nestas páginas, começando pelas listas. Podemos configurar a parte de estilos. Temos ainda uma parte "Pro", com componentes que precisam ser pagos.

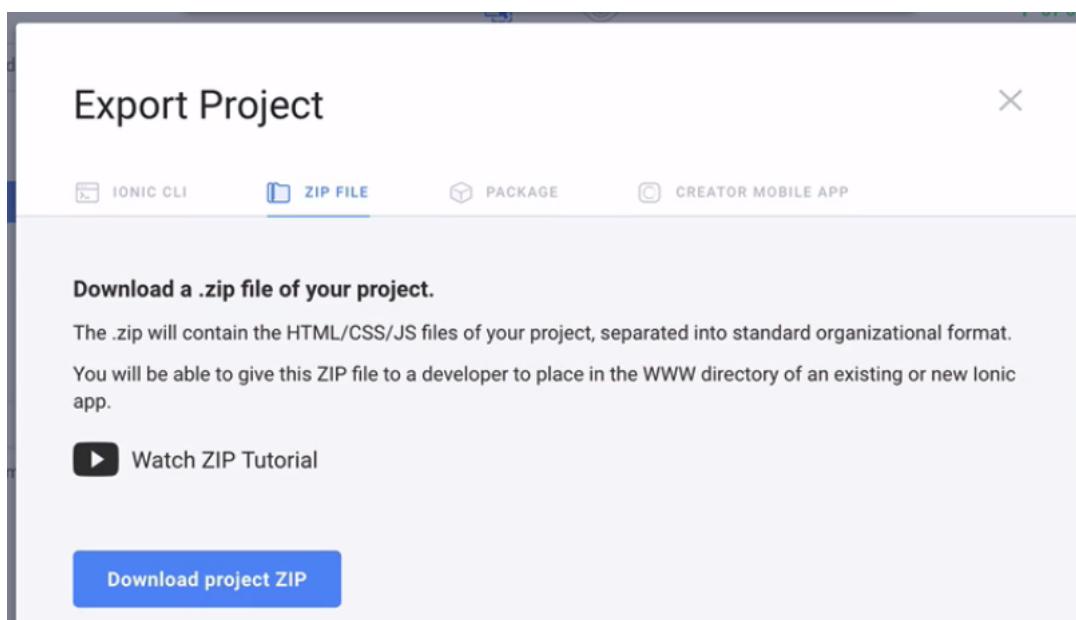
Nós temos a opção de alterar os textos dos itens. Mas nós não usaremos os textos fixos.



Já criamos o protótipo, agora, quero programar em cima dele. Para isso, iremos exportá-lo.

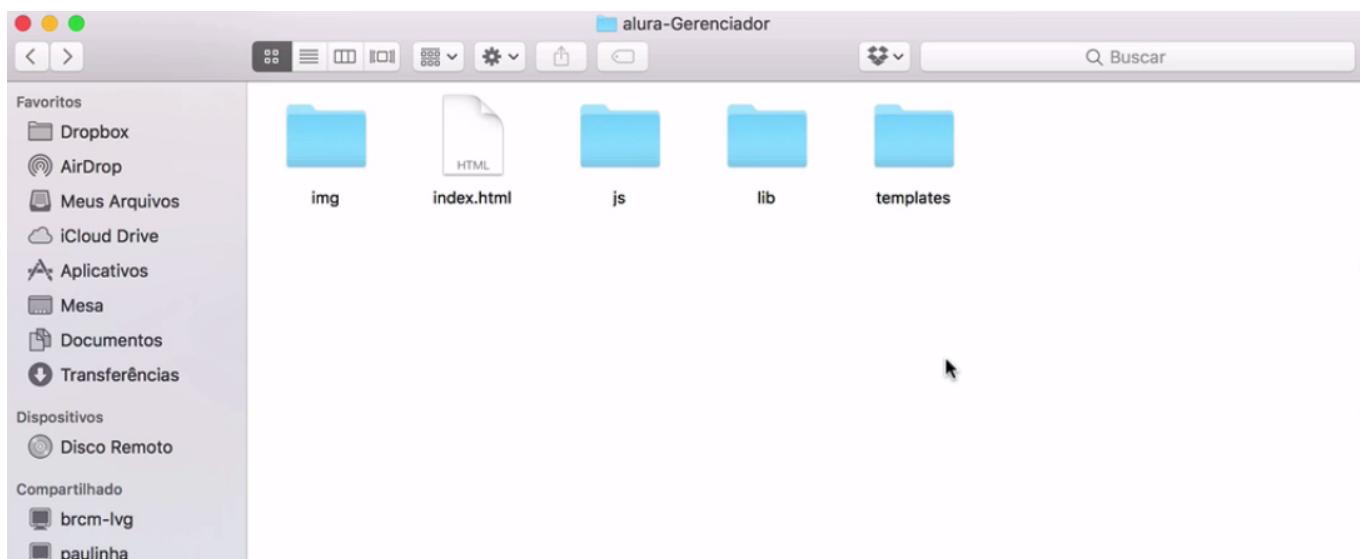


Teremos quatro opções das formas que podemos exportar o projeto. Até o momento em que gravamos a aula, o "Ionic CLI" não estava habilitado.



Mas ainda temos as opções "Zip File", "Package" e "Creator Mobile App" - ressaltando que as duas últimas são pagas. Nós usaremos a opção "Zip File".

Após fazermos o download do projeto, até termos o arquivo `alura-Gerenciador.zip`. Vamos extrair o arquivo, que não estará completo.



Nós vamos construir aplicação do zero, para substituir os arquivos gerados, por estes. No Terminal, vamos renomear o arquivo:

```
Alura-gerenciador mv alura-Gerenciador alura-Gerenciador-arquivos
```

Agora, podemos remover o arquivo `.zip`:

```
Alura-gerenciador rm alura-Gerenciador.zip
```

```
Alura-gerenciador ls
alura-Gerenciador alura-Gerenciador.zip
Alura-gerenciador mv alura-Gerenciador alura-Gerenciador-arquivos
Alura-gerenciador ls
alura-Gerenciador-arquivos alura-Gerenciador.zip
Alura-gerenciador rm alura-Gerenciador.zip
Alura-gerenciador ls
alura-Gerenciador-arquivos
Alura-gerenciador
```

Então, o arquivo se chama `alura-Gerenciador-arquivos` e excluímos o ZIP.

Agora, criaremos uma aplicação em branco e vamos substituir os arquivos.

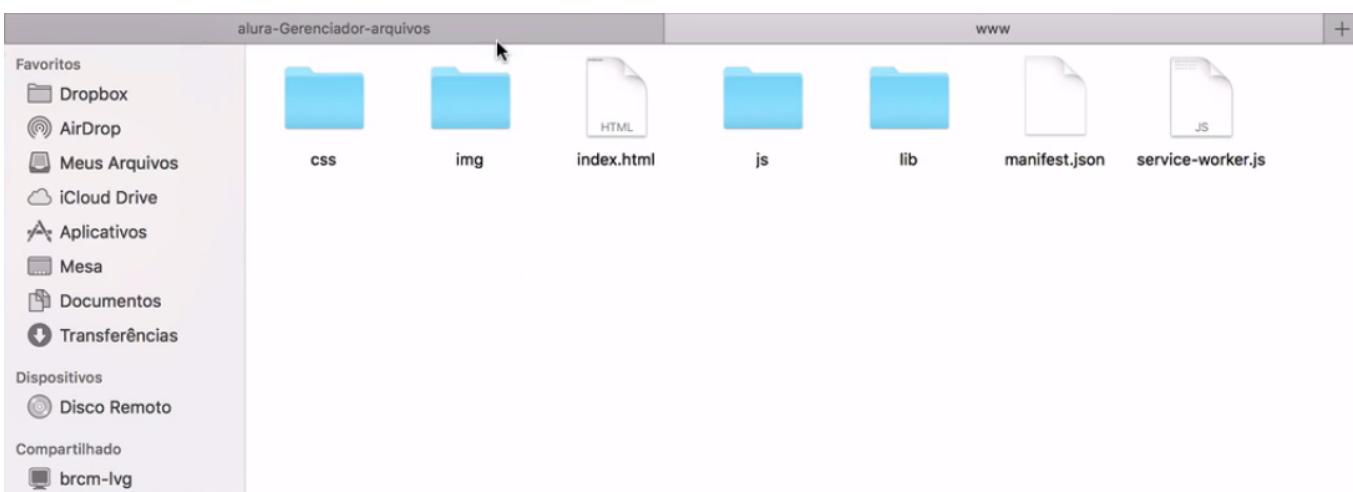
```
Alura-gerenciador ionic start Alura-Gerenciador blank
```

```
Alura-gerenciador — ionic start Alura-Gerenciador blank — ionic — node × ionic TERM_PROGRAM=Apple_Terminal SHELL=/bin/zsh TERM=xterm-256color TMPDIR=/var/folders/79/gb8t459s0lx40frv...
Alura-gerenciador ls
alura-Gerenciador-arquivos
Alura-gerenciador ionic start Alura-Gerenciador blank
Creating Ionic app in folder /Users/lazaro/Desktop/Alura-gerenciador/Alura-Gerenciador based on blank project
Downloading: https://github.com/driftyco/ionic-app-base/archive/master.zip
[=====] 100% 0.0s
Downloading: https://github.com/driftyco/ionic-starter-blank/archive/master.zip
[=====] 100% 0.0s
Updated the hooks directory to have execute permissions
Update Config.xml
Initializing cordova project
```

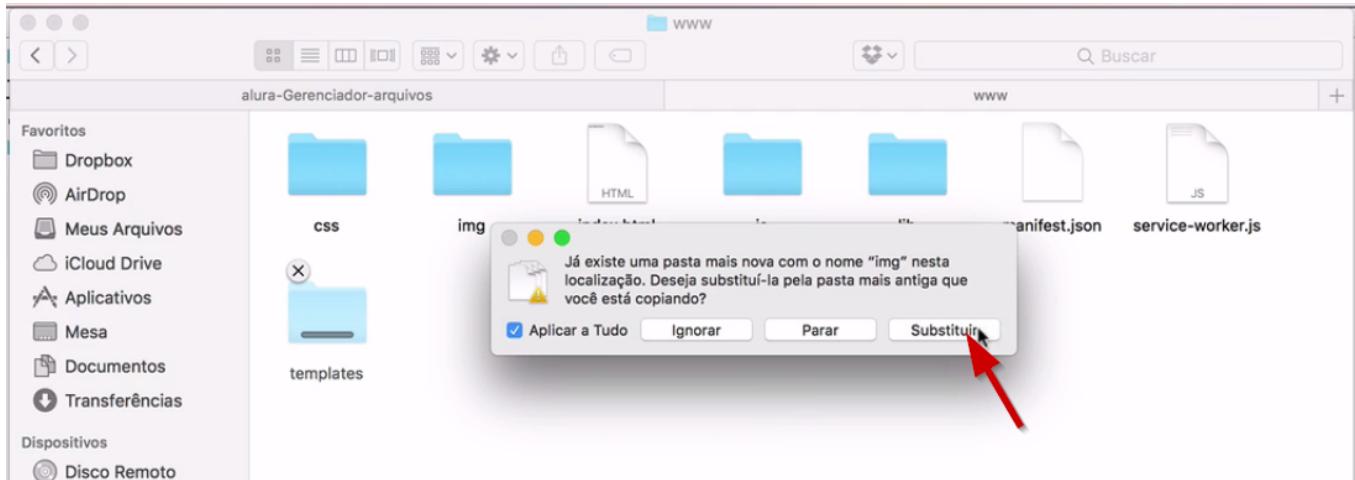
Ele irá criar a aplicação com outros arquivos.

```
Alura-gerenciador ls
bower.json gulpfile.js ionic.project platforms scss
config.xml hooks package.json plugins www
Alura-gerenciador
```

Em seguida, vamos copiaremos os arquivos do `alura-Gerenciador-arquivos`, que foram dados pelo Ionic Creator.



E vamos colar na pasta `www` do `alura-Gerenciador`. Iremos substituir todos os arquivos:



Agora, vamos subir a aplicação no Terminal para testar:

```
ionic serve -l
```

A aplicação estará rodando.

Lembrando que não temos apenas o componente de lista, mas vários outros. Só o componente que nos permite usar o Google Maps é pago.