Configuração do ambiente no link **docs.rocketseat.dev**

Para visualizar a versão do java, digite **java –version**

Para instalar a cli do react-native, digite **yarn global add react-native-cli**

Para rodar a aplicação do native, digite:

**react-native run-android** ou **react-native run-ios**

Para visualizar as opções do cli do react-native, digite **react-native –h**

Para criar um projeto chamado mobile com o react-native, digite:

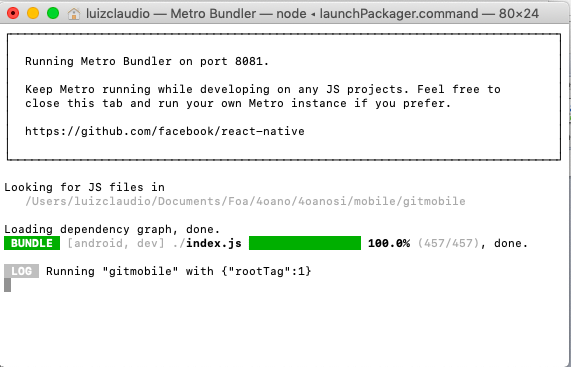
**react-native init mobile**

Entre na pasta do projeto digitando **cd mobile**

Dentro da pasta do projeto, com a emulador do android aberto ou o smartphone android conectado na usb ou wi-fi, digite:

**react-native run-android** e a aplicação será exibida.

**Metro Bundler**



Ao iniciar a aplicação será aberto o Metro Bundler, ele tem como objetivo fazer o bundle da aplicação, ou seja, empacotar todos os arquivos JS e unifica em um único e enviar para o smartphone. Para a aplicação funcionar no smartphone ou emulador, ele deve permanecer aberto e sendo executado. Se por algum motivo a janela do Metro Bundle não inicializar dentro da pasta do projeto digite **react-native start**

No smartphone para abrir o menu do dispositivo, dê uma chacoalha nele e será exibido uma janela com várias opções.

**Visualizando o código do projeto**

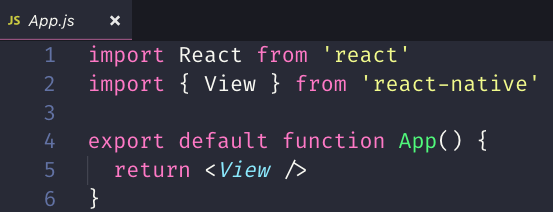
Abra o VS Code e adicione a pasta do seu projeto

**Edite o arquivo App.js**

Neste arquivo você encontra tags específicas do native, como por exemplo, <View> como sendo a div do html, a <Text> representando qualquer texto.

Para configurar o css não tem id e nem classe, é utilizada uma tag style que recebe um objeto com estilização.

Apague todo o conteúdo do arquivo App.js e defina o código abaixo.



Verifique se uma tela em branco será exibida no smartphone ou emulador.

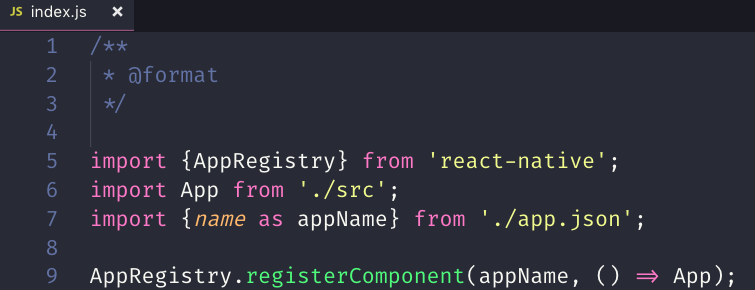
**Entendendo a estrutura de arquivos**

Observe que a estrutura é bem parecida com o React-Web, tem o arquivo package.json, node\_modules, \_\_tests\_\_, a pasta android onde estão os códigos nativos java da aplicação e a pasta ios,

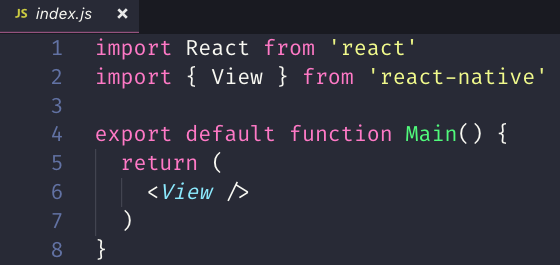
O arquivo index.js é o arquivo inicial da aplicação e o registro é do react-native e não do react-web. Observe que ele está importando o arquivo App.js

**Criando a aplicação**

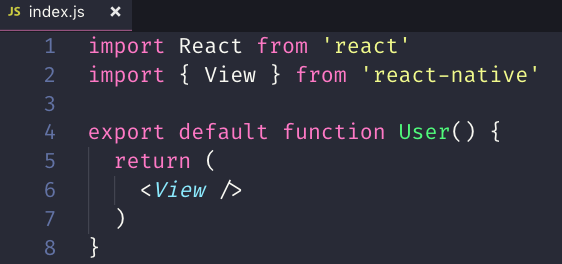
* Dentro da pasta do projeto, crie a src
* Crie o arquivo src/index.js
* Crie a pasta src/pages
* Crie a pasta src/pages/Main
* Crie a pasta src/pages/User
* Copie o conteúdo do arquivo App.js para dentro do src/index.js
* Apague o arquivo App.js que está na pasta raiz.
* Altere a linha 6 do arquivo index.js que está na raiz, conforme print abaixo



Crie o arquivo src/pages/Main/Index.js



**Crie o arquivo src/pages/User/index.js**



**Instalando os pacotes de navegação (react-navigation, react-navigation-stack) e gestos (react-native-gesture-handler)**

**yarn add react-navigation react-navigation-stack react-native-gesture-handler**

A partir deste momento será feita todas as modificações necessárias dentro dos códigos android e ios nativos para ela funcionar.

Toda vez que fizer esta alteração será necessário instalar a aplicação novamente no smartphone digitando:

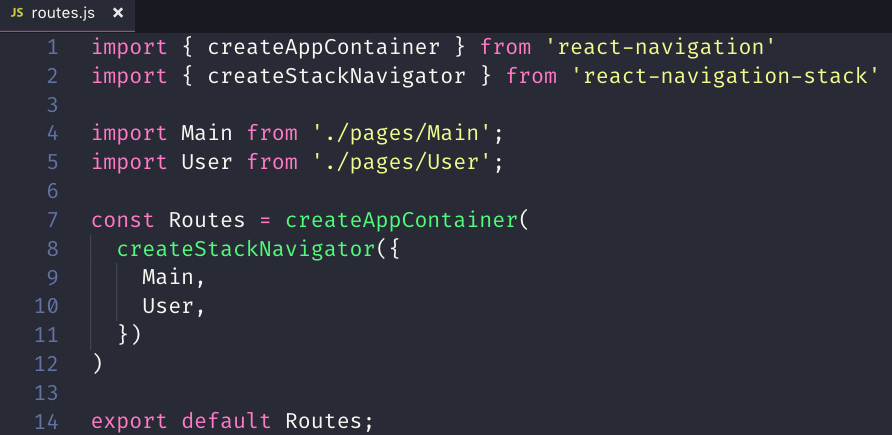
**react-native run-android ou react-native run-ios**

**Criando as rotas da aplicação**

**Definindo a rota com o createStackNavigator**

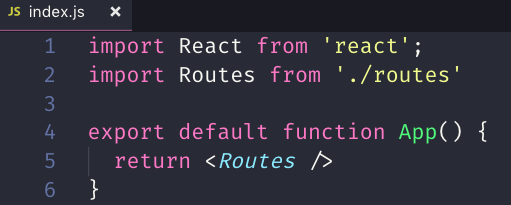
Crie o arquivo src/routes.js, conforme print abaixo:

* Na linha 1 importe o createAppContainer do react-navigation
* Na linha 2 importa o createStackNavigator do react-navigation-stack
* Nas linhas 4 e 5 importam as páginas Main e User respectivamente.
* Na linha 7 cria o objeto Routes com o createAppContainer (O createAppContainer é como se fosse o browser-router da web, ele contém as configurações para o roteamento funcionar independente do tipo de rota que estiver utilizando, ele sempre deve estar entre todas as rotas)
* Na linha 8, o createStackNavigator recebe um objeto com as rotas da aplicação. (O createStackNavigator contém um tipo de configuração de rota, ele mostra um header na parte superior, faz a navegação por stack (pilha) cada vez que o usuário acessa um rota será criada uma pilha e ele conseguirá acessá-la normalmente, ele as rotas anteriores em background e não remove o funcionamento delas).



**Importando o arquivo de rotas**

**Edite o arquivo src/index.js, conforme print abaixo**



**Obs: Ao instalar novas dependências o react pode apresentar algum erro na aplicação, para resolver cancele a execução do Metro Bundle pressionando Ctrl + C e digite o comando react-native start –reset-cache. Este comando reinicia o cache e o bundle da aplicação.**

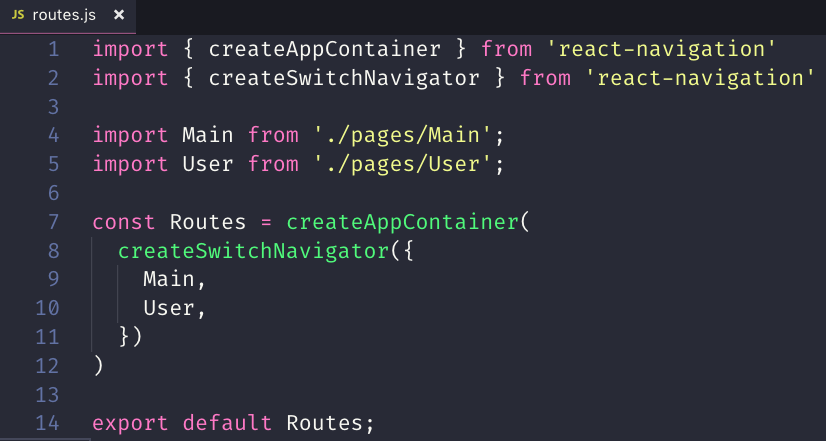
Abra a aplicação no smartphone e verifique a exibição na parte superior do header

**Definindo a rota com o createSwitchNavigator**

O createSwitchNavigator ele não tem um feedback visual e também não cria a pilha de rotas, se o usuário navegar da pagina Main para User, a página Main é totalmente deletada

Edite o arquivo src/routes.js, conforme print abaixo:

Altere a linha 2 e 8.



Abra a aplicação no smartphone e verifique que não será exibida o header.

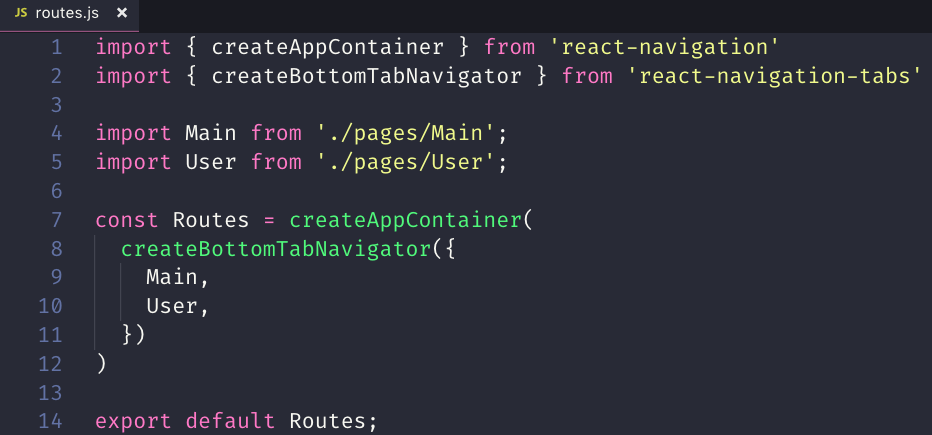
**Definindo a rota com o createBottomTabNavigator**

O createBottomTabNavigator cria a navegação por abas na parte inferior da página, para utilizá-la tem que instalar os seguintes pacotes:

* **yarn add react-navigation-tabs**
* **yarn add react-native-reanimated**

Edite o arquivo src/routes.js, conforme print abaixo:

Altere a linha 2 e 8.



Abra a aplicação no smartphone e verifique as abas do header na parte inferior.

**Definindo a rota com o createMaterialTopTabNavigator**

O createMaterialTopTabNavigator cria as abas na parte superior do smartphone para utilizá-lo tem que instalar o pacote react-navigation-tabs, digitando:

* **yarn add react-navigation-tabs**

Edite o arquivo src/routes.js, conforme print abaixo:

Altere a linha 2 e 8.



Abra a aplicação no smartphone e verifique as abas do header na parte superior.