

Praticando os react Native

Disponível em:
https://www.oxnar.com.br/aulas/react

Prof. André Aparecido da Silva

Arquivo App.js define os componentes

```
1
2   import React, { useState, useEffect, useRef } from 'react';
3   import {
4     View,
5     Text,
6     Image,
7     TouchableOpacity,
8     StyleSheet,
9     SafeAreaView,
10    StatusBar,
11    Dimensions,
12    Alert,
13  } from 'react-native';
14
15  const { width } = Dimensions.get('window');
16  const CARD_SIZE = (width - 48 - 30) / 4;
17
18  const personagens = [
19    { id: 1, imagem: require('./assets/personagem1.png') },
20    { id: 2, imagem: require('./assets/personagem2.png') },
21    { id: 3, imagem: require('./assets/personagem3.png') },
22    { id: 4, imagem: require('./assets/personagem4.png') },
23    { id: 5, imagem: require('./assets/personagem5.png') },
24    { id: 6, imagem: require('./assets/personagem6.png') },
25    { id: 7, imagem: require('./assets/personagem7.png') },
26    { id: 8, imagem: require('./assets/personagem8.png') },
27    { id: 9, imagem: require('./assets/personagem9.png') },
28    { id: 10, imagem: require('./assets/personagem10.png') },
29  ];
30
31  const embaralhar = (lista) =>
32      [...lista]
33          .sort(() => Math.random() - 0.5);
34
```

Componentes

Componentes Básicos	Componentes UI
●View ● Image	●Button ●Switch
●Text ●Text Input ●Stylesheet	●Stylesheet

Exemplo View

```
import { View } from 'react-native';

const AppView = () => {
  return <View style={{ flex: 1, backgroundColor: 'lightblue' }} />;
};

export default AppView;
```

Exemplo view

```
 JavaScript
import { View } from 'react-native';
        const AppView = () =>
        {
                return <View style={{ flex: 1, backgroundColor: 'lightblue' }} />;
        };

export default AppView;
```

Exemplo Text

```
import { View, Text } from 'react-native';

const AppText = () => {
  return (
                <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
                        <Text>Olá turminha do mal!</Text>
                </View>
          );
};

export default AppText;
```

Exemplo Image

```
import { View, Image } from 'react-native';

const AppImage = () => {
 return (
  <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
   <Image
    style={{ width: 60, height: 60 }}
    source={{
     uri: 'https://reactnative.dev/img/header_logo.svg',
    }}
   />
  </View>
 );
};

export default AppImage;
```

Obs: Troque esta imagem por qualquer outra que você goste. Modifique o tamanho caso a imagem for grande.

Exemplo TextInput

```
 import React, { useState } from 'react';
import { View, TextInput, Text } from 'react-native';

const AppTextInput = () => {
  const [nome, setNome] = useState('');

  return (
    <View style={{ padding: 20 }}>
      <TextInput
        style={{ borderWidth: 1, height: 40, paddingLeft: 10 }}
        onChangeText={setNome}
        placeholder="Digite seu Nome"
        value={nome}
      />
      <Text style={{ marginTop: 10 }}>Seu Nome: {nome}</Text>
    </View>
  );
};

export default AppTextInput;
```

Exemplo StyleSheet

```
import { StyleSheet, View, Text } from 'react-native';

const AppStyle = () =>
        {
         return
                (
                    <View style={styles.viewContainer}>
                            <Text style={styles.bigText}>Estilo!
                     </Text> </View>
                 );
        };
         const styles = StyleSheet.create({
                viewContainer: {
                        flex: 1, justifyContent: 'center',
                        alignItems: 'center',
                        backgroundColor: '#e0f7fa', },
                        bigText: { fontSize: 36, color: 'blue',
                        fontWeight: 'bold',
                        },
         }
);

export default AppStyle;
```

Passos para rodas os apps

Passo a passo para rodar no VS Code:

Abra o projeto no VS Code: Navegue até a pasta do seu projeto React Native.

Cole o código no arquivo principal: Substitua o conteúdo do seu arquivo de entrada principal (geralmente App.js ou App.tsx) por qualquer um dos exemplos acima.

Inicie o servidor de desenvolvimento: No terminal do VS Code, execute:

Se estiver usando **Expo**: npx expo start

Se estiver usando **React Native CLI**: npx react-native start

Execute no emulador ou dispositivo:

Pressione a para abrir no Android Emulator.

Pressione i para abrir no iOS Simulator.

Ou escaneie o QrCode usando o aplicativo **Expo Go** no seu celular físico.

Exemplo de código com os componentes



Criando os primeiros componentes.

Crie a pasta src/components/title no ./ do app
index.js

```
import React from "react"
import {View, Text} from "react-native"
```

Criação do Componente

```
export default function Title(){
    return(
        <View>
            <Text>Titulo App</Text>
        </View>
    );

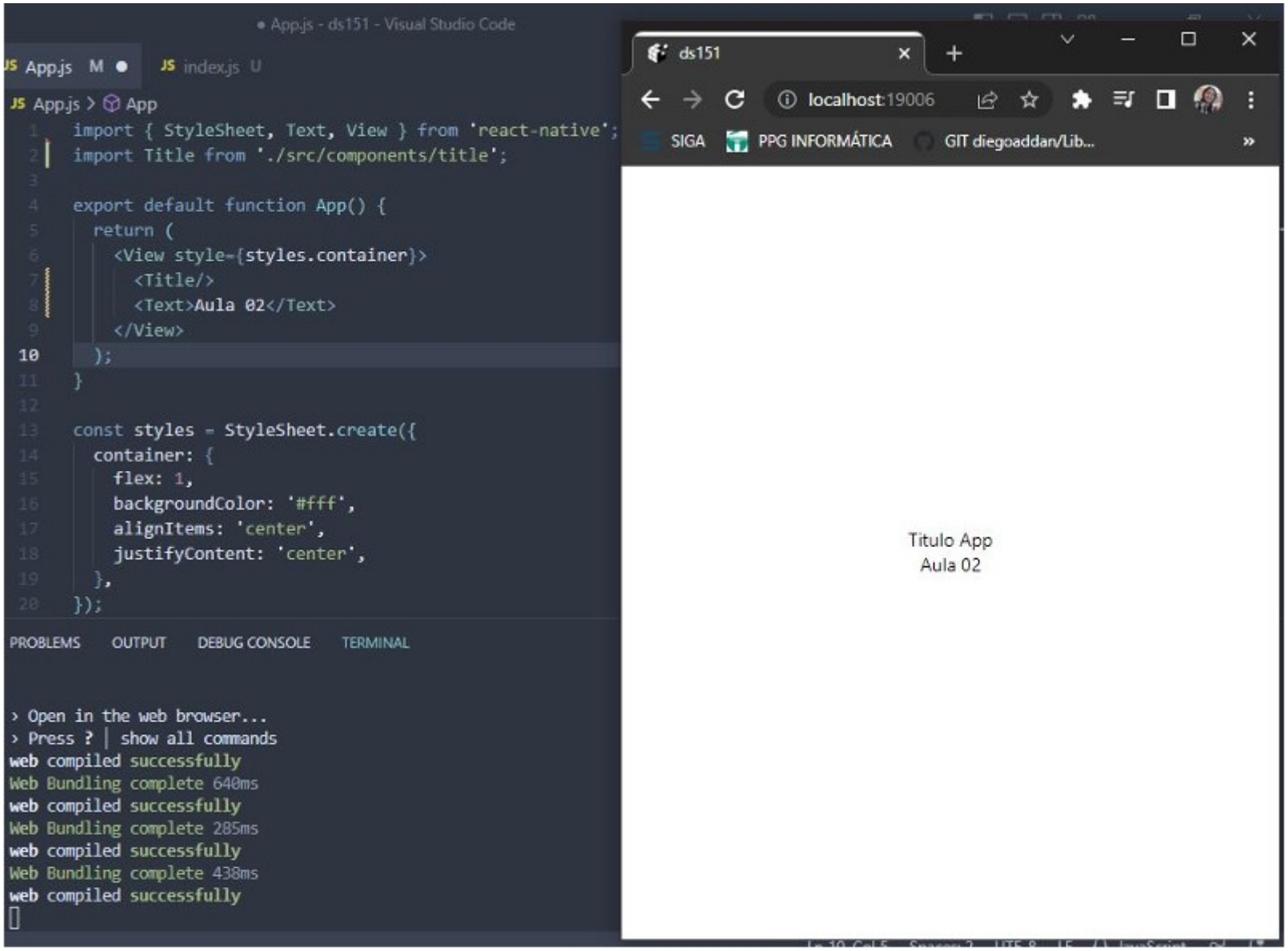
}
```

Desta forma facilita manutenção e atualizações

no App.js importamos o componente

```
import Title from './src/components/title';
```

Visão Geral


• App.js - ds151 - Visual Studio Code
App.js M
index.js U
App.js > App
import { StyleSheet, Text, View } from 'react-native';
import Title from './src/components/title';
export default function App() {
return (
<View style={styles.container}>
<Title/>
<Text>Aula 02</Text>
</View>
);
}
const styles = StyleSheet.create({
container: {
flex: 1,
backgroundColor: '#fff',
alignItems: 'center',
justifyContent: 'center',
},
});
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
› Open in the web browser...
› Press ? | show all commands
web compiled successfully
Web Bundling complete 640ms
web compiled successfully
Web Bundling complete 285ms
web compiled successfully
Web Bundling complete 438ms
web compiled successfully
ds151
localhost:19006
SIGA
PPG INFORMÁTICA
GIT diegoaddan/Lib...
Titulo App
Aula 02

Crie a pasta src/components/main no ./ do app index.js

Adicione a tag na função.

Crie a pasta src/components/form no ./ do app index.js

Componentes

Main/index.js



Podemos inserir elementos no formulário através de views

```
export default function Form ()
{
        return( //conteudo do formulario ) ;
}
```

Criaremos no form um label e um input return (//conteudo do formulario Altura Peso);

```
    return(
        //conteudo do formulario
        <View>
            <View>
                <Text>Altura</Text>
                <TextInput></TextInput>
                <Text>Peso</Text>
                <TextInput></TextInput>
            </View>
        </View>
        );
```

Podemos adicionar atributos para as tags como placeholder e tipo de entrada

```
<Text>Peso</Text>
<TextInput
placeholder="Ex. 70.10"
keyboardType="numeric"
/>
```

Componentes

Importamos o Main e o Form no App.js

```
Import Main from ‘./src/components/Main/index’
```

O form é importado indiretamente na Landing Page através da Main

```
Import Form from ‘../Form/’
```

E na função principal chamamos o componente através da função

```
export default function App() {
  return (
    <View style={styles.container}>
      <Title/>
      <Main/>
    </View>
  );
}
```

```
5  export default function Main(){
6    return(
7      <View>
8        <Form/>
9      </View>
10   );
11 }
```

Componentes

Importamos o Main no App.js

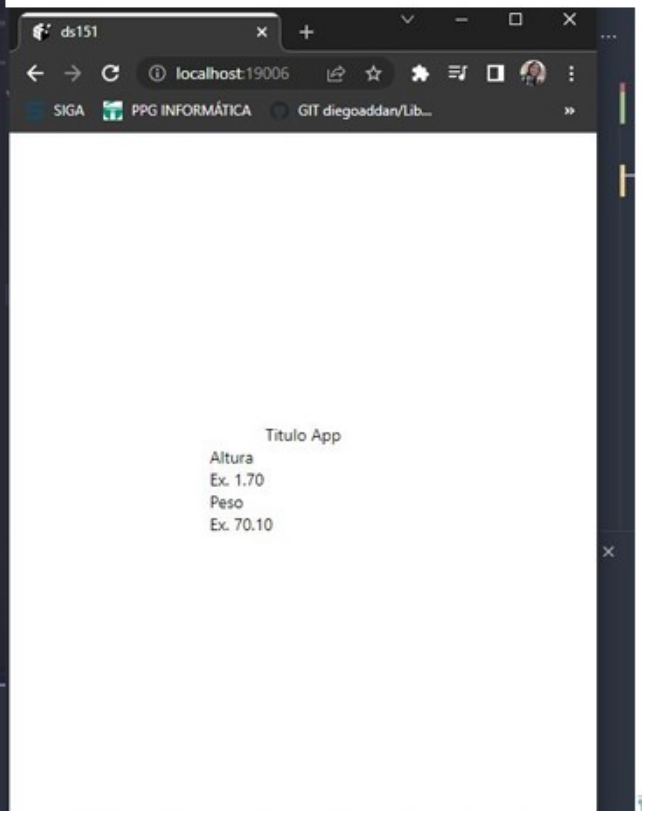
```
import React from "react"
import { View, Text, TextInput} from "react-native"

export default function Form(){
  return(
   <View>
     <View>

       <Text>Altura</Text>
       <TextInput
       placeholder="Ex. 1.75"
       keyboardType="numeric"
       />

       <Text>Peso</Text>
       <TextInput
```

```
web compiled successfully
Web Bundling complete 400ms
web compiled successfully
Web Bundling complete 265ms
web compiled successfully
```

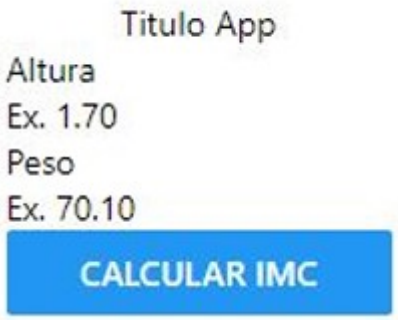


Componentess

Incluimos no Form o elemento Button

```
import {Text, TextInput, Button } from "react-native-web";
```

```
<Button title="Calcular IMC"/>
```



Precisamos incluir uma função “onPress” que adiciona comportamento ao componente.

Além disso criamos um componente para o resultado do cálculo

Crie a pasta src/components/Result/ no ./ do app index.js

```
export default function ResultImc(props){
    return(
        <View>
            <Text>{props.resultImc}</Text>
            <Text>{props.messageResultImc}</Text>
        </View>
    );
```

Propus. Para que serve???

No React Native, a palavra props é uma abreviação de properties (propriedades). Sua função é receber informações de outro componente e permitir que elas sejam utilizadas dentro do componente atual.

</> JavaScript

```
export default function ResultImc(props) {
  return (
    <View>
      <Text>{props.resultImc}</Text>
      <Text>{props.messageResultImc}</Text>
    </View>
  );
}
```

Propus. Para que serve???

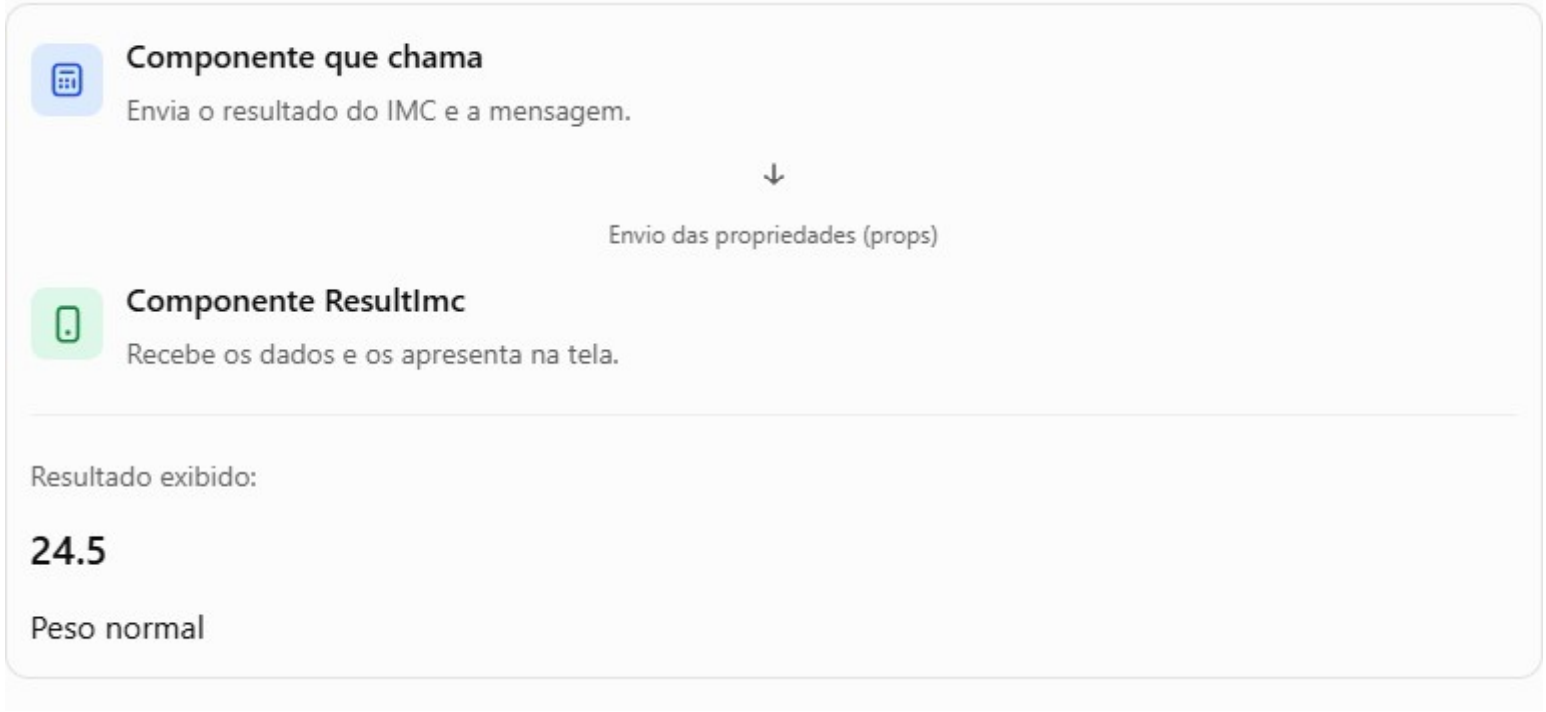
Nesse exemplo, `ResultImc` é um componente que recebe informações por meio de `props` e as apresenta na tela.

- `ResultImc` : nome do componente.
- `props` : objeto que contém as informações recebidas.
- `props.resultImc` : acessa a propriedade `resultImc` .
- `props.messageResultImc` : acessa a propriedade `messageResultImc` .
- `<Text>` : componente que exibe o conteúdo na tela.

Propus. Para que serve???

```
</> JavaScript

<ResultImc
  resultImc="24.5"
  messageResultImc= "Peso normal"
/>
```



O código continua funcionando da mesma forma, pois dados recebe o mesmo objeto de propriedades.

Outra forma mais eficiente

Uma forma mais moderna de escrever

Também é possível utilizar a desestruturação para acessar diretamente as propriedades:

JavaScript

```
export default function ResultImc({
  resultImc,
  messageResultImc
}) {
  return (
    <View>
      <Text>{resultImc}</Text>
      <Text>{messageResultImc}</Text>
    </View>
  );
}
```

No form criaremos os parâmetros que estarão sendo enviados pela função de estado (abaixo do botão)

```
<ResultImc messageResultImc={messageImc}
           resultImc={imc}></ResultImc>
```

Vamos utilizar o useState

```
import React, {useState} from "react"
```

Precisaremos de alguns parametros

Componentes

Criamos os parametros e estados iniciais

```
const [height, setHeight] = useState(null)
const [weight, setWeight] = useState(null)
const [messageImc, setMessageImc] = useState("preencha os dados")
const [imc, setImc] = useState(null)
const [textButton, setTextButton] = useState("Calcular")
```

agora criamos a função de cálculo:
altura * altura / peso

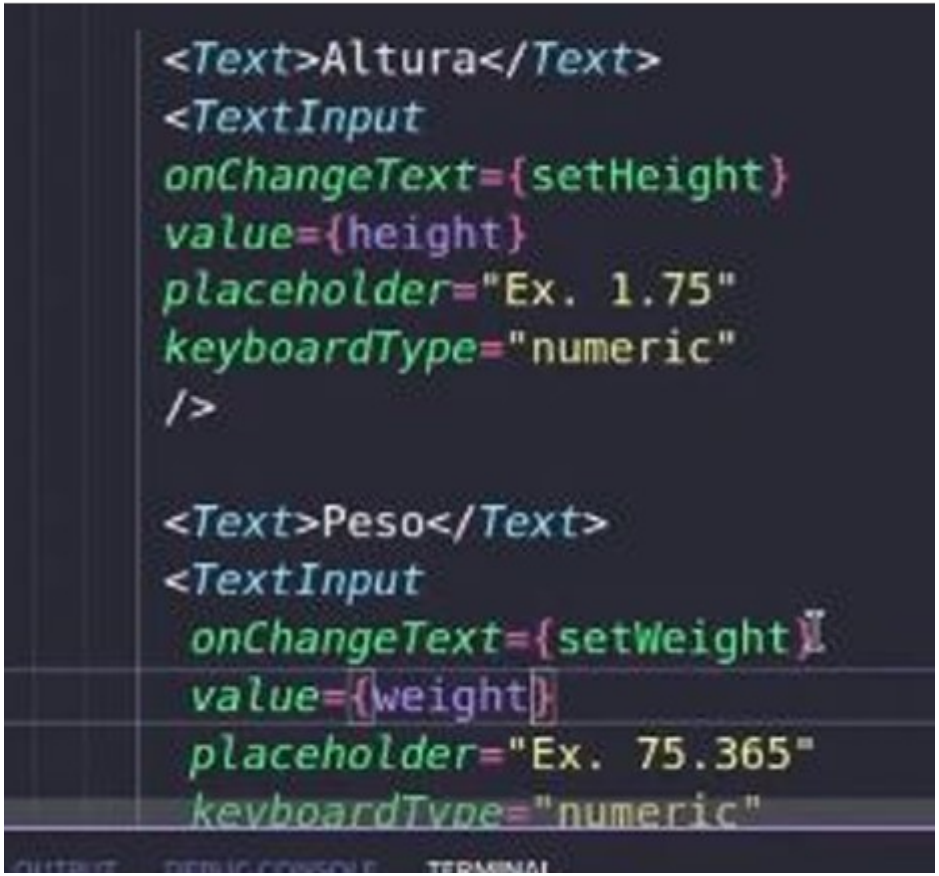
Função calcular imc ()

```
    function imcCalculator(){
        return setImc((weight/(height*height)).toFixed(2))
    }

    function validationImc(){
        if(weight != null && height != null){
            imcCalculator()
            setMessageImc("seu IMC eh igual: ")
            setTextButton("Calcular Novamente")
            return
        }
        setImc(null)
        setTextButton("Calcular")
        setMessageImc("preencha os dados")
    }
```

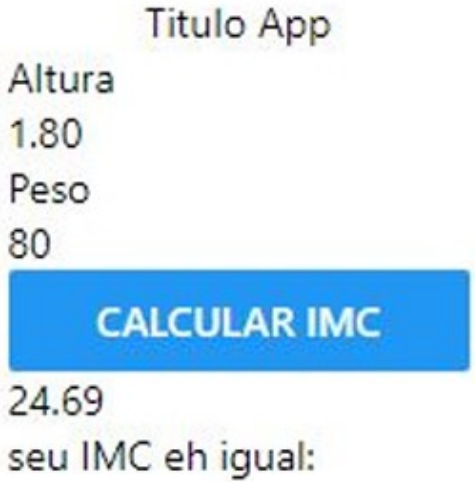
Componentes

Adiciona o comportamento no botão com onPress e estados nos Inputs



Adiciona o comportamento no botão com onPress

```
/>
<Button
 onPress={() => validationImc()}
 title="Calcular IMC"
 />
```



XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX