



*framework*  
**IONIC**

**3 HISTÓRICO**  
Histórico e principais características do framework


**7 INSTALAÇÃO**  
Processo de instalação da IDE para se trabalhar com o respectivo framework.

**21 APLICAÇÕES**

**TRIÂNGULO..... 21-26**  
Calcular a área do triângulo cuja base e altura devem ser digitadas pelo usuário.

**GPS..... 27-33**  
Descobrir a latitude/longitude da localização e exibir essa informação na tela.

**CRUD..... 34-42**  
Permite salvar, consultar, deletar e alterar dados em um banco de dados.



# HISTÓRICO

## IONIC 1: O COMEÇO

Em 2013 foi publicada a primeira versão do Ionic, construída em **AngularJS** e **Cordova**. Sendo construído no AngularJS, ele herda as virtudes e defeitos desta estrutura. Por outro lado, graças a Cordova, teve o **wrapeo nativo** e sua infinidade de plug-ins para fornecer certas funcionalidades nativas.

**Ionic** é uma revolução na época, permitindo facilmente o que o mercado estava procurando: uma maneira fácil de criar aplicativos híbridos.

## IONIC 2: A GRANDE ATUALIZAÇÃO

Em 2016 a equipe Angular atualizou completamente o seu **framework**, sendo a mudança tão grande que foi considerado um novo framework em vez de uma atualização, assim nasceu o **Angular**. Assim, o Ionic é forçado a atualizar para usar a nova estrutura. Devido as mudanças drásticas na estrutura base, a atualização é um grande esforço para a equipe do Ionic. Da mão do Angular vem o uso do **Typescript** e modularização, o que permite ao Ionic separar sua estrutura em partes: core, angular, nativa, etc.

## IONIC 3: A PAUSA

Em vez de repetir o caso do AngularJS, o Angular decide adotar o **semver e um ciclo de lançamento principal** de seis meses e **LTS semestral**. Isso permite ter um framework em constante evolução, com um período de seis meses entre as mudanças de quebra, a primeira versão publicada com este modelo foi o Angular 4, pulando a terceira para padronizar o versionamento interno de suas partes.

A Ionic, portanto, lança sua versão 3 nesta versão LTS do Angular e decide que suas atualizações serão limitadas à versão LTS do Angular. Em sua versão 3, o Ionic foca em melhorar sua própria estrutura e continuar com sua integração com o Angular.

- **IONIC-CLI**

Melhorias consideráveis no Ionic cli, para permitir de forma simples configurar, desenvolver e compilar os projetos e abstrair o desenvolvedor dos comandos internos: angular-cli, Webpack ou Cordova cli

- **IONIC-NATIVE**

Na parte nativa, oferecendo suas próprias classes que cobrem plugins Cordova para trabalhar mais facilmente com eles.

- **MELHOR INTEGRAÇÃO ANGULAR**

- Integração com Roteamento Angular;
- Definição das próprias anotações.

## **MUDANÇA DE CURSO**

A revolução híbrida mundial.

Desde a publicação de Ionic em 2013, o mundo híbrido passou por algumas mudanças:

- **O WPA:** o surgimento do WPA, aplicativos web que podem ser instalados diretamente no dispositivo sem a necessidade de distribuição na loja, fez com que o wrapeo não seja a única forma de publicar aplicativos web.
- **A maturidade dos padrões web:** certas funcionalidades que antes só estavam disponíveis em aplicativos híbridos, como notificações PUSH, uso da câmera ou geolocalização, agora estão disponíveis no padrão web e qualquer site sem a necessidade de instalação de um aplicativo pode usá-los.
- **Frameworks verdadeiramente híbridos:** aparece uma série de frameworks como **React Native** e **Native Script**, que mudam o conceito de aplicativos híbridos. Em vez de oferecer componentes visualmente nativos, o que essas estruturas fazem é identificar certos componentes **HTML** que podem ser substituídos por componentes nativos. O exemplo mais típico são listas, em vez de emular uma lista nativa na web, essas estruturas constroem uma lista nativa com conteúdo da web, mas o usuário interage com uma lista nativa.
- **Unificação de design entre plataformas:** é cada vez mais comum que um **aplicativo iOS** e **Android** sejam visualmente idênticos, fazendo com que a funcionalidade central do Ionic: componentes visualmente nativos alinhados com cada plataforma, fiquem em segundo plano.

## IDES

A IDE, conhecido no português como Ambiente de Desenvolvimento Integrado, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software. Ele tem o objetivo de agilizar este processo.

Algumas características e ferramentas encontradas nos IDEs são:

- **Editor** = edita o código-fonte do programa escrito na(s) linguagem(ns) suportada(s) pela IDE;
- **Compilador (compiler)** = compila o código-fonte do programa, editado em uma linguagem específica e a transforma em linguagem de máquina;
- **Linker** = liga (linka) os vários "pedaços" de código-fonte, compilados em linguagem de máquina, em um programa executável que pode ser executado em um computador ou outro dispositivo computacional;
- **Depurador (debugger)** = auxilia no processo de encontrar e corrigir defeitos no código-fonte do programa, na tentativa de aprimorar a qualidade de software;
- **Modelagem de dados (modeling)** = criação do modelo de classes, objetos, interfaces, associações e interações dos artefatos envolvidos no software com o objetivo de solucionar as necessidades-alvo do software final;
- **Geração de código** = característica mais explorada em Ferramentas CASE, a geração de código também é encontrada em IDEs, contudo com um escopo mais direcionado a templates de código comumente utilizados para solucionar problemas rotineiros. Todavia, em conjunto com ferramentas de modelagem, a geração pode gerar praticamente todo o código-fonte do programa com base no modelo proposto, tornando muito mais rápido o processo de desenvolvimento e distribuição do software;
- **Distribuição (deploy)** = auxilia no processo de criação do instalador do software, ou outra forma de distribuição, seja discos ou via internet;
- **Testes Automatizados (automated tests)** = realiza testes no software de forma automatizada, com base em scripts ou programas de testes previamente especificados, gerando um relatório, assim auxiliando na análise do impacto das alterações no código-fonte. Ferramentas deste tipo mais comuns no mercado são chamadas robôs de testes;

Existem vários exemplos de IDEs existentes atualmente em que se é possível utilizar Ionic, são eles:

- **Visual Studio** = é um ambiente de desenvolvimento integrado da Microsoft para desenvolvimento de software especialmente dedicado ao .NET Framework e às linguagens Visual Basic, C, C++, C# e F#;
- **Sublime Text** = é um editor de código-fonte multi-plataforma e shareware com uma interface de programação de aplicativos para a linguagem Python;
- **Atom** = é um editor de texto de código aberto disponível para as plataformas Linux, macOS e Microsoft Windows, desenvolvido pelo GitHub sob a licença MIT. A ferramenta foi criada usando Electron e tecnologias web como HTML, JavaScript e CSS. Permite a instalação de extensões desenvolvidos com Node.js;
- **WebStorm** = é um IDE leve e poderoso, perfeitamente equipado para o complexo desenvolvimento do lado do cliente e desenvolvimento do lado do servidor com Node.js;
- **Backers** = é um editor de código-fonte com foco principal no desenvolvimento web. Criado pela Adobe Systems, é um software livre e de código aberto licenciado sob a Licença MIT, e atualmente é mantido no GitHub pela Adobe e outros desenvolvedores de código aberto. Está escrito em JavaScript, HTML e CSS;
- **Textmates** = é um editor de texto de GUI de uso geral para o macOS criado por Allan Odgaard. O TextMate apresenta personalizações declarativas, guias para documentos abertos, macros graváveis, seções dobráveis, trechos, integração de shell e um sistema extensível de pacotes. O Visual Studio é um exemplo de um IDE mais conhecido mundialmente. Por conta disso, foi o escolhido pelo grupo para ser trabalhado.

**WIKIPEDIA, AMBIENTE DE DESENVOLVIMENTO INTEGRADO. DISPONÍVEL EM:**  
"HTTPS://PT.M.WIKIPEDIA.ORG/WIKI/AMBIENTE\_DE\_DESENVOLVIMENTO\_INTEGRADO#:~:TEXT=IDE%2C%20DO%20INGL%C3%AAS%20INTEGRATED%20DEVELOPMENT,OBJETIVO%20DE%20AGILIZAR%20ESTE%20PROCESSO"

**WIKIPEDIA, MICROSOFT VISUAL STUDIO. DISPONÍVEL EM:**  
"HTTPS://PT.M.WIKIPEDIA.ORG/WIKI/MICROSOFT\_VISUAL\_STUDIO"

**WIKIPEDIA, TEXTMATE. DISPONÍVEL EM:**  
"HTTPS://EN.M.WIKIPEDIA.ORG/WIKI/TEXTMATE"

**WIKIPEDIA, ATOM (EDITOR DE TEXTO). DISPONÍVEL EM:**  
"HTTPS://PT.M.WIKIPEDIA.ORG/WIKI/ATOM\_(EDITOR\_DE\_TEXTO)"

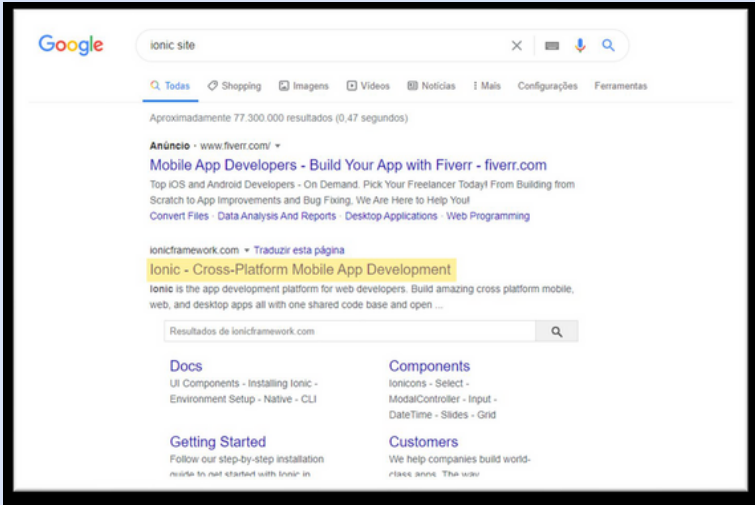
**WIKIPEDIA, SUBLIMETEXT. DISPONÍVEL EM:**  
"HTTPS://PT.M.WIKIPEDIA.ORG/WIKI/SUBLIME\_TEXT"

**CRÉDITOS POR ILUSTRAÇÃO:** HTTPS://BR.FREEPIK.COM

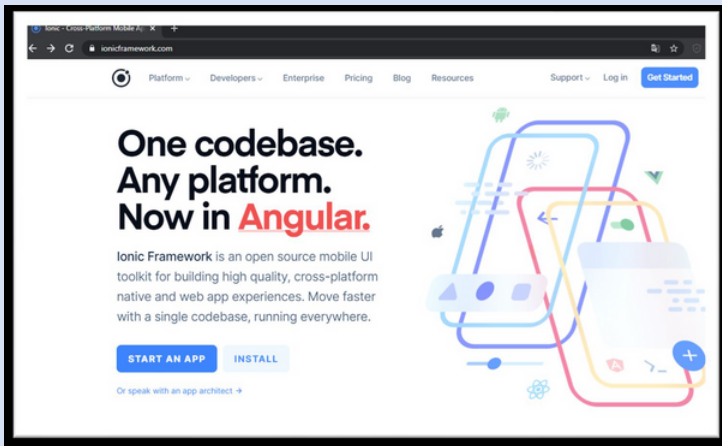
# INSTALAÇÃO

## TUTORIAL DE INSTALAÇÃO

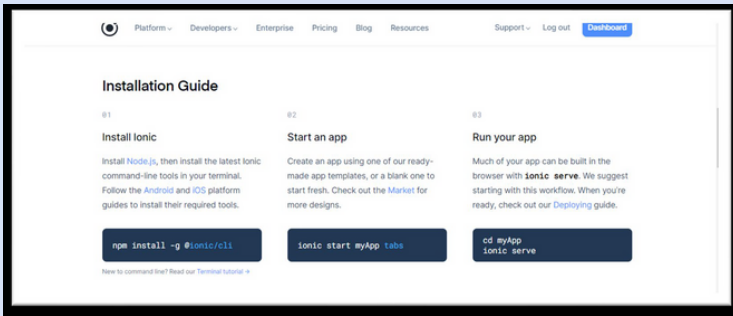
No seu navegador, buque por IONIC, acessando o site conforme mostra a imagem abaixo:



Em seguida, clique em Install (Instalar, caso sua página esteja sendo traduzida)

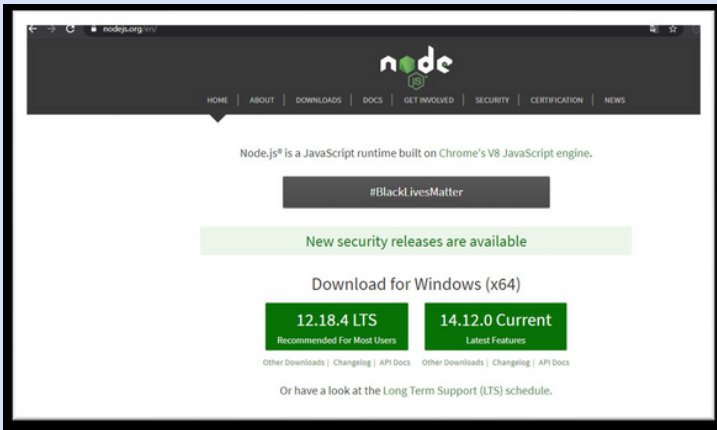


Role para baixo seu cursor na nova aba que foi aberta, até encontrar o seguinte tópico:

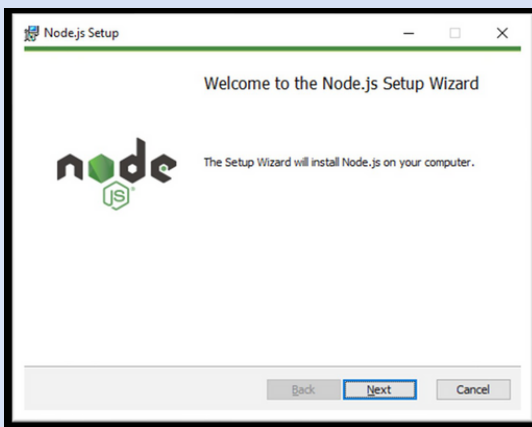


Clique em "Node.js" e você será redirecionado para um novo site, onde você terá duas opções de download, clique na que achar melhor.

- LTS = são aquelas cujos pacotes, aplicações e principais recursos, foram testados de maneira mais extensiva;
- Current = mais atual.



Após escolher o local de destino, execute o arquivo recém baixado. Quando aparecer a tela inicial (Welcome) prossiga com Next.

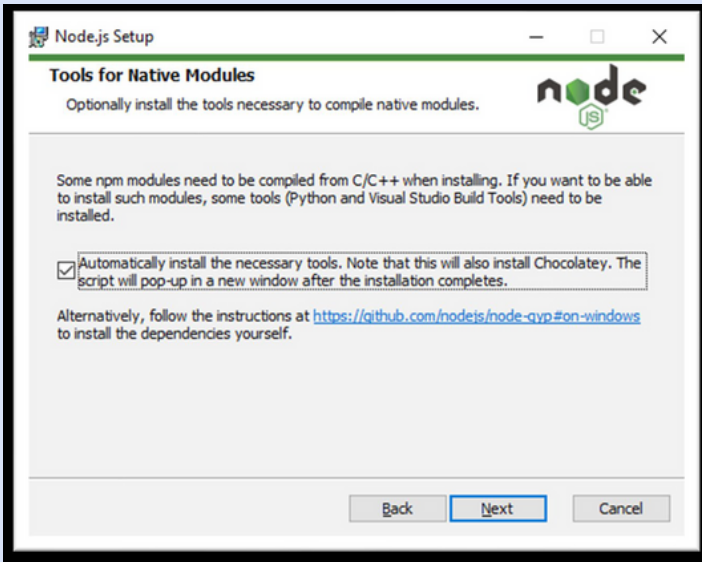




Marque a caixa de seleção aceitando os termos de licença e prossiga com Next.

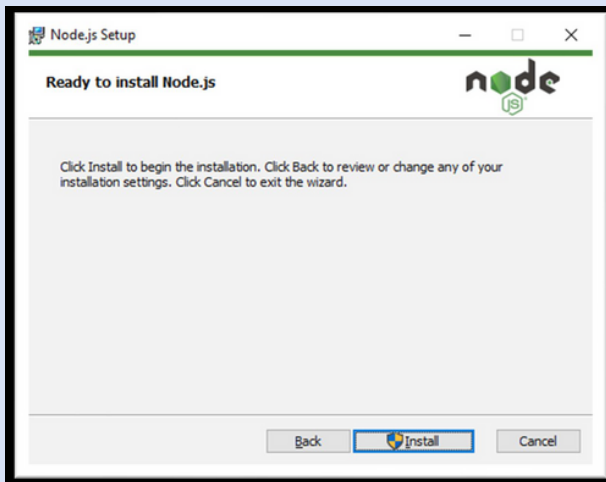
Caso queira mudar o local onde o arquivo será salvo, clique em Change...” e escolha o novo local, caso contrário, prossiga em Next nas duas etapas que seguem.

Marque a caixa de seleção para que a instalação das ferramentas necessárias seja feita e prossiga com Next.

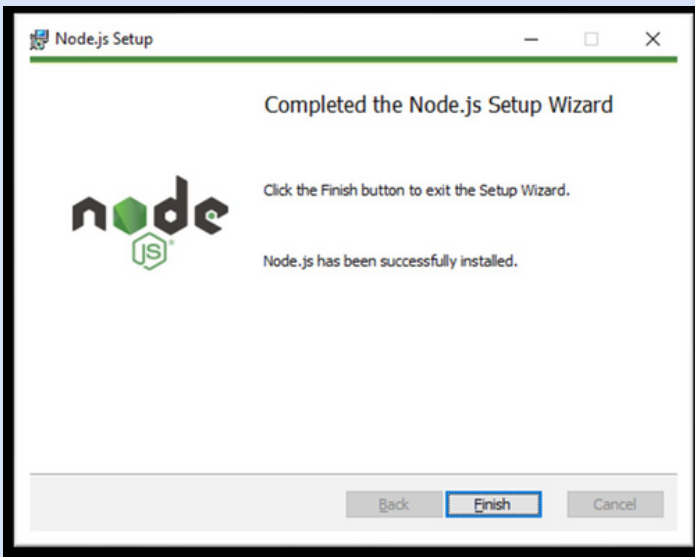
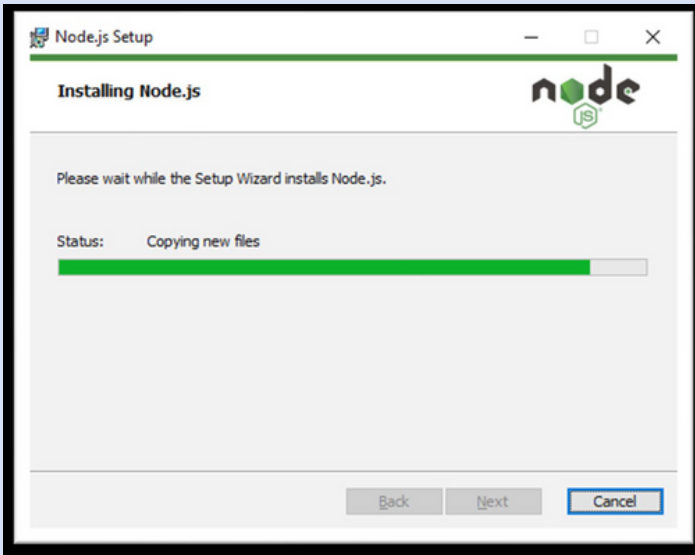


Prossiga com Install.

Logo em seguida será feito um pedido de autorização, aceite apertando em sim.



Aguarde a instalação, apertando em Finish quando estiver completa.



Um Prompt de Comando aparecerá, faça como é pedido e pressione qualquer tecla.

```
cmd: Selecionar install Additional Tools for Nodejs
-----
Tools for Node.js Native Modules Installation Script
-----

This script will install Python and the Visual Studio Build Tools, necessary
to compile Node.js native modules. Note that Chocolatey and required Windows
updates will also be installed.

This will require about 3 Gb of free disk space, plus any space necessary to
install Windows updates. This will take a while to run.

Please close all open programs for the duration of the installation. If the
installation fails, please ensure Windows is fully updated, reboot your
computer and try to run this again. This script can be found in the
Start menu under Node.js.

You can close this window to stop now. Detailed instructions to install these
tools manually are available at https://github.com/nodejs/node-64#on-windows

Pressione qualquer tecla para continuar. . . .
```

Um pedido de autorização para que o Windows ShellPower seja aberto aparecerá, confirme em sim.

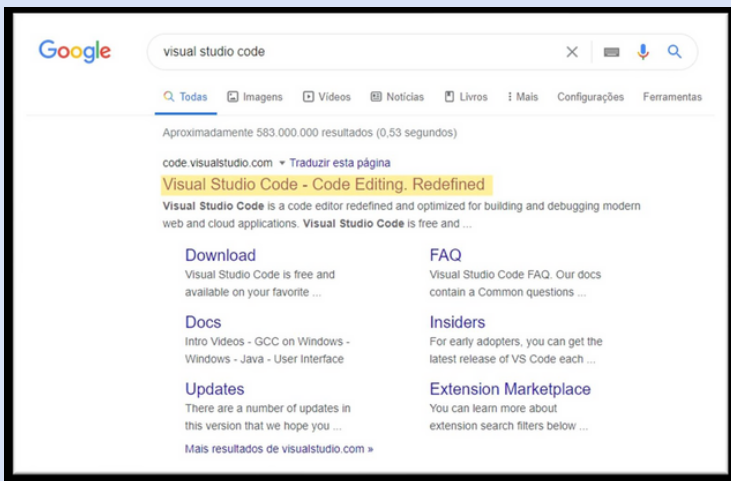
Aguarde o processo ser finalizado completamente para poder fechar a janela do shell.

```
Selecionar Administração Windows PowerShell
-----
Getting latest version of the Chocolatey package for download.
Getting Chocolatey from https://chocolatey.org/api/v2/package/chocolatey/0.10.15.
```

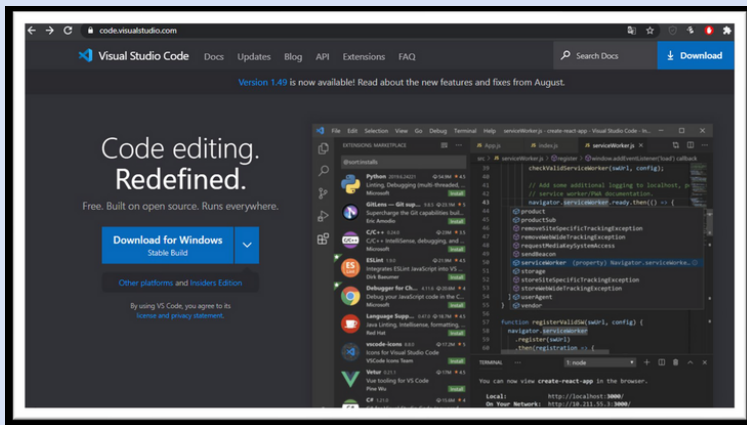


# INSTALAÇÃO DE UMA IDE (AMBIENTE DE DESENVOLVIMENTO)

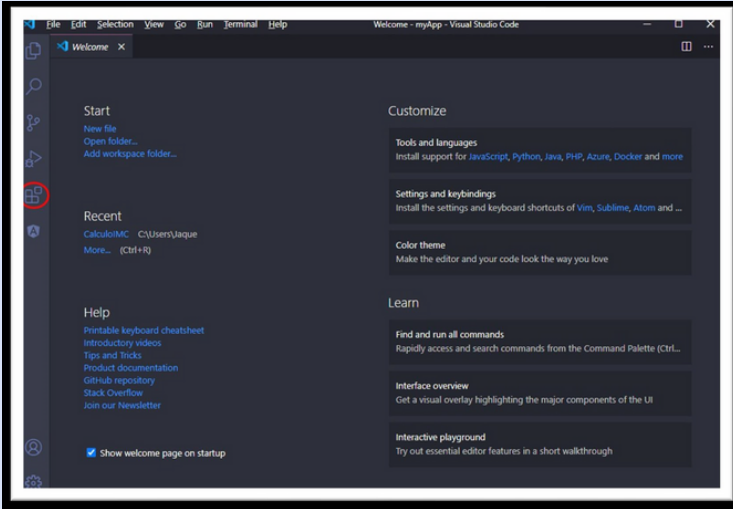
Busque em seu navegador por “Visual Studio Code” e acesse o site, conforme mostra a imagem abaixo:



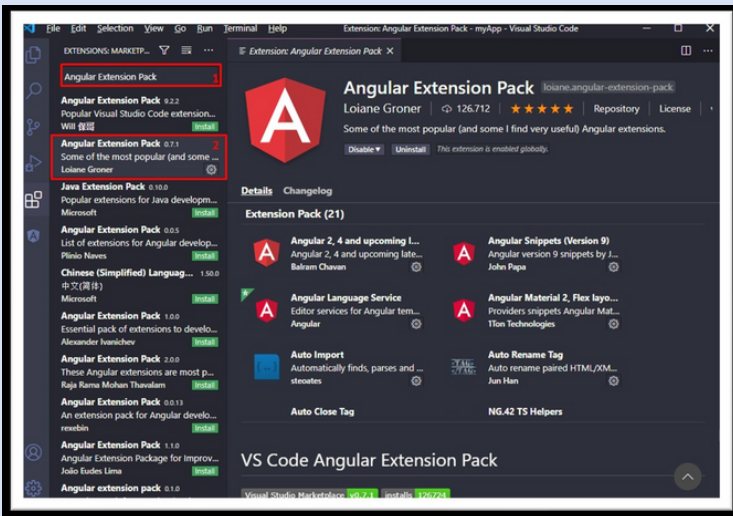
Clique em Download for Windows ou clique na seta ao lado para escolher outro sistema operacional, e aguarde que o download começará.



Faça a instalação e execute o programa recém baixado.  
Clique no ícone destacado:

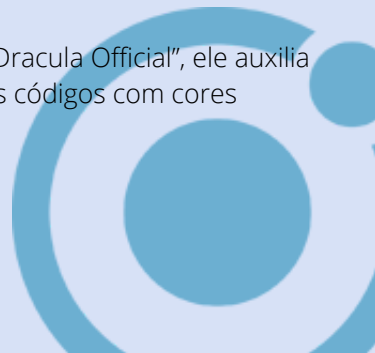


Busque por “Angular Extension Pack” e instale a que for da Loiane Groner



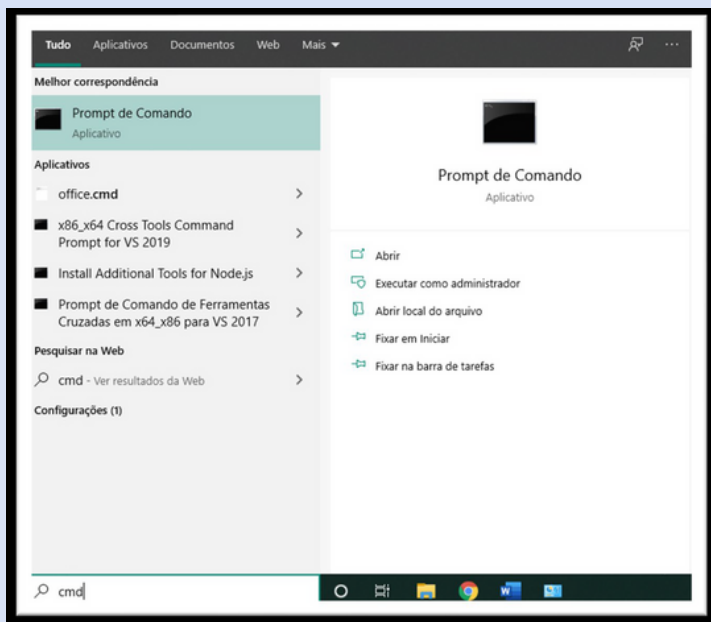
## OPCIONAL

- Faça o mesmo processo de busca e instale o “Dracula Official”, ele auxilia no quesito visual da programação, deixando os códigos com cores diferentes e assim facilitando a compreensão.

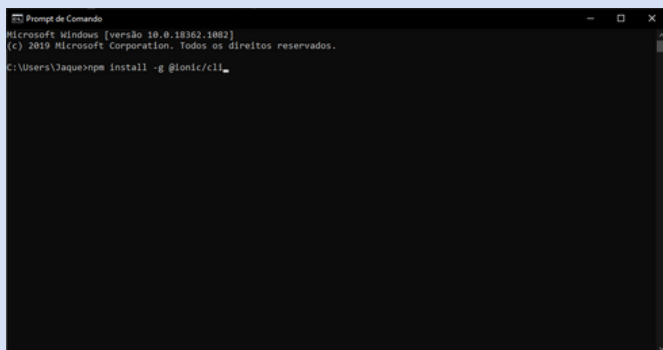


## DESENVOLVENDO COM O IONIC

Após isso, feche o programa, busque o prompt de comando em sua barra de pesquisa e execute



Digite o seguinte comando **npm install -g @ionic/cli**, dê Enter e aguarde a instalação



Para começar a criar uma aplicação digite **ionic start nomedapasta -appName "Nome do aplicativo"**

```
ionic
Microsoft Windows [versão 10.0.18362.1082]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Jaque>ionic start MeuPrograminha -appname "O nome do aplicativo"
```

Selecione com qual das duas opções você quer usar integrado “Angular ou React” e logo depois selecione o template que será usado.

```
Selecionar ionic
Microsoft Windows [versão 10.0.18362.1082]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Jaque>ionic start MeuPrograminha -appname "O nome do aplicativo"

Pick a framework!

Please select the JavaScript framework to use for your new app. To bypass this prompt next time, supply a value for the --type option.

Framework: Angular

Let's pick the perfect starter template!

Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your app. To bypass this prompt next time, supply template, the second argument to ionic start.

Starter template: (Use arrow keys)
tabs | A starting project with a simple tabbed interface
sidemenu | A starting project with a side menu with navigation in the content area
blank | A blank starter project
list | A starting project with a list
my-first-app | An example application that builds a camera with gallery
conference | A kitchen-sink application that shows off all Ionic has to offer
```

Escolha se você deseja fazer integração com o Capacitor, sendo y ou Y para Yes e N ou n para No. Dê Enter e aguarde enquanto algumas dependências são instaladas



```
Selecione um rpm
Microsoft Windows [versão 10.0.18362.1082]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Jaque>ionic start MeuPrograminha --appname "O nome do aplicativo"

Pick a framework!

Please select the JavaScript framework to use for your new app. To bypass this prompt next time, supply a value for the
--type option.

Framework: Angular

Let's pick the perfect starter template!

Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your app. To bypass this
prompt next time, supply template, the second argument to ionic start.

Starter template: sidemenu
You are already in an Ionic project directory. Do you really want to start another project here? Yes
Preparing directory .\MeuPrograminha in 33.71ms
Downloading and extracting sidemenu starter in 3.84s
Integrate your new app with Capacitor to target native iOS and Android? (y/N)
```

- Após isso, talvez lhe apareça “Create free ionic account?” (Criar uma conta gratuita Ionic), digite y ou Y para Yes e N ou n para No, caso Yes, abrirá uma nova guia em seu navegador para a criação. Aperte Enter e aguarde o projeto terminar de ser criado.

Agora qualquer comando que você queira rodar do ionic, precisará estar dentro da pasta do projeto.

Digite **cd nomedapasta** e dê Enter.

```
Selecione um Prompt de Comando
npm WARN notsup Not compatible with your version of node/npm: watchpack-chokidar2@2.0.0
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node_modules\watchpack-chokidar2\node_modules\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {"os":"darwin","arch":"arm"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node_modules\webpack-dev-server\node_modules\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {"os":"darwin","arch":"arm"} (current: {"os":"win32","arch":"x64"})

added 1499 packages from 1224 contributors and audited 1504 packages in 478.746s

65 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

your Ionic app is ready! Follow these next steps:

  - Go to your new project: cd .\MeuPrograminha
  - Run ionic capacitor within the app directory to see your app in the browser
  - Run ionic capacitor add to add a native iOS or Android project using Capacitor
  - Generate your app icon and splash screens using cordova-res --skip-config --copy
  - Explore the Ionic docs for components, tutorials, and more: https://ion.link/docs
  - Building an enterprise app? Ionic has Enterprise Support and Features: https://ion.link/enterprise-edition

C:\Users\Jaque>cd MeuPrograminha
C:\Users\Jaque\MeuPrograminha>
```

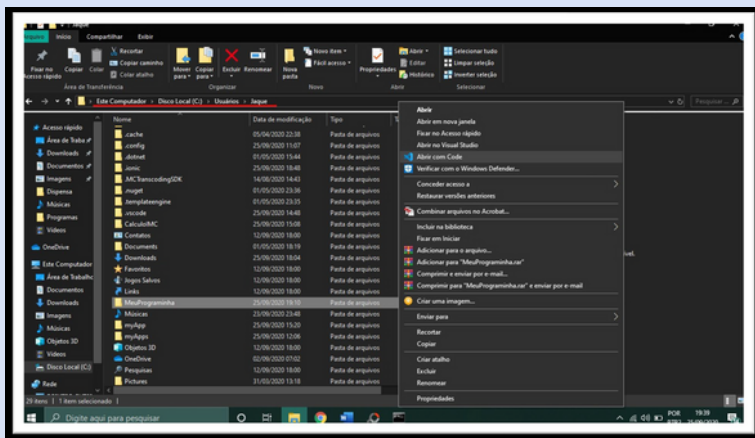
Agora qualquer comando que você queira rodar do ionic, precisará estar dentro da pasta do projeto.

Digite **cd nomedapasta** e dê Enter.

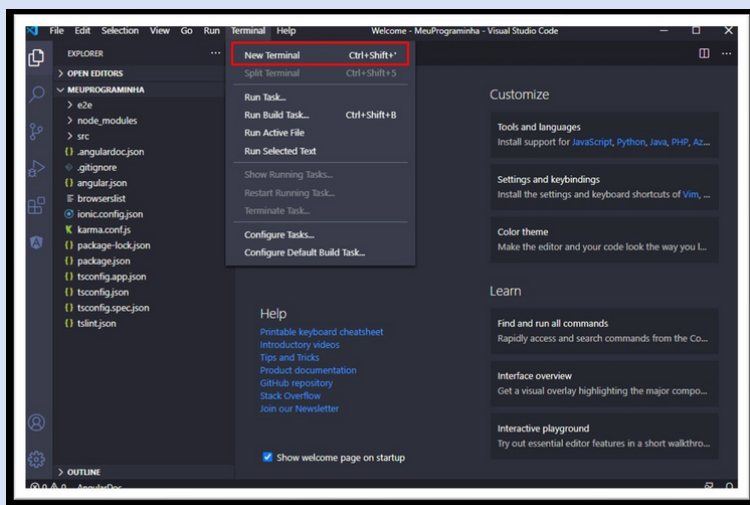


## TESTANDO LOCALMENTE

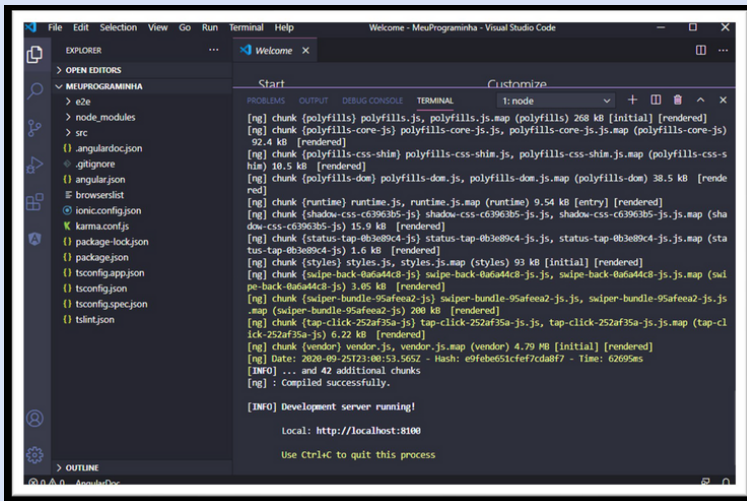
Busque no Explorador de Arquivos por sua pasta criada, logo depois, clique com o botão direito em cima dela e selecione "Abrir com Code".



Após o programa ser aberto, clique em Terminal e New Terminal.

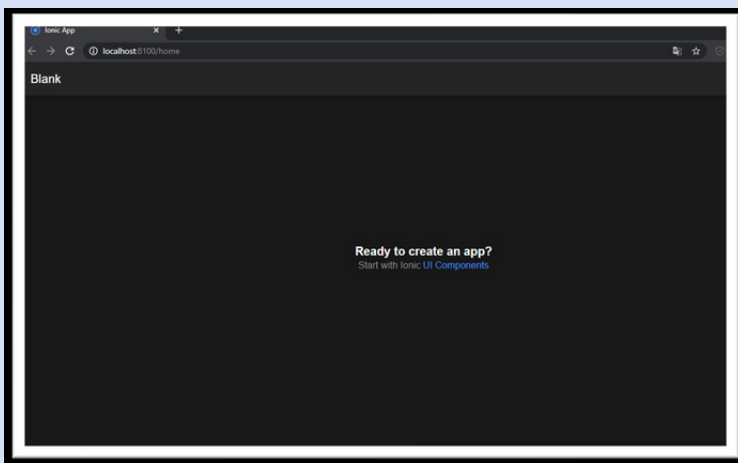


No campo que irá aparecer, digite **ionic serve**, dê Enter e aguarde. Uma nova guia será aberta em seu navegador.



- Caso apresente erro, execute o programa “Windows PowerShell” como administrador e digite o seguinte comando **Set-ExecutionPolicy by-pass** confirmando caso peça. Depois faça o passo 26 novamente.

Essa será a forma como está sua aplicação visualmente, se desejar ver como o modelo ficará tanto em Android quanto em IOS, digite **ionic lab** (pedirá autorização para instalação)



Acessando o lado esquerdo do seu Visual Studio Code, na aba Explorer, você pode fazer modificações em sua aplicação. Por exemplo:

```
1 <ion-header [translucent]="true">
2   <ion-toolbar>
3     <ion-title>
4       Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content [fullscreen]="true">
10  <ion-header collapse="condense">
11    <ion-toolbar>
12      <ion-title size="large">Blank</ion-title>
13    </ion-toolbar>
14  </ion-header>
15
16  <div id="container">
17    <strong>Ready to create an app?</strong>
18    <p>Start with Ionic 4s target="blank" rel="noopener noreferrer" href="https://ionicframework.com/docs/comp">
19  </div>
20 </ion-content>
```

```
[ng] Date: 2020-09-26T22:00:09-0322 - Hash: e438ed556478c806 - Time: 1473ms
[INFO] ... and 42 additional chunks
[ng] : Compiled successfully.

[INFO] Development server running!
      local: http://localhost:8100
      Use Ctrl/C to quit this process

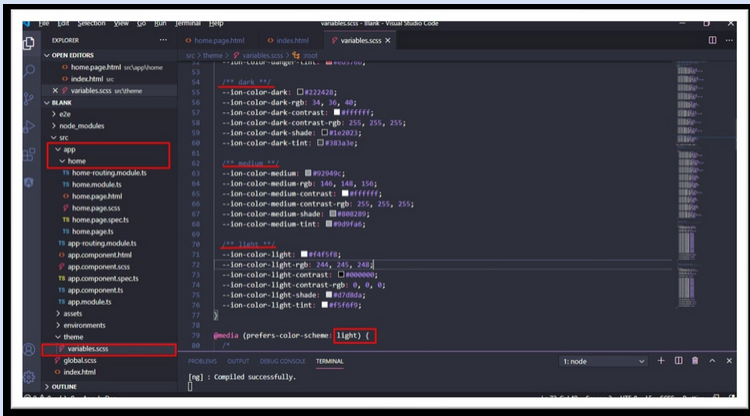
[INFO] Browser window opened to http://localhost:8100!
```

```
1 <ion-header [translucent]="true">
2   </ion-header>
3   <ion-content [fullscreen]="true">
4
5     <div id="container">
6       <h1>Hello World</h1>
7       <p>Ohana means family, crFamily means nobody gets left behind or forgotten.</p>
8     </div>
9   </ion-content>
10 </ion-content>
```

```
[ng] Time: 808ms
[ng] : Compiled successfully.
[ng] Date: 2020-09-26T22:52:33-0322 - Hash: 5f4d286e4c5b7c6287aa
[ng] 49 additional chunks
```

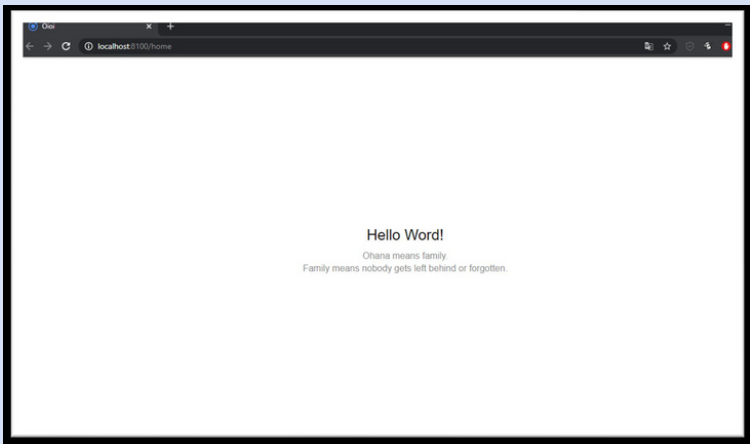
```
1 <ion-header [translucent]="true">
2   </ion-header>
3   <ion-content [fullscreen]="true">
4
5     <div id="container">
6       <h1>Hello World</h1>
7       <p>Ohana means family, crFamily means nobody gets left behind or forgotten.</p>
8     </div>
9   </ion-content>
10 </ion-content>
```

```
[ng] Time: 808ms
[ng] : Compiled successfully.
[ng] Date: 2020-09-26T22:52:33-0322 - Hash: 5f4d286e4c5b7c6287aa
[ng] 49 additional chunks
```



- O campo **@media (prefers-color-scheme: dark) {** foi modificado para **@media (prefers-color-scheme: light) {** estando agora no modo branco/claro.
- Obs: conforme as mudanças forem sendo feitas em cada página, salve com **Ctrl+S** para que assim sejam aplicadas no seu programa e visualizadas.

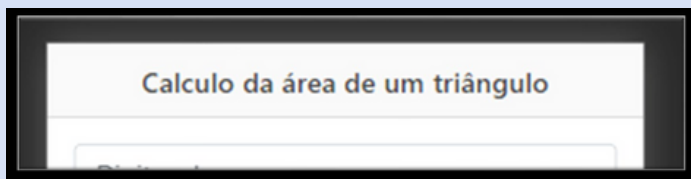
O resultado das modificações:



## "TRIANGULO"

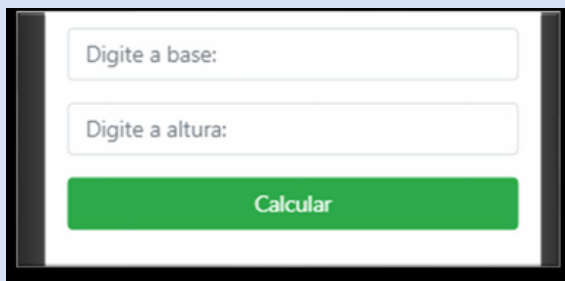
### EXPLICANDO COMPONENTES:

```
<ion-pane>
  <ion-header-bar class="bar-stable">
    <h1 class="title" style="text-align: center ;">Calculo da área de um triângulo</h1>
  </ion-header-bar>
  <ion-content>
```



- A tag **<ion-header-bar>** faz referência ao cabeçalho da sua aplicação, contendo dentro de si a tag **<h1...>**, a qual define o que está dentro desse cabeçalho, no meu caso, o intuito da minha aplicação.

```
<div class="w-100 h-100 p-3 d-flex align-items-center">
  <div class="w-100">
    <input type="number" class="form-control w-100 mb-3" id="input_base" name="base" placeholder="Digite a base" />
    <input type="number" class="form-control w-100 mb-3" id="input_altura" name="altura" placeholder="Digite a altura" />
    <button class="btn btn-success btn-block" id="btn_calcular" (click)="insertdata()">Calcular</button>
  </div>
</div>
```

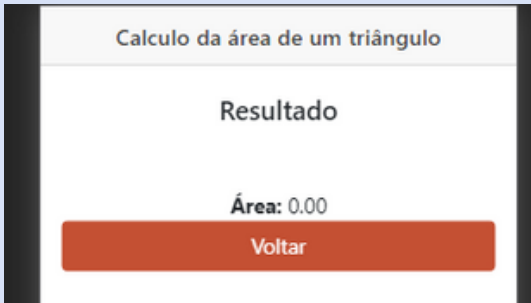


- A tag **<div...>** representa nosso contêiner inicial, o qual contém um input do tipo number, ele define nosso campo de entrada como numérico.
  - Você poderá ver acessando: [https://www.w3schools.com/html/html\\_form\\_input\\_types.asp](https://www.w3schools.com/html/html_form_input_types.asp)
- Nesse input há um id que é definido para identificar e diferenciar os campos de entrada.
- Logo em seguida há mensagem pré escrita, o que facilita para usuário saber o que deve ser digitado naquele campo (placeholder).
- Também há um botão que efetua um cálculo assim que clicado.

```

<div class="w-100 h-100 d-flex align-items-center justify-content-center p-3" id="cont_resultado" style="position: fixed; top: 0; right: -110%;
<div class="w-100">
  <div class="text-center mb-5">Resultado</div>
  <p class="mb-0 text-center"><strong>Área: <span class="area"></span></p>
  <button class="btn btn-info btn-block btn_close">Voltar</button>
</div>

```

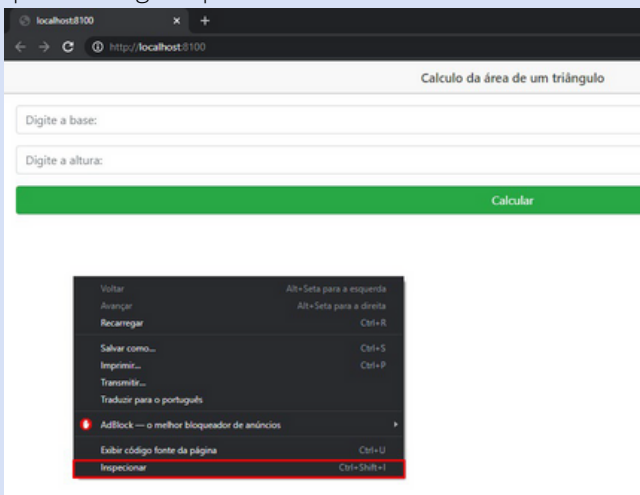


- Essa tag **<div...>** representa o container do resultado, contendo um outro div dentro com o texto "Resultado".
- Logo abaixo há uma tag parágrafo (**<p...>**) com a tag **<strong>** dentro, a qual é usada para textos de importância ou deixando em negrito.
  - Dentro da tag Strong há um **<span class="area">**, ele retornará o resultado do cálculo.
- Também é possível ver um botão "Voltar", o qual é definido para regressar a página anterior ao ser clicado.

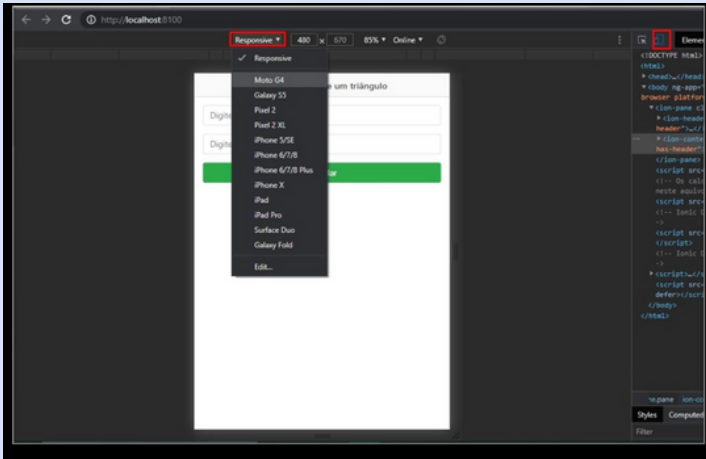
## CRIAÇÃO DE CLASSES

Execute seu projeto com "Ionic serve" em seu terminal para que se possa ter uma melhor visualização.

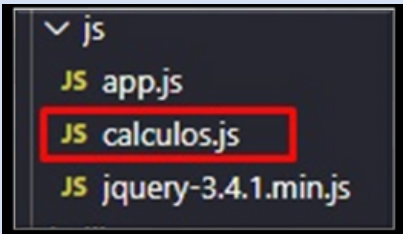
Para que essa visualização se assemelhe ao mobile, no seu navegador, inspecione a guia que foi aberta.



Em seguida, ao lado de “Elements”, clique em no ícone com dois retângulos. Se trata do “Toggle device toolbar”, ele mudará a visualização da sua aplicação para ficar mais parecida com a do seu dispositivo móvel, podendo ser configurada a sua escolha em “Responsive”.



Crie um arquivo com a extensão “.js”, é o arquivo onde o cálculo deve ser feito.



As informações abaixo serão efetuadas quando o botão foi acionado, ou melhor dizendo, clicado.

```
$(function() {  
  $('#btn_calcular').on('click', function(argument) {
```

Declare as variáveis da classe que referenciará o campo de entrada e o resultado.

```
var base = $('#input_base').val();  
var altura = $('#input_altura').val();  
var cont_resultado = $('#cont_resultado');
```

Instancie o objeto.

```
var c = new Calculo();
```

```
// set base  
c.base = base;  
  
// set altura  
c.altura = altura;  
  
// get area  
var area = c.area;
```

- Defina as variáveis com os métodos get ou set, e declare a variável área que será onde ocorrerá toda a operação de cálculo.
  - Lembrando que set atribuiu um valor, enquanto get pega.

Primeira linha: irá escrever o valor de área na tela. Segunda linha: irá mostrar a tela de resultados. Terceira linha: limpa o campo de entrada “base”. Quarta linha: limpa o campo de entrada “altura”.

```
cont_resultado.find('.area').text(area.toFixed(2));  
cont_resultado.css({"right": "0"});  
$('#input_base').val('');  
$('#input_altura').val('');
```

O código abaixo fecha a tela de resultados, regressando para a inicial.

```
cont_resultado.find('.btn_close').on('click', function() {  
    cont_resultado.css({"right": "-110%"});  
});
```

O valor que foi digitado em base será atribuído a “this\_base”. O valor que foi digitado em altura será atribuído a “this\_altura”. A variável área receberá o retorno do cálculo feito.



```

class Calculo {

  set base(base) {

    this._base = base;
  }

  set altura(altura) {

    this._altura = altura;
  }

  get area() {

    return (this._base * this._altura)/2;
  }
}

```

Não se esqueça de declarar a localização no final do seu documento "index.html".

```

</ion-pane>

<script src="js/jquery-3.4.1.min.js"></script>
<script src="js/calculos.js"></script>
</body>
</html>

```

**OBS:** Todas definições de cores, formatos e etc, são definidas no arquivo style de seu projeto. Você pode encontra-lo acessando a pasta "css".  
Caso tenha dúvidas, acesse:

<https://ionicframework.com/docs/theming/basics>,

<https://ionicframework.com/docs/theming/platform-styles>,

<https://ionicframework.com/docs/theming/css-variables>,

<https://ionicframework.com/docs/theming/css-shadow-parts>,

<https://ionicframework.com/docs/theming/colors>.



## GERANDO APK DO SEU PROJETO

No seu projeto, vá até o arquivo “config.xml”, por ele você é capaz de configurar sua aplicação.

Mude o nome do seu aplicativo em **<name>**, a descrição em **<description>**, em value de **<preference name="SplashScreenDelay" ...>** o tempo de duração da tela inicial.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <widget id="io.ionic.starter" version="0.0.1" xmlns="http://www.w3.org/ns/widgets" xmlns:cdv="http://cordova.apache.org/ns/
3 <name>Triangulo</name>
4 <description>Ionic/Cordova app.</description>
5 <author email="hi@ionicframework.com" href="http://ionicframework.com">Ionic Framework Team</author>
6 <content src="index.html" />
7 <access origin="*" />
8 <allow-intent href="http:///*/*" />
9 <allow-intent href="https:///*/*" />
10 <allow-intent href="tel:*" />
11 <allow-intent href="sms:*" />
12 <allow-intent href="mailto:*" />
13 <allow-intent href="geo:*" />
14 <preference name="ScrollEnabled" value="false" />
15 <preference name="BackupWebStorage" value="none" />
16 <preference name="SplashMaintainAspectRatio" value="true" />
17 <preference name="FadeSplashScreenDuration" value="300" />
18 <preference name="SplashShowOnlyFirstTime" value="false" />
19 <preference name="SplashScreen" value="screen" />
20 <preference name="SplashScreenDelay" value="3000" />
21 <platform name="android">
```

Para modificar a imagem de SplashScreen ou Icon, é preciso se atentar nas medidas necessárias para cada um deles, e a extensão, sempre sendo “.png”. Para mudar o icon, na pasta “resources”, exclua a imagem “icon.png” e coloque a desejada, tendo o mesmo nome, dentro da pasta. No terminal, digite **“ionic cordova resources --icon”**, será feito upload do arquivo e automaticamente gerado um para cada tamanho necessário. O mesmo deve ser feito com o splash screen, a imagem poderá ter nome “splash.png”, “splash.psd”. No terminal deve ser digitado **“ionic cordova resources –splash”**.

Modificações feitas, é hora de gerar o apk.

No seu terminal, digite **“ionic cordova build android”**, esse comando irá gerar seu arquivo apk no seguinte caminho:

“...\PastaDoSeuProjeto\platforms\android\app\build\outputs\apk\debug”



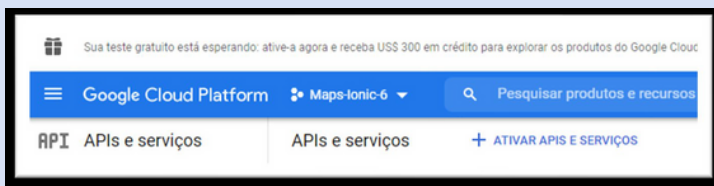
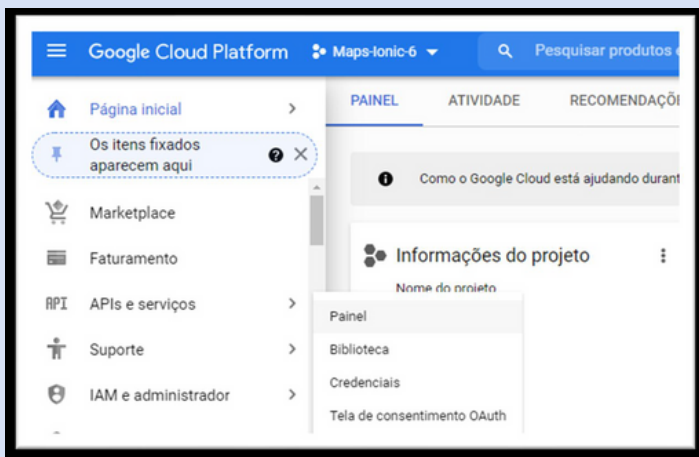
# "GPS"

## CONFIGURANDO PROJETO:

Inicialmente faça login no Google Cloud (<https://cloud.google.com/>) e crie um **novo projeto**

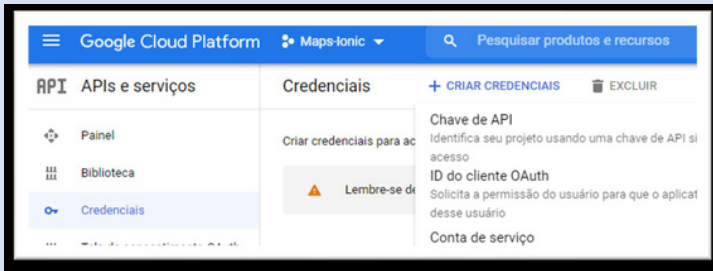


Após selecionar o projeto recém criado, acesse **"painel"** e **"ativar APIS e serviços"**.



Busque por "Maps JavaScript API" e ative.

Em seguida, crie uma chave API indo em "credenciais", "criar credenciais", "Chave de API". Copie e guarde a chave gerada. Obs: todo local que tiver key=API\_KEY você irá substituir por ela.



Logo depois adicione os types do Google Maps. Acesse a documentação do Google Maps, através do link ["https://developers.google.com/maps/documentation/javascript/overview?hl=pt\\_BR"](https://developers.google.com/maps/documentation/javascript/overview?hl=pt_BR) e busque, na coluna lateral, **"Using TypeScript"**. Execute, dentro da pasta do seu projeto, o seguinte comando em seu terminal:

```
npm i -D @types/googlemaps
```

Dentro do arquivo **"tsconfig.app.json"**, em **"types:" []**, dentro do colchete, adicione o termo **googlemaps**. Também modifique o **"tsconfig.spec.json"**, adicionando, **"googlemaps"**:

```
tsconfig.app.json X
tsconfig.app.json > [ ] files
1 {
2   "extends": "./tsconfig.json",
3   "compilerOptions": {
4     "outDir": "./out-tsc/app",
5     "types": ["googlemaps"]
6   },

```

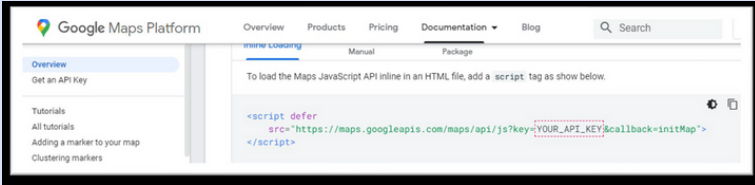
```
tsconfig.spec.json X
tsconfig.spec.json > ...
1 {
2   "extends": "./tsconfig.json",
3   "compilerOptions": {
4     "outDir": "./out-tsc/spec",
5     "types": [
6       "jasmine",
7       "node",
8       "googlemaps"
9   ]

```



## ADICIONANDO O MAPA

Volte para o Overview da documentação e adicione o JS do Maps no "**index.html**", logo acima da tag head e coloque a chave API, removendo a função de call-back (**&callback=initMap**).



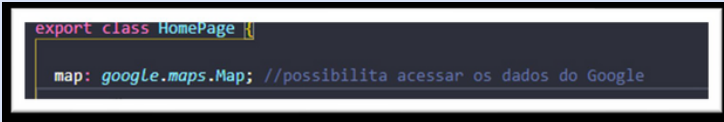
Vá no seu arquivo "**home.page.html**", apague tudo que estiver dentro da tag **<ion-content>** e adicione o seguinte comando:

```
<div #map id="map"></div>
```

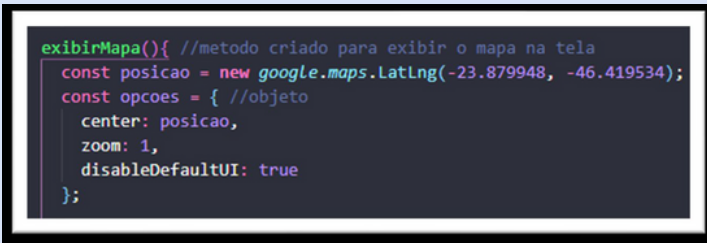
No arquivo "**home.page.scss**", adicione a altura e a largura como 100%.

•Acesse seu arquivo "**home.page.ts**", local onde toda a programação ocorre.

Crie a primeira variável do tipo mapa:



Crie o método **exibirMapa()**



- Escolha uma coordenada aleatória para preencher o **LatLng**;
- Dentro do objeto opções é definido a posição como centralizada, o zoom como uma visão de países, continentes, uma visão "mundo" (que varia de 1 para mundo a 20 para construções) e desabilita funções do google maps, como ferramenta de zoom, satélite e etc.

Defina o elemento:

```
@ViewChild('map',{read: ElementRef, static: false}) mapRef:ElementRef; //ViewChild interface que permite pegar elementos
```

Chama o elemento mapRef na função, passando um elemento nativo e as opções declaradas:

```
this.map = new google.maps.Map(this.mapRef.nativeElement, opcoes);
```

A função do ciclo de vida do Ionic deve ser chamada para que o mapa seja exibido. Antes mesmo de exibir a tela pro usuário, o mapa já começa a ser renderizado.

```
ionViewWillEnter(){  
  this.exibirMapa();  
}
```

## BUSCANDO POSIÇÃO E ADICIONANDO MARCADOR

Vá até a documentação do Ionic e busque por Geolocation (<https://ionicframework.com/docs/native/geolocation/>). Instale as seguintes linhas no terminal, dentro da pasta de sua aplicação:

```
ionic cordova plugin add cordova-plugin-geolocation npm install @ionic-native/geolocation
```

Copie e cole o comando **import { Geolocation } from '@ionic-native/geolocation/ngx'**; para seu arquivo **"home.module.ts"**, e o declare como **providers: [Geolocation]**.

```
import { Geolocation } from '@ionic-native/geolocation/ngx';  
  
@NgModule({  
  imports: [  
    CommonModule,  
    FormsModule,  
    IonicModule,  
    HomePageRoutingModule  
  ],  
  declarations: [HomePage],  
  providers: [Geolocation]  
})  
export class HomePageModule {}
```

Cole o mesmo comando em "home.page.ts" e crie uma variável chamada minhaPosicao:

```
minhaPosicao: google.maps.LatLng;
```

Crie o método buscarPosicao() e cole o segundo trecho de código, que a documentação do Ionic disponibiliza:

```
this.geolocation.getCurrentPosition().then((resp) => {  
  // resp.coords.latitude  
  // resp.coords.longitude  
}).catch((error) => {  
  console.log('Error getting location', error);  
});
```

Mude seu construtor para **constructor(private geolocation: Geolocation) {}**

Declare sua posição dentro da variável para obter sua localização dentro do mapa.

```
this.minhaPosicao=new google.maps.LatLng(resp.coords.latitude,resp.coords.longitude);
```

Crie um método **irParaMinhaPosicao()** para focar em sua posição dentro do mapa

```
irParaMinhaPosicao(){  
  this.map.setCenter(this.minhaPosicao);  
  this.map.setZoom(15);  
}
```

- Posiciona o mapa centralizando em sua posição;
- Dá um zoom numa visão de "ruas" para exibir sua localização.

Chame sua função para buscarPosicao logo abaixo do código que chama o elemento mapRef:

```
this.map = new google.maps.Map(this.mapRef.nativeElement, opcoes);  
this.buscarPosicao();
```

Chame a função `irParaMinhaPosicao`, dentro do método `buscarPosicao`:

```
buscarPosicao(){
  this.geolocation.getCurrentPosition().then((resp) => { //pega a posição do usuário uma s
    // resp.coords.latitude
    // resp.coords.longitude

    this.minhaPosicao=new google.maps.LatLng(resp.coords.latitude,resp.coords.longitude);

    this.irParaMinhaPosicao();

  }).catch((error) => {
    console.log('Error getting location', error);
  });
}
```

Um marcador é colocado para sinalizar a posição do usuário

```
const marker = new google.maps.Marker({ //a
  position: this.minhaPosicao,
  title: 'Você está aqui!',
  animation: google.maps.Animation.BOUNCE,
  map: this.map
});
```

- Marcador recebe um objeto de position “minhaPosicao”;
- Ao posicionar o ponteiro do mouse em cima do marcador, num navegador web, exibe o título do marcador que diz “você está aqui!”;
- Marcador com uma animação do tipo BOUNCE que faz ele pular;
- Passa o mapa.

Em “home.page.html”, dentro de uma div, foi criada uma tag `<h5>` que exibe por escrito a localização do usuário

```
<ion-header [translucent]="true">
  <ion-toolbar>
    <ion-title>
      GPS - IONIC
    </ion-title>
  </ion-toolbar>
  <div>
    <h5>Sua localização: {{minhaPosicao}}</h5>
  </div>
</ion-header>
```



## INSERINDO BOTÃO

Usando o código "**ionic g page nomedapagina**", crie uma pagina que será usada como inicial.

Para que isso ocorra, em **app-routing.module.ts** mude o `redirectTo` para **`redirectTo: 'nomedapagina'`**,

Já em **nomedapagina.page.html** modifique seu código para que ele tenha a seguinte aparência:

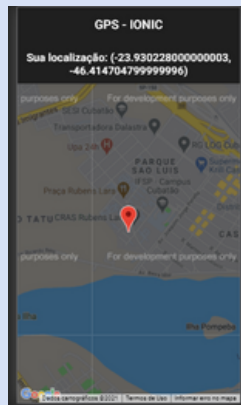
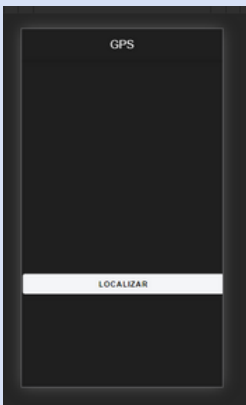
```
src > app > primeira > primeira.page.html
1  <ion-header>
2  <ion-toolbar>
3  <ion-title>GPS</ion-title>
4  </ion-toolbar>
5  </ion-header>
6  <div></div>
7  <ion-content>
8  <ion-button (click)="btnClicked()" color="dark">
9  LOCALIZAR
10 </ion-button>
11 </ion-content>
12
```

Em **nomedapagina.page.ts** crie a ação que será executada quando o button for clicado e logo após modifique o arquivo **nomedapagina.page.scss** conforme preferir

```
src > app > primeira > primeira.page.ts ...
1  import { Component, OnInit } from '@angular/core';
2  import { Router } from '@angular/router';
3
4  @Component({
5  selector: 'app-primeira',
6  templateUrl: './primeira.page.html',
7  styleUrls: ['./primeira.page.scss'],
8  })
9  export class PrimeiraPage implements OnInit {
10
11  constructor(private router: Router) {}
12
13  btnClicked(){
14  console.log('Btn Clicked');
15  this.router.navigate(['home']);
16  }
17  }
18
19  ngOnInit() {
20  }
21  }
22
23  }
```

```
src > app > primeira > primeira.page.scss > ...
1  ion-title{
2  text-align: center;
3  }
4  ion-button{
5  display:inline-block;
6  width:100%;
7  font-weight:bold;
8  }
9  }
10 }
11 div{
12 height: 10cm;
13 }
```

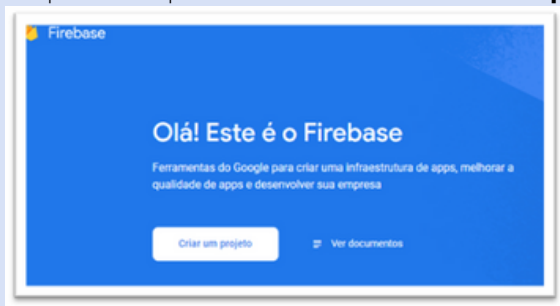
Esse será o visual da sua aplicação:



# "CRUD"

## CONFIGURANDO PROJETO:

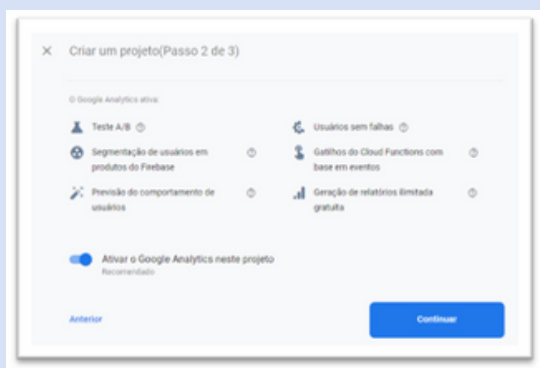
Inicialmente faça login no site do Firebase (<https://firebase.google.com/>).  
Clique em "Ir para o console" e crie um **novo projeto**;



Escolha o nome do projeto e aceite os termos do Firebase;



Mantenha a opção "Ativar o Google Analytics neste projeto" ativada e prossiga;



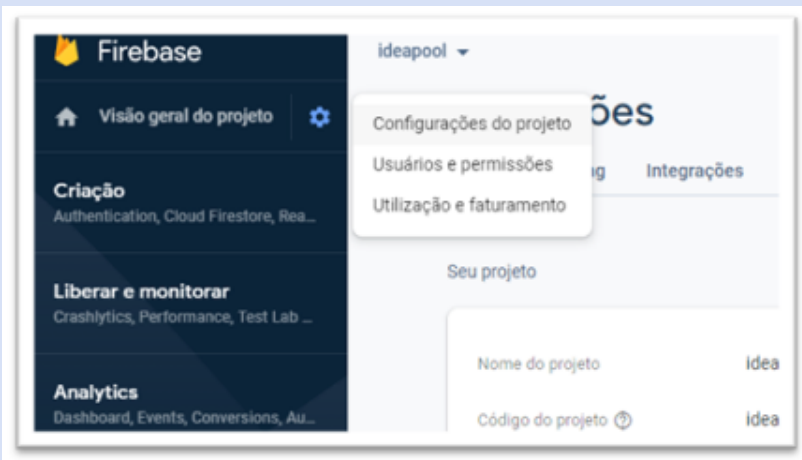
Escolha a localização, marque as caixas de seleção e crie o projeto;



Selecione a opção que apresenta “</>”, registre seu app e prossiga;



Na engrenagem que se encontra ao lado de “Visão geral do projeto”, selecione “Configurações do projeto”;



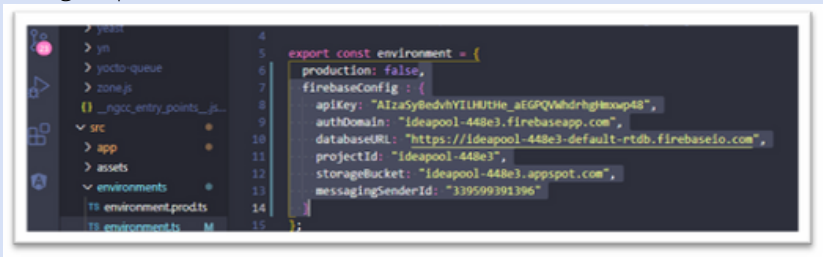
- Desça a página até encontrar “Firebase SDK snippet”, marque o radiobox de CDN e copie o seguinte conjunto de códigos:



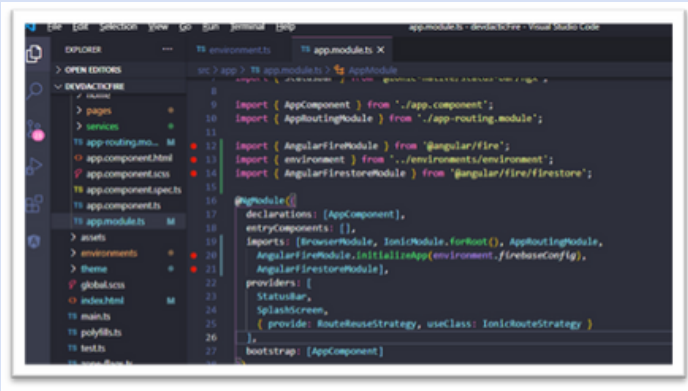
## CRIANDO PROJETO

- Em seu console, digite os seguintes códigos:
  - o `ionic start NomeDoProjeto blank --type=angular`
  - o `cd NomeDoProjeto`
  - o `npm install firebase @angular/fire`
    - Aqui estamos instalando o pacote Firebase e AngularFire do NPM.
  - o `ionic g page pages/idealista`
  - o `ionic g page pages/ideaDetails`
  - o `ionic g service services/idea`
  - o `code`

Após o projeto ser aberto no Visual Code, vá até seu arquivo **environment.ts** e cole o conjunto de códigos recém copiados, substituindo o sinal de igual por “:”;



Agora vá até `app.module.ts` e declare os imports necessários;



```
import { AppComponent } from './app.component';
import { AppRoutingModule } from './app-routing.module';

import { AngularFireModule } from '@angular/fire';
import { environment } from '../environments/environment';
import { AngularFireFirestoreModule } from '@angular/fire/firestore';

@NgModule({
  declarations: [AppComponent],
  entryComponents: [],
  imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireFirestoreModule],
  providers: [
    StatusBar,
    SplashScreen,
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
  ],
  bootstrap: [AppComponent]
})
```

Altere as informações de roteamento dentro do `app-routing.module.ts`;



```
import { NgModule } from '@angular/core';
import { PreloadAllModules, RouterModule, Routes } from '@angular/router';

const routes: Routes = [
  { path: '', loadChildren: './pages/idea-list/idea-list.module#IdeaListPageModule' },
  { path: 'idea', loadChildren: './pages/idea-details/idea-details.module#IdeaDetailsPageModule' },
  { path: 'idea/:id', loadChildren: './pages/idea-details/idea-details.module#IdeaDetailsPageModule' },
];

@NgModule({
  imports: [
    RouterModule.forRoot(routes, { preloadingStrategy: PreloadAllModules })
  ],
  exports: [RouterModule]
})

export class AppRoutingModule {}
```

Ajuste seu arquivo `idea.service.ts` para que tenha a seguinte cara:

```
import { Injectable } from '@angular/core';
import { AngularFireStore, AngularFireStoreCollection, AngularFireStoreDocument, DocumentReference } from '@angular/fire/firestore';
import { map, take } from 'rxjs/operators';
import { Observable } from 'rxjs';

export interface Idea {
  id: string,
  name: string,
  notes: string
}

@Injectable({
  providedIn: 'root'
})
export class IdeaService {
  private ideas: Observable<Idea>;
  private ideaCollection: AngularFireStoreCollection<Idea>;

  constructor(private afs: AngularFireStore) {
    this.ideaCollection = this.afs.collection<Idea>('ideas');
    this.ideas = this.ideaCollection.snapshotChanges().pipe(
      map(actions => {
        return actions.map(a => {
          const data = a.payload.doc.data();
          const id = a.payload.doc.id;
          return { id, ...data };
        });
      });
    );
  }
}
```

```
getIdeas(): Observable<Idea[]> {
  return this.ideas;
}

getIdea(id: string): Observable<Idea> {
  return this.ideaCollection.doc<Idea>(id).valueChanges().pipe(
    take(1),
    map(idea => {
      idea.id = id;
      return idea;
    });
  );
}

addIdea(idea: Idea): Promise<DocumentReference> {
  return this.ideaCollection.add(idea);
}

updateIdea(idea: Idea): Promise<void> {
  return this.ideaCollection.doc(idea.id).update({ name: idea.name,
  notes: idea.notes });
}

deleteIdea(id: string): Promise<void> {
  return this.ideaCollection.doc(id).delete();
}
```

Ajuste seu arquivo **idea-list.page.ts** para que tenha a seguinte cara:

```
import { Component, OnInit } from '@angular/core';
import { IdeaService, Idea } from 'src/app/services/idea.service';
import { Observable } from 'rxjs';

@Component({
  selector: 'app-idea-list',
  templateUrl: './idea-list.page.html',
  styleUrls: ['./idea-list.page.scss'],
})
export class IdealistPage implements OnInit {

  private ideas: Observable<Idea[]>;

  constructor(private ideaService: IdeaService) { }

  ngOnInit() {
    this.ideas = this.ideaService.getIdeas();
  }
}
```

Ajuste seu arquivo **idea-list.page.html** para que tenha a seguinte cara:

```
<ion-header>
  <ion-toolbar color="warning">
    <ion-title>NOTAS</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>
  <ion-fab vertical="bottom" horizontal="end" slot="fixed">
    <ion-fab-button routerLink="/idea" color="warning">
      <ion-icon name="add"></ion-icon>
    </ion-fab-button>
  </ion-fab>

  <ion-list>
    <ion-
item button [routerLink]="['/idea', idea.id]" *ngFor="let idea of (idea
as | async)">
      {{ idea.name }}
    </ion-item>
  </ion-list>
</ion-content>
```

Ajuste seu arquivo **idea-list.page.html** para que tenha a seguinte cara:

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { IdeaService, Idea } from 'src/app/services/idea.service';
import { ToastController } from '@ionic/angular';

@Component({
  selector: 'app-idea-details',
  templateUrl: './idea-details.page.html',
  styleUrls: ['./idea-details.page.scss'],
})
export class IdeaDetailsPage implements OnInit {

  idea: Idea = {
    name: '',
    notes: ''
  };
}
```

```
constructor(private activatedRoute: ActivatedRoute, private ideaServ
ice: IdeaService,
             private toastCtrl: ToastController, private router: Rout
er) {}

ngOnInit() { }

ionViewDidEnter() {
  let id = this.activatedRoute.snapshot.paramMap.get('id');
  if (id) {
    this.ideaService.getIdea(id).subscribe(idea => {
      this.idea = idea;
    });
  }
}
```

```

addIdea() {
  this.ideaService.addIdea(this.idea).then(() => {
    this.router.navigateByUrl('/');
    this.showToast('Nota adicionada');
  }, err => {
    this.showToast('Ocorreu um problema ao adicionar a nota :(');
  });
}

```

```

}

deleteIdea() {
  this.ideaService.deleteIdea(this.idea.id).then(() => {
    this.router.navigateByUrl('/');
    this.showToast('Nota apagada');
  }, err => {
    this.showToast('Ocorreu um problema ao apagar a nota :(');
  });
}

updateIdea() {
  this.ideaService.updateIdea(this.idea).then(() => {
    this.showToast('Nota atualizada');
  }, err => {
    this.showToast('Ocorreu um problema ao atualizar a nota :(');
  });
}

showToast(msg) {
  this.toastCtrl.create({
    message: msg,
    duration: 2000
  }).then(toast => toast.present());
}
}

```

Ajuste seu arquivo **idea-details.page.html** para que tenha a seguinte cara:

```

<ion-header>
  <ion-toolbar color="warning">
    <ion-buttons slot="start">
      <ion-back-button defaultHref="/"></ion-back-button>
    </ion-buttons>
    <ion-title>Detalhes da nota</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>

  <ion-item>
    <ion-label position="stacked">Título</ion-label>
    <ion-input [(ngModel)]="idea.name"></ion-input>
  </ion-item>

  <ion-item>
    <ion-label position="stacked">Descrição</ion-label>
    <ion-textarea [(ngModel)]="idea.notes" rows="8"></ion-textarea>
  </ion-item>

```

```

</ion-content>

<ion-footer *ngIf="!idea.id">
  <ion-toolbar color="warning">
    <ion-
  button expand="full" fill="clear" color="light" (click)="addIdea()">
    <ion-icon name="checkmark" slot="start"></ion-icon>
    Adicionar nota
  </ion-button>
  </ion-toolbar>
</ion-footer>

<ion-footer *ngIf="idea.id">
  <ion-row no-padding text-center>
    <ion-col size="6">
      <ion-
  button expand="block" fill="outline" color="danger" (click)="deleteIdea()">
    <ion-icon name="trash" slot="start"></ion-icon>
    Apagar
  </ion-button>
  </ion-col>
  <ion-col size="6">
    <ion-
  button expand="block" fill="solid" color="warning" (click)="updateIdea()">
    <ion-icon name="save" slot="start"></ion-icon>
    Atualizar
  </ion-button>
  </ion-col>
  </ion-row>
</ion-footer>

```

## CONFIGURANDO BANCO DE DADO

Selecione "Realtime Database" e crie seu banco de dados;



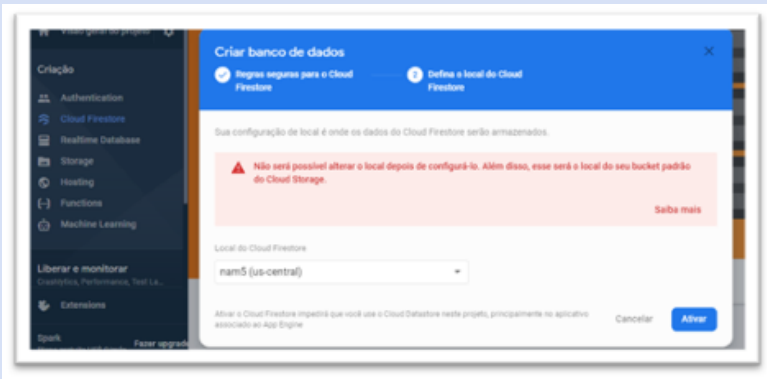




Selecione "Ativar";

Agora selecione "Cloud Firestore", "Criar banco de dados" e por fim "Ativar";





E pronto, essa será a visão do seu programa sem e com nota adicionada:

- A cada vez que uma nota for adicionada, os dados serão gravados no banco de dados criado.

