



UNIVERSIDADE
Estácio de Sá

Universidade	Estácio de Sá
Campus	Polo de Cobilândia / Vila – Velha/ES
Nome do Curso	Desenvolvimento Full Stack
Nome da Disciplina	RPG0023 - Vamos criar um App!
Turma	9001
Semestre	Segundo Semestre de 2024
Integrantes do Grupo	Tiago de Jesus Pereira Furtado
Matrícula	202306189045

**VILA VELHA
2024**

Micro atividade 1:

1º Configurar o ambiente de desenvolvimento React Native

1- Objetivo da Prática:

- Configurar o ambiente de desenvolvimento React Native;
- Implementar a funcionalidade de entrada de texto em um componente React Native;
- Implementar um Componente de Lista Dinâmica (ScrollView);
- Implementar componentes React Native para exibir informações de forma dinâmica em listas;
- Empregar elementos visuais em um aplicativo React Native.

Procedimentos:

Esta atividade tem como objetivo guiar os passos para a configuração do ambiente de desenvolvimento React Native para a plataforma Windows, MacOS e Linux, fornecendo-lhes as ferramentas essenciais e orientações para começar a criar aplicativos móveis com esta tecnologia. A seguir, você terá os passos necessários para a instalação do Node.JS e do framework React para os três ambientes.

Instalando o React Native no Windows

1. Para instalar o React no Windows, você precisa ter o Node.js e o npm
2. (Node Package Manager) instalados em seu sistema. Se você ainda não
3. os instalou, siga estas etapas:
 - Visite a página de download do Node.js em: <https://nodejs.org/en/download/>
 - Faça o download do instalador para o seu sistema Windows utilizando a versão.
 - Para instalar o Node.js e o npm, execute o instalador e siga cuidadosamente as instruções fornecidas.



The screenshot shows the Node.js website's download page for source code. The browser address bar displays 'nodejs.org/en/download/source-code/current'. The main heading is 'Download Node.js®' with the subtext 'Download Node.js the way you want.' Below this, there are four tabs: 'Package Manager', 'Prebuilt Installer', 'Prebuilt Binaries', and 'Source Code', with 'Source Code' being the active tab. A dropdown menu shows 'v22.11.0 (LTS)' selected, with the text 'I want the' on the left and 'version of the Node.js source code.' on the right. A large green button with a download icon and the text 'Download Node.js v22.11.0 source' is prominently displayed. Below the button, there are several links: 'Node.js includes npm (10.9.0) ↗', 'Read the changelog ↗ for this version.', 'Read the blog post ↗ for this version.', 'Learn how to verify signed SHASUMS ↗', and 'Check out how to build Node.js ↗ from source.'


nodejs.org/en/download/source-code/current

Download Node.js®

Download Node.js the way you want.

Package Manager Prebuilt Installer Prebuilt Binaries Source Code

I want the v22.11.0 (LTS) ▾ version of the Node.js source code.

 **Download Node.js v22.11.0 source**

Node.js includes [npm \(10.9.0\)](#) ↗.

Read the [changelog](#) ↗ for this version.

Read the [blog post](#) ↗ for this version.


Learn how to [verify signed SHASUMS](#) ↗

Check out how to [build Node.js](#) ↗ from source.

2. Após a conclusão da instalação, você pode verificar se o Node.js e o npm estão instalados abrindo um prompt de comando e executando os seguintes comandos:

`“node -v npm -v”`

Esses comandos devem exibir os números de versão do Node.js e do npm, respectivamente.

 Administrador: Prompt de Comando

```
Microsoft Windows [versão 10.0.19045.3633]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Administrador>npm -v
10.1.0

C:\Users\Administrador>
C:\Users\Administrador>node -v
v20.9.0

C:\Users\Administrador>
```

3. Agora, para instalar o Create React App globalmente, abra um prompt de comando e execute o seguinte comando:

“npm install -g create-react-app”

Esse comando instala o Create React App em seu sistema, tornando disponível para uso em qualquer diretório.



```
PROBLEMS OUTPUT TERMINAL PORTS powershell + - [ ] ... ^
PS C:\Users\Administrador\Desktop\Mundo 4 nivel 1> npm install -g create-react-app
>>
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a
good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not receive security updates. Please upgrade asap.

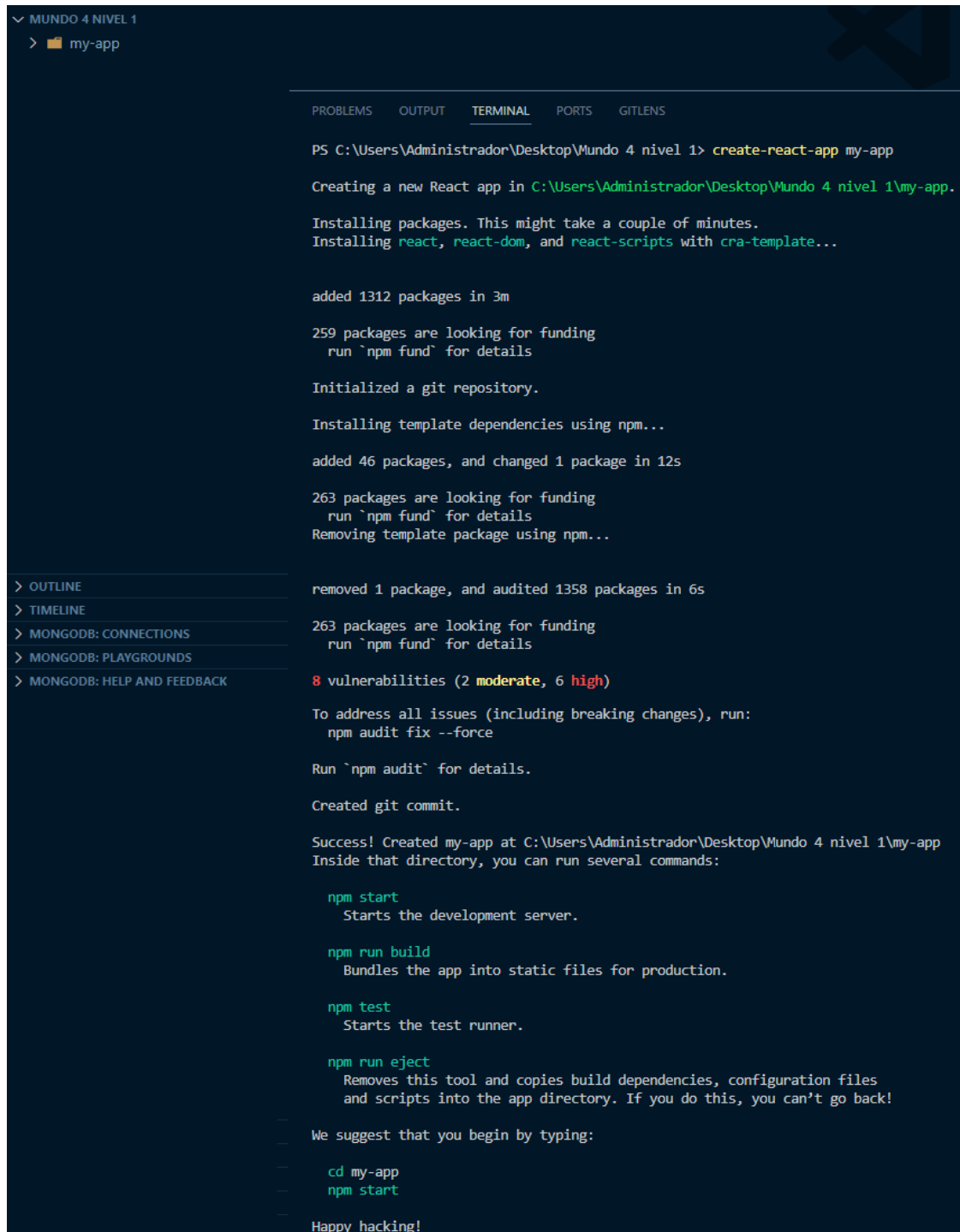
added 64 packages in 10s

4 packages are looking for funding
  run `npm fund` for details
PS C:\Users\Administrador\Desktop\Mundo 4 nivel 1> |
```

4. Agora que você tem o Create React App instalado, pode usá-lo para criar um novo projeto React. Para fazer isso, abra um prompt de comando, vá para o diretório onde você deseja que o projeto fique e execute o seguinte comando:

“create-react-app my-app”

Substitua “my-app” pelo nome desejado para seu projeto. O Create React App criará um novo diretório com o nome especificado e gerará um novo projeto React com uma estrutura e configuração de projeto recomendadas.



```

MUNDO 4 NIVEL 1
> my-app

PROBLEMS OUTPUT TERMINAL PORTS GITLENS

PS C:\Users\Administrador\Desktop\Mundo 4 nivel 1> create-react-app my-app

Creating a new React app in C:\Users\Administrador\Desktop\Mundo 4 nivel 1\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1312 packages in 3m

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

Initialized a git repository.

Installing template dependencies using npm...

added 46 packages, and changed 1 package in 12s

263 packages are looking for funding
  run `npm fund` for details
Removing template package using npm...

removed 1 package, and audited 1358 packages in 6s

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

8 vulnerabilities (2 moderate, 6 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

Created git commit.

Success! Created my-app at C:\Users\Administrador\Desktop\Mundo 4 nivel 1\my-app
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-app
  npm start

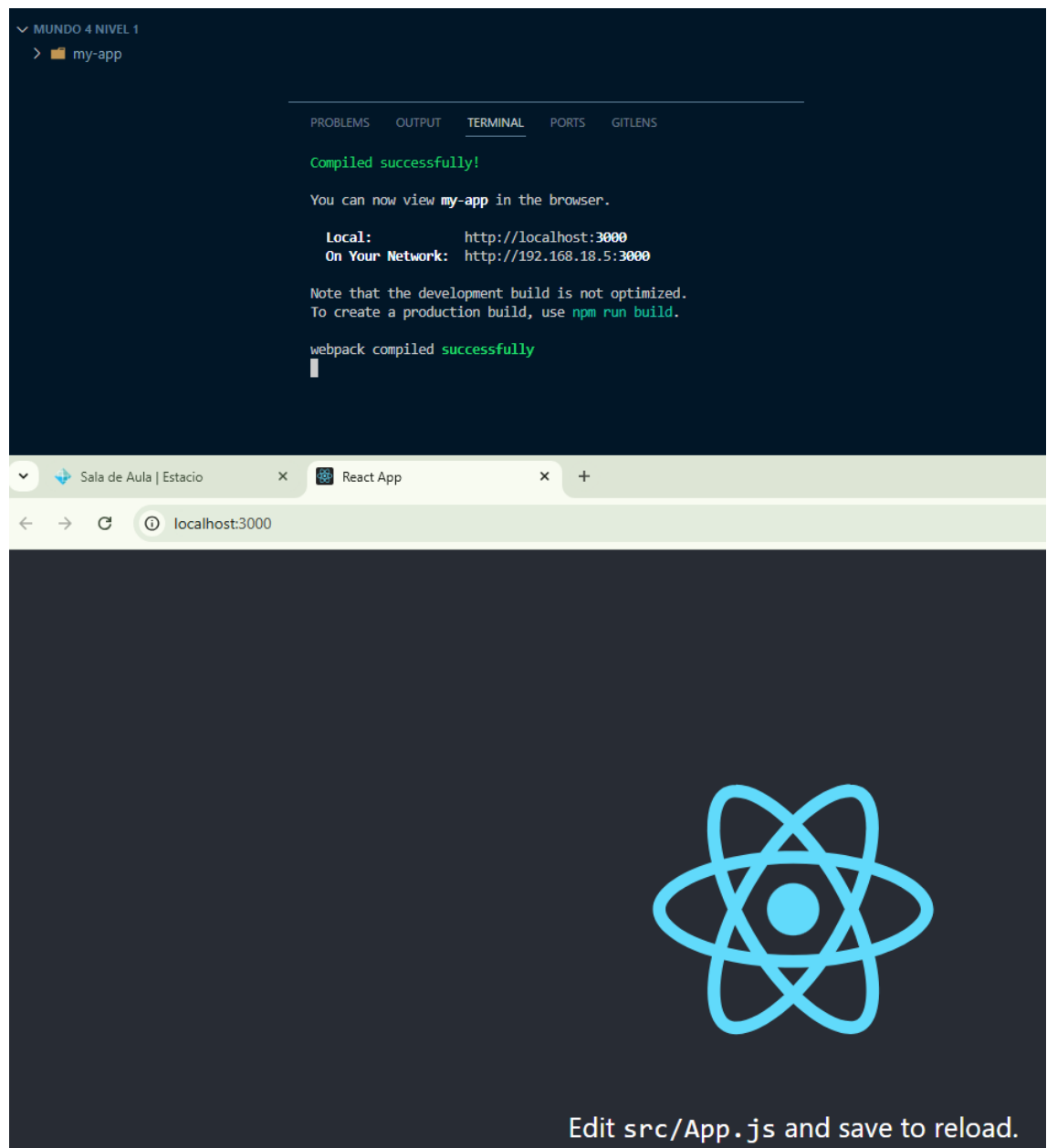
Happy hacking!
```

5. Depois que o projeto for criado, vá para o diretório do projeto executando o seguinte comando no prompt de comando:

`“cd my-app”`

6. Substitua “my-app” pelo nome do diretório do projeto. Agora, inicie o servidor de desenvolvimento executando o seguinte comando:

`“npm start”`



Micro atividade 2:

2º Implementar a funcionalidade de entrada de texto em um componente React Native

- **Objetivo da Prática:**

- Configurar o ambiente de desenvolvimento React Native;
- Implementar a funcionalidade de entrada de texto em um componente React Native;
- Implementar um Componente de Lista Dinâmica (ScrollView);
- Implementar componentes React Native para exibir informações de forma dinâmica em listas;
- Empregar elementos visuais em um aplicativo React Native.

Nesta atividade, você irá aprender a implementar a funcionalidade de entrada de texto em um componente React Native. Você criará um componente que permite aos usuários inserirem textos e ver a tradução desse texto em forma de emojis de pizza. Isso envolve o uso do componente `TextInput`, a manipulação de eventos de alteração de texto (`onChangeText`), e a exibição do resultado traduzido em tempo real. Ao final desta atividade, você terá um componente React Native funcional que demonstra uma interação de entrada de texto.

Material necessário:

Um editor de código, como o Visual Studio Code (VS Code).
Node.js e npm instalados em seu sistema.

Procedimentos

Abra o seu editor de código (por exemplo, VS Code).

Crie um novo projeto React Native ou utilize um projeto existente.

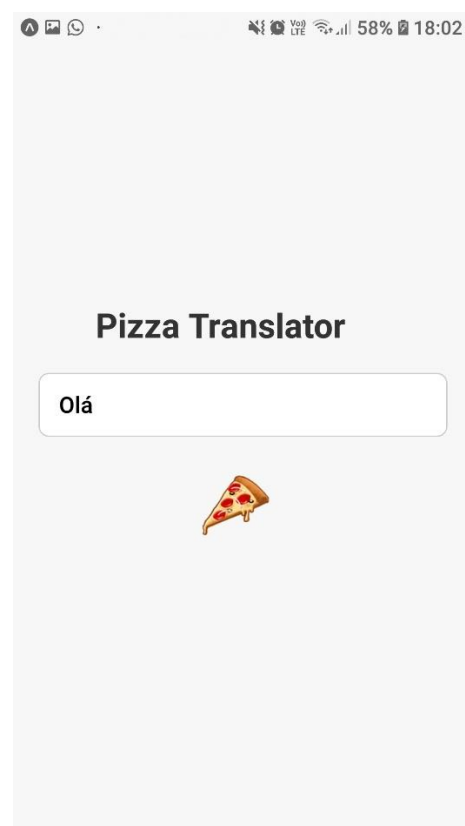
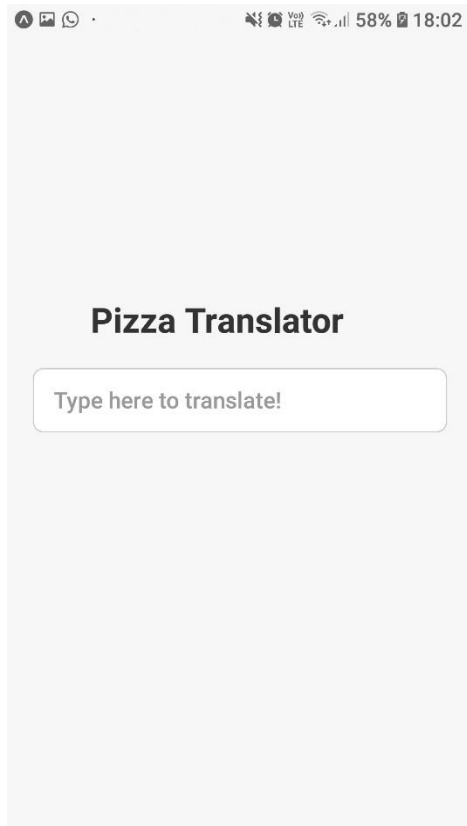
Crie um TextInput para inserir um texto;

A função a ser chamada toda vez que o texto é alterado, é o onChangeTextprop; A função a ser chamada quando o texto é enviado, é onSubmitEditingprop. Por exemplo, digamos que enquanto o usuário digita, você traduz as palavras dele para um idioma diferente. Neste novo idioma, cada palavra é escrita da mesma maneira: 🍕. Portanto, a frase "Olá, Bob" seria traduzida como "🍕 🍕 🍕". O código a seguir realiza essa ação:

```
JS App.js M X
AppPizza > JS App.js > ...
You, 22 hours ago | 1 author (You)
1 import React, { useState } from 'react'; 6.9k (gzipped: 2.7k)
2 import { Text, TextInput, View, StyleSheet } from 'react-native';
3
4 const PizzaTranslator = () => {
5   const [text, setText] = useState('');
6
7   return (
8     <View style={styles.container}>
9       <Text style={styles.header}>Pizza Translator </Text>
10      <TextInput
11        style={styles.input}
12        placeholder="Type here to translate!"
13        placeholderTextColor="#999"
14        onChangeText={newText => setText(newText)}
15        defaultValue={text}
16      />
17      <Text style={styles.result}>
18        {text
19          .split(' ')
20          .map(word => (word ? '🍕' : ''))
21          .join(' ')}
22      </Text>
23    </View>
24  );
25 };
26
27 const styles = StyleSheet.create({
28   container: {
29     flex: 1,
30     justifyContent: 'center',
31     alignItems: 'center',
32     padding: 20,
33     backgroundColor: '#f7f7f7',
34   },
35   header: {
36     fontSize: 24,
37     fontWeight: 'bold',
38     marginBottom: 20,
39     color: '#333',
40   },
41   input: {
42     height: 50,
43     width: '100%',
44     borderColor: '#ccc',
45     borderWidth: 1,
46     borderRadius: 8,
47     paddingHorizontal: 15,
48     fontSize: 16,
49     backgroundColor: '#fff',
50     shadowColor: '#000',
51     shadowOffset: { width: 0, height: 2 },
52     shadowOpacity: 0.1,
53     shadowRadius: 4,
54     elevation: 2,
55   },
56   result: {
57     marginTop: 20,
58     fontSize: 42,
59     color: '#ff6347',
60     textAlign: 'center',
61   },
62 });
63
64 export default PizzaTranslator;
65
```

Resultado da Micro Atividade 2:

Segue abaixo o resultado da execução do código que foi
Mencionado na parte acima. Onde cada palavra que e digitado
Ele transforma em um pedaço de 🍕



Micro atividade 3:

3º Implementar um Componente de Lista Dinâmica (ScrollView)

- **Objetivo da Prática:**

- Configurar o ambiente de desenvolvimento React Native;
- Implementar a funcionalidade de entrada de texto em um componente React Native;
- Implementar um Componente de Lista Dinâmica (ScrollView);
- Implementar componentes React Native para exibir informações de forma dinâmica em listas;
- Empregar elementos visuais em um aplicativo React Native.

Descrição

Nesta atividade, você aprenderá a criar um componente de lista que permitirá exibir informações de forma dinâmica em um aplicativo React Native. Isso é útil quando você deseja criar uma lista que pode conter vários tipos de elementos, como texto e imagens, e que pode ser rolada verticalmente.

Material necessário para a prática

Editor de texto ou IDE sendo opções sugeridas: Notepad++, Nano Editor, VS Code;
Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera;

Procedimentos

1. Criando um Componente de Lista dinâmica:

Abra o seu editor de código (por exemplo, VS Code).

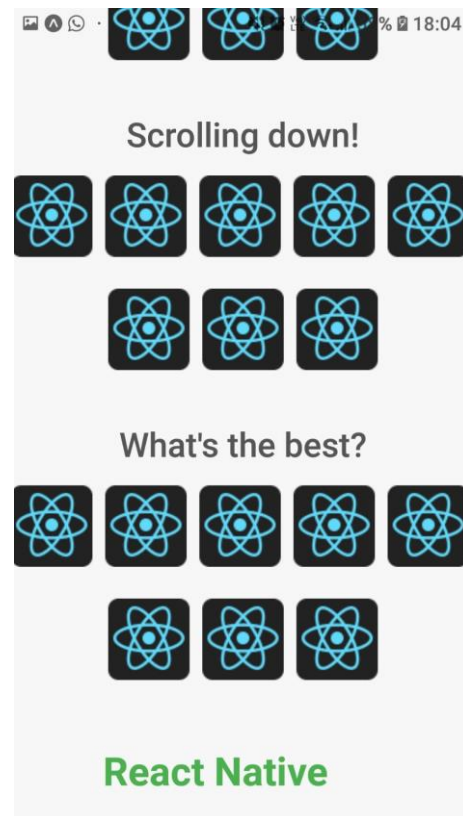
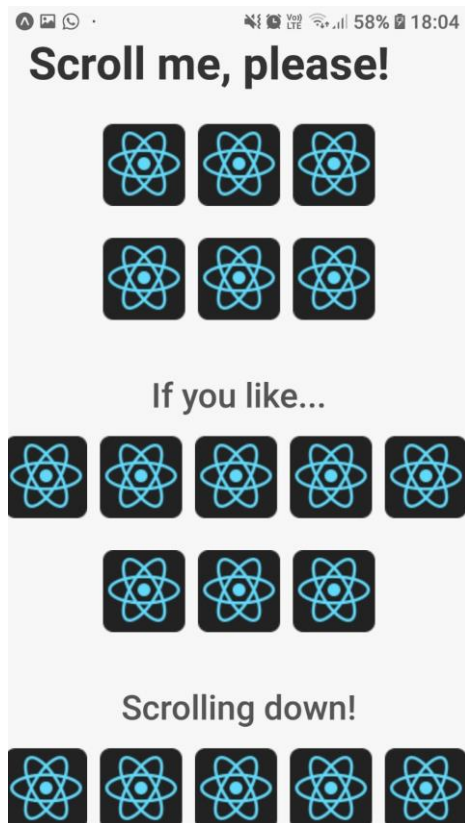
Crie um novo arquivo JavaScript (ou JSX) para o seu componente.

Copie e cole o código a seguir para criar um componente de lista que utiliza o ScrollView para rolagem vertical:

```
JS App.js M X
AppScroll > JS App.js > ...
You, 20 hours ago | 1 author (You)
1 import React from 'react'; 6.9k (gzipped: 2.7k) You, 20 hours ago +
2 import { Image, ScrollView, Text, StyleSheet, View } from 'react-native';
3
4 const Logo = {
5   uri: 'https://reactnative.dev/img/tiny_logo.png',
6   width: 64,
7   height: 64,
8 };
9
10 const App = () => {
11   <ScrollView contentContainerStyle={styles.container}>
12     <Text style={styles.title}>Scroll me, please! 😊</Text>
13     <View style={styles.logoRow}>
14       <Image source={logo} style={styles.logo} />
15       <Image source={logo} style={styles.logo} />
16       <Image source={logo} style={styles.logo} />
17     </View>
18     <View style={styles.logoRow}>
19       <Image source={logo} style={styles.logo} />
20       <Image source={logo} style={styles.logo} />
21       <Image source={logo} style={styles.logo} />
22     </View>
23
24     <Text style={styles.text}>If you like...</Text>
25     <View style={styles.logoRow}>
26       <Image source={logo} style={styles.logo} />
27       <Image source={logo} style={styles.logo} />
28       <Image source={logo} style={styles.logo} />
29       <Image source={logo} style={styles.logo} />
30       <Image source={logo} style={styles.logo} />
31     </View>
32     <View style={styles.logoRow}>
33       <Image source={logo} style={styles.logo} />
34       <Image source={logo} style={styles.logo} />
35       <Image source={logo} style={styles.logo} />
36     </View>
37
38     <Text style={styles.text}>Scrolling down!</Text>
39     <View style={styles.logoRow}>
40       <Image source={logo} style={styles.logo} />
41       <Image source={logo} style={styles.logo} />
42       <Image source={logo} style={styles.logo} />
43       <Image source={logo} style={styles.logo} />
44       <Image source={logo} style={styles.logo} />
45     </View>
46     <View style={styles.logoRow}>
47       <Image source={logo} style={styles.logo} />
48       <Image source={logo} style={styles.logo} />
49       <Image source={logo} style={styles.logo} />
50     </View>
51
52     <Text style={styles.text}>What's the best?</Text>
53     <View style={styles.logoRow}>
54       <Image source={logo} style={styles.logo} />
55       <Image source={logo} style={styles.logo} />
56       <Image source={logo} style={styles.logo} />
57       <Image source={logo} style={styles.logo} />
58       <Image source={logo} style={styles.logo} />
59     </View>
60
61     <View style={styles.logoRow}>
62       <Image source={logo} style={styles.logo} />
63       <Image source={logo} style={styles.logo} />
64       <Image source={logo} style={styles.logo} />
65     </View>
66
67     <Text style={styles.footer}>React Native 🚀</Text>
68   </ScrollView>
69 };
70
71 const styles = StyleSheet.create({
72   container: {
73     flexGrow: 1,
74     backgroundColor: '#7f7f7f',
75     paddingVertical: 20,
76     alignItems: 'center',
77   },
78   title: {
79     fontSize: 32,
80     fontWeight: 'bold',
81     marginBottom: 20,
82     color: '#333',
83   },
84   text: {
85     fontSize: 24,
86     marginTop: 20,
87     marginBottom: 10,
88     color: '#555',
89   },
90   footer: {
91     fontSize: 28,
92     fontWeight: 'bold',
93     marginTop: 30,
94     color: '#4CAF50',
95   },
96   logo: {
97     width: 64,
98     height: 64,
99     margin: 5,
100   },
101   logoRow: {
102     flexDirection: 'row',
103     justifyContent: 'center',
104     alignItems: 'center',
105     marginBottom: 15,
106   },
107 });
108
109 export default App;
```

Resultado da Micro Atividade 3:

Segue abaixo o resultado da execução do código que foi mencionado na parte acima. Onde Este código cria um componente de lista que contém texto e imagens.



Micro atividade 4:

4º Criando o visualizador de listas

- **Objetivo da Prática:**

- Configurar o ambiente de desenvolvimento React Native;
- Implementar a funcionalidade de entrada de texto em um componente React Native;
- Implementar um Componente de Lista Dinâmica (ScrollView);
- Implementar componentes React Native para exibir informações de forma dinâmica em listas;
- Empregar elementos visuais em um aplicativo React Native.

Descrição

Como parte desta atividade, vamos criar um componente React Native que permitirá exibir informações de forma dinâmica em uma lista. Isso é útil quando você tem um grande conjunto de dados e deseja que apenas os itens visíveis sejam renderizados para economizar recursos.

Material necessário para a prática

Editor de texto ou IDE sendo opções sugeridas: Notepad++, Nano Editor, VS Code;

Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera.

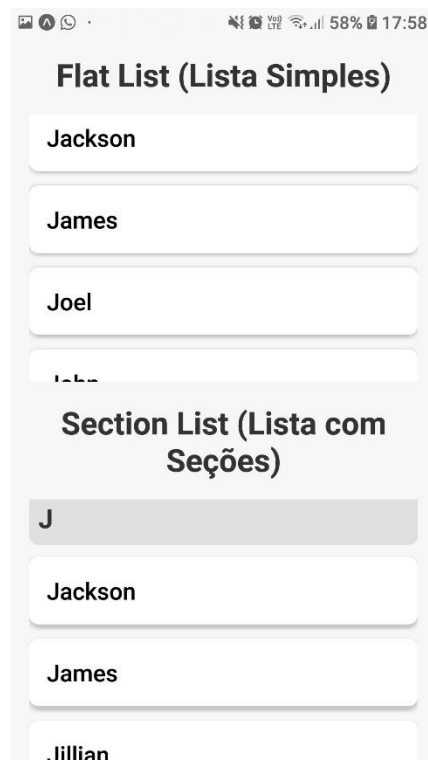
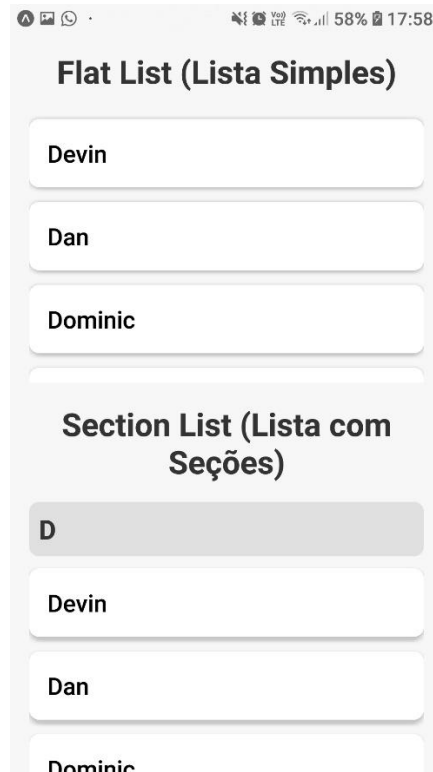
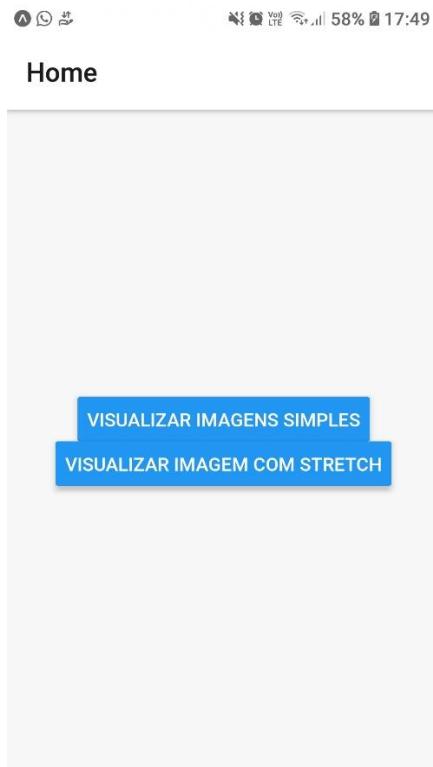
Procedimentos

Criando uma Lista Simples: Abra o seu editor de código (por exemplo, VS Code). Crie um novo arquivo JavaScript (ou JSX) para o seu componente. Copie e cole o código a seguir para criar uma lista simples usando FlatList (você pode substituir os nomes da lista de acordo com sua necessidade):

```
1 import React from 'react';
2 import { FlatList, StyleSheet, Text, View } from 'react-native';
3
4 const App = () => {
5   return (
6     <View style={styles.container}>
7       /* Lista Simples */
8       <Text style={styles.header}>Flat List (Lista Simples)</Text>
9       <FlatList
10         data={
11           [
12             { key: 'Devin' },
13             { key: 'Dan' },
14             { key: 'Dominic' },
15             { key: 'Jackson' },
16             { key: 'James' },
17             { key: 'Joel' },
18             { key: 'John' },
19             { key: 'Jillian' },
20             { key: 'Jimmy' },
21             { key: 'Julie' },
22           ]
23         }
24         renderItem={({ item }) => <Text style={styles.item}>{item.key}</Text>}
25         keyExtractor={(item) => `flatlist-${item.key}`}
26       />
27     </View>
28   );
29 };
30
31 /* Lista com Seções */
32 <Text style={styles.header}>Section List (Lista com Seções)</Text>
33 <SectionList
34   sections={[
35     {
36       title: 'D', data: ['Devin', 'Dan', 'Dominic'] },
37     {
38       title: 'J',
39       data: ['Jackson', 'James', 'Jillian', 'Jimmy', 'Joel', 'John', 'Julie'],
40     },
41   ]}
42   renderItem={({ item }) => <Text style={styles.item}>{item}</Text>}
43   renderSectionHeader={({ section }) => (
44     <Text style={styles.sectionHeader}>{section.title}</Text>
45   )}
46   keyExtractor={(item) => `sectionlist-${item}`}
47 />
48 </View>
49 );
50
51 const styles = StyleSheet.create({
52   container: {
53     flex: 1,
54     backgroundColor: '#f7f7f7',
55     padding: 16,
56     padding: 22,
57   },
58   header: {
59     fontSize: 24,
60     fontWeight: 'bold',
61     marginVertical: 16,
62     color: '#333',
63     textAlign: 'center',
64   },
65   sectionHeader: {
66     padding: 8,
67     fontSize: 20,
68     fontWeight: 'bold',
69     backgroundColor: '#e0e0e0',
70     borderRadius: 8,
71     margin: 4,
72     color: '#333',
73   },
74   item: {
75     padding: 15,
76     fontSize: 18,
77     backgroundColor: '#fff',
78     borderRadius: 8,
79     marginVertical: 6,
80     shadowColor: '#000',
81     shadowOffset: { width: 0, height: 2 },
82     shadowOpacity: 0.1,
83     shadowRadius: 4,
84     elevation: 3,
85   },
86 });
87
88 export default App;
```

Resultado da Micro Atividade 4:

Segue abaixo o resultado da execução do código que foi
Mencionado na parte acima. Este código cria uma lista simples de nomes.



Micro atividade 5:

👉 **5º Empregar imagens, seja para exibir gráficos, ícones, fotos ou outros elementos visuais em um aplicativo React Native**

- **Objetivo da Prática:**

- Configurar o ambiente de desenvolvimento React Native;
- Implementar a funcionalidade de entrada de texto em um componente React Native;
- Implementar um Componente de Lista Dinâmica (ScrollView);
- Implementar componentes React Native para exibir informações de forma dinâmica em listas;
- Empregar elementos visuais em um aplicativo React Native.

Nesta atividade, você aprenderá a incorporar imagens em um aplicativo React Native. Imagens são usadas para exibir gráficos, ícones, fotos e outros elementos visuais em um aplicativo. Vamos seguir alguns passos simples para realizar esta tarefa.

Material necessário para a prática

Editor de texto ou IDE sendo opções sugeridas: Notepad++, Nano Editor, VS Code;
Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera;

Procedimentos

1. Exibindo Imagens de Diferentes Fontes:

Abra o seu editor de código (por exemplo, VS Code).

Crie um novo arquivo JavaScript (ou JSX) para o seu componente.

Copie e cole o código a seguir para exibir imagens de diferentes fontes, incluindo armazenamento local, rede e um esquema 'data':

Codigo App.js

```
JS App.js M X
AppImagens > JS App.js > [App]
You, 1 hour ago | 1 author (You)
1 import React from 'react'; 6.9k (gzipped: 2.7k)
2 import { NavigationContainer } from '@react-navigation/native'; 37.1k (gzipped: 12.9k)
3 import { createStackNavigator } from '@react-navigation/stack';
4 import { Button, View, StyleSheet } from 'react-native';
5 import DisplayAnImage from './DisplayAnImage.js';
6 import DisplayAnImageWithStyle from './DisplayAnImageWithStyle.js';
7
8 const Stack = createStackNavigator();
9
10 const HomeScreen = ({ navigation }) => {
11   return (
12     <View style={styles.container}>
13       <Button
14         title="Visualizar Imagens Simples"
15         onPress={() => navigation.navigate('Imagens Simples')}
16       />
17       <Button
18         title="Visualizar Imagem com Stretch"
19         onPress={() => navigation.navigate('Imagem com Stretch')}
20         style={styles.button}
21       />
22     </View>
23   );
24 };
25
26 const App = () => {
27   return (
28     <NavigationContainer>
29       <Stack.Navigator initialRouteName="Home">
30         <Stack.Screen name="Home" component={HomeScreen} />
31         <Stack.Screen name="Imagens Simples" component={DisplayAnImage} />
32         <Stack.Screen name="Imagem com Stretch" component={DisplayAnImageWithStyle} />
33       </Stack.Navigator>
34     </NavigationContainer>
35   );
36 };
37
38 const styles = StyleSheet.create({
39   container: {
40     flex: 1,
41     justifyContent: 'center',
42     alignItems: 'center',
43     backgroundColor: '#f7f7f7',
44   },
45   button: {
46     marginTop: 20,
47   },
48 });
49
50 export default App; You, 1 hour ago • Uncommitted changes
```

Codigo DisplayAnimage.js

```

35 DisplayAnImage.js X
Appimagens > JS DisplayAnImage.js @ default
1 import React from 'react'; // 6.9K (gipped: 2.7k)
2 import { View, Image, StyleSheet, Text } from 'react-native';
3
4 const DisplayAnImage = () => {
5   return (
6     <View style={styles.container}>
7       <Text style={styles.title}>Imagem Local</Text>
8       <Image
9         style={styles.imageFull}
10        source={require('./assets/000111.png')} // Atualizado
11        resizeMode="contain" // Garante que a imagem seja exibida inteira
12      />
13
14      <Text style={styles.title}>Imagem da Internet</Text>
15      <Image
16        style={styles.image}
17        source={{
18          uri: 'https://reactnative.dev/img/tiny_logo.png',
19        }}
20      />
21
22      <Text style={styles.title}>Imagem Base64</Text>
23      <Image
24        style={styles.imageLarge}
25        source={{
26          uri: `data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAADMAAAMAAQAAAAIAAQAQAAQAAAExBFRW`
27            + `NRtZ2d0fYzQ0bWVmdjcnVzaE81SFMAAABQSURBVGcjo7dszCQBACAR8v2`
28            + `ab8EEQNHF1mGSy-VLyuDQYGGBVGGBYGGBYGGBvcvDqYGBgmhivGYGBg`
29            + `YGBYGGBYGGBYGGBmQP/eMrc5UTVAAAAABJRUSEnkJgg==`,
30        }}
31      />
32    </View>
33  );
34 };
35

```

Codigo DisplayAnimageWithStyles.js

```

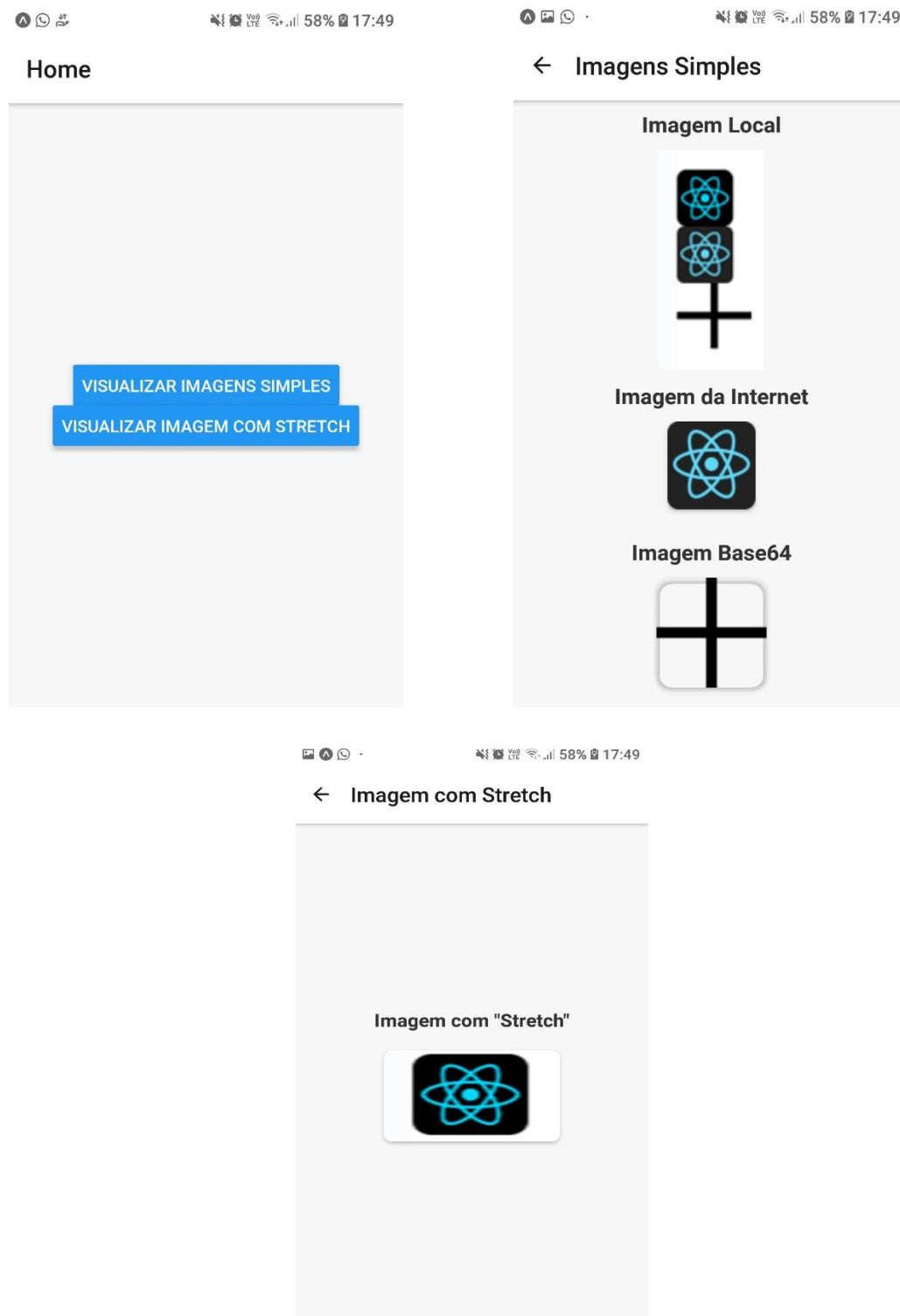
1  import React from 'react'; 6.9k (gzipped: 2.7k)
2  import { View, Image, StyleSheet, Text } from 'react-native';
3
4  const DisplayAnImageWithStyle = () => {
5    return (
6      <View style={styles.container}>
7        <Text style={styles.title}>Imagem com "Stretch"</Text>
8        <Image
9          style={styles.stretch}
10         source={require('./assets/000222.png')} // Atualizado
11        />
12      </View>
13    );
14  };
15
16  const styles = StyleSheet.create({
17    container: {
18      flex: 1,
19      backgroundColor: '#f7f7f7',
20      justifyContent: 'center',
21      alignItems: 'center',
22      padding: 20,
23    },
24    title: {
25      fontSize: 18,
26      fontWeight: 'bold',
27      marginBottom: 20,
28      color: '#333',
29      textAlign: 'center',
30      flexWrap: 'wrap',
31      width: '90%',
32      alignSelf: 'center',
33    },
34    stretch: {
35      width: 180,
36      height: 100,
37      resizeMode: 'stretch',
38      borderRadius: 8,
39      shadowColor: '#000',
40      shadowOffset: { width: 0, height: 2 },
41      shadowOpacity: 0.1,
42      shadowRadius: 4,
43      elevation: 3,
44    },
45  });
46
47  export default DisplayAnImageWithStyle;

```

Resultado da Micro Atividade 5:

Após seguir esses passos, você terá incorporado imagens em seu aplicativo React Native e aprendido a aplicar estilos às imagens para personalizá-las de acordo com suas necessidades. Isso é fundamental para criar interfaces de usuário interativas e atraentes em seu aplicativo.

A seguir você tem a imagem resultado da primeira parte da atividade.



Missão Prática | Vamos criar um App! 📱

Segue Abaixo o Código Referente ao Exemplo do Aplicativo do Gato
Que foi sugerido como Exemplo o mesmo teve algumas alterações para
Uma interação com o usuário:

App.js

```
1  import React from 'react'; 6.9k (gzipped: 2.7k)
2  import { SafeAreaView, View, StyleSheet } from 'react-native';
3  import Cat from './Cat';
4  import CatApp from './CatApp';
5  import CatInput from './CatInput';
6
7  const App = () => {
8    <SafeAreaView style={styles.container}>
9      /* Linha 1: Texto e Imagem Lado a Lado */
10     <View style={styles.row}>
11       <View style={styles.box}>
12         <Cat name="Maru" />
13       </View>
14       <View style={styles.box}>
15         <CatApp />
16       </View>
17     </View>
18
19     /* Linha 2: Interação com Input */
20     <View style={styles.fullWidthBox}>
21       <CatInput />
22     </View>
23   </SafeAreaView>
24 };
25
26 const styles = StyleSheet.create({
27   container: {
28     flex: 1,
29     padding: 16,
30     justifyContent: 'space-around',
31     backgroundColor: '#f9f9f9',
32   },
33   row: {
34     flexDirection: 'row',
35     justifyContent: 'space-around',
36     alignItems: 'center',
37     flexWrap: 'wrap', // Garante que os elementos quebrem linha em telas menores
38     marginBottom: 20,
39   },
40   box: {
41     flex: 1,
42     marginHorizontal: 8,
43     justifyContent: 'center',
44     alignItems: 'center',
45     borderWidth: 1,
46     borderColor: '#ccc',
47     borderRadius: 10,
48     padding: 16,
49     minWidth: '45%', // Ocupa 45% da largura para telas maiores
50     maxWidth: '100%', // Ocupa a largura total em telas menores
51     height: 200, // Altura fixa para uniformidade
52     backgroundColor: '#fff',
53   },
54   fullWidthBox: {
55     width: '90%', // Ocupa 90% da largura da tela
56     justifyContent: 'center',
57     alignItems: 'center',
58     borderWidth: 1,
59     borderColor: '#ccc',
60     borderRadius: 10,
61     padding: 16,
62     backgroundColor: '#fff',
63     alignSelf: 'center',
64   },
65 });
66
67 export default App;
```

Cat.js

```
1  import React from 'react'; 6.9k (gzipped: 2.7k)
2  import { Text, View, StyleSheet } from 'react-native';
3
4  const Cat = ({ name }) => {
5    <View style={styles.container}>
6      <Text style={styles.text}>Hello, I am {name}!</Text>
7    </View>
8  };
9
10 const styles = StyleSheet.create({
11   container: {
12     alignItems: 'center',
13   },
14   text: {
15     fontSize: 18,
16     color: '#333',
17     textAlign: 'center',
18   },
19 });
20
21 export default Cat;
```

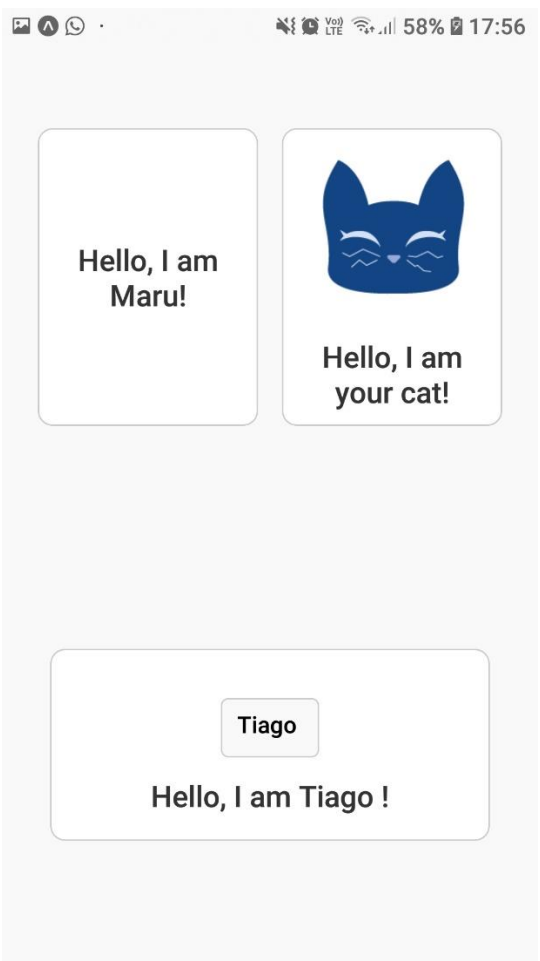
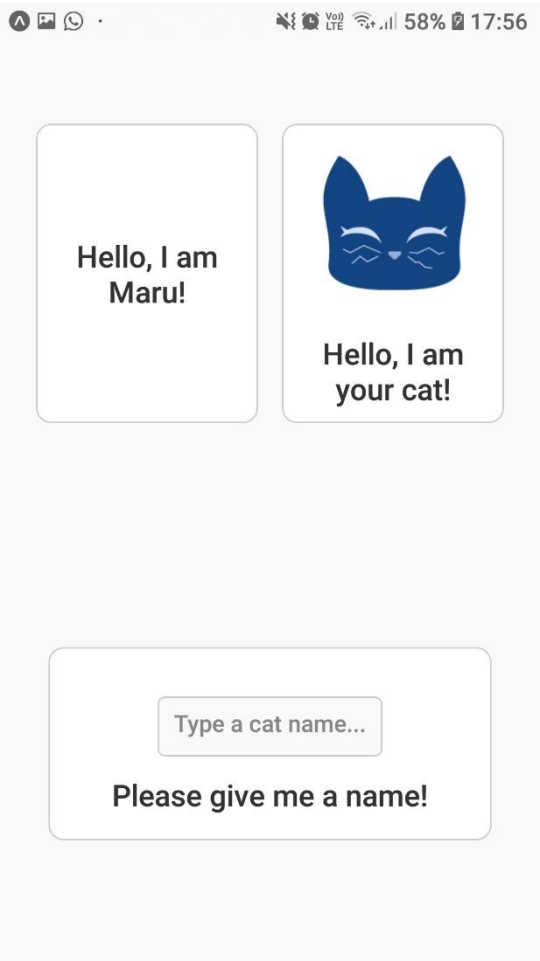
CatApp.js

```
JS CatApp.js U X
AppCat > JS CatApp.js > ...
1 import React from 'react'; 6.9k (gzipped: 2.7k)
2 import { Text, View, Image, StyleSheet } from 'react-native';
3
4 const CatApp = () => (
5   <View style={styles.container}>
6     <Image
7       source={{ uri: 'https://reactnative.dev/docs/assets/p_cat1.png' }}
8       style={styles.image}
9     />
10    <Text style={styles.text}>Hello, I am your cat!</Text>
11  </View>
12 );
13
14 const styles = StyleSheet.create({
15   container: {
16     alignItems: 'center',
17   },
18   image: {
19     width: 120,
20     height: 120,
21     marginBottom: 10,
22   },
23   text: {
24     fontSize: 18,
25     color: '#333',
26     textAlign: 'center',
27   },
28 });
29
30 export default CatApp;
31
```

CatInput.js

```
JS CatInput.js U X
AppCat > JS CatInput.js > [0] default
1 import React, { useState } from 'react'; 6.9k (gzipped: 2.7k)
2 import { Text, TextInput, View, StyleSheet } from 'react-native';
3 const CatInput = () => {
4   const [name, setName] = useState('');
5   return (
6     <View style={styles.container}>
7       <TextInput
8         style={styles.input}
9         value={name}
10        onChangeText={(text) => setName(text)}
11        placeholder="Type a cat name..."
12        placeholderTextColor="#888"
13      />
14      <Text style={styles.output}>
15        {name === '' ? 'Please give me a name!' : `Hello, I am ${name}!`}
16      </Text>
17    </View>
18  );
19 };
20 const styles = StyleSheet.create({
21   container: {
22     alignItems: 'center',
23     marginTop: 16,
24   },
25   input: {
26     height: 40,
27     borderColor: '#ccc',
28     borderWidth: 1,
29     width: '80%',
30     paddingHorizontal: 10,
31     borderRadius: 5,
32     backgroundColor: '#f9f9f9',
33   },
34   output: {
35     marginTop: 12,
36     fontSize: 18,
37     color: '#333',
38     textAlign: 'center',
39   },
40 });
41 export default CatInput;
```


Resultado do AppCat:



Missão Prática | Vamos criar um App!

Nessa atividade revisaremos tudo o que utilizamos nas micro atividades anteriores. Além disso, veremos também como o React Native responde a eventos e qual lógica de manipulação será utilizada neste exercício.

Contextualização

A empresa "Meeting" busca criar um aplicativo móvel eficaz para o cadastro de fornecedores, com listas e imagens de alta qualidade, economizando recursos e proporcionando uma excelente experiência ao usuário. A escolha da tecnologia React Native é crucial para estabelecer uma presença sólida no mercado móvel. Nesta atividade, você aprenderá os princípios básicos do React Native.

Requisitos Funcionais:

Cadastro de Fornecedores: O aplicativo deve permitir o cadastro de fornecedores, incluindo informações detalhadas, como nome, endereço, contato e categorias de produtos fornecidos. Essas informações serão exibidas utilizando componentes como `<Text>`, `<TextInput>`, e `<Image>`.

Listagem de Fornecedores: Deve ser possível visualizar uma lista de fornecedores cadastrados, com opções de pesquisa e filtragem com base em critérios como categoria ou localização. A lista de fornecedores será exibida utilizando componentes como `<Text>` e `<Image>`.

Associação de Imagens: O aplicativo deve permitir a associação de imagens aos perfis dos fornecedores. Os usuários devem poder fazer o upload de logotipos ou fotos relacionadas ao fornecedor, utilizando o componente `<Image>`.

Experiência de Usuário Intuitiva: A interface do aplicativo deve ser intuitiva e fácil de usar, garantindo que os usuários possam navegar, adicionar e editar informações de forma eficiente. Isso será alcançado usando componentes como `<Text>`, `<TextInput>`, e `<Image>`.

Código da parte inicial do app onde mostra o cadastro do fornecedor
Onde tem as opções para colocar as informações do cadastro do fornecedor.

```

App.js M CadastroFornecedores.js U X
AppMeeting > components > CadastroFornecedores.js > CadastroFornecedor > pickImage
1 import React, { Component } from 'react';
2 import { Alert, Image, StyleSheet, TextInput, View, TouchableOpacity, Text } from 'react-native';
3 import * as ImagePicker from 'expo-image-picker';
4
5 class CadastroFornecedor extends Component {
6   state = {
7     nome: '',
8     endereco: '',
9     contato: '',
10    categorias: '',
11    descricao: '',
12    imagem: null,
13  };
14
15  handleCadastro = () => {
16    const { nome, endereco, contato, categorias, descricao, imagem } = this.state;
17    if ([nome, endereco, contato, categorias, descricao].length > 0) {
18      this.showAlert('Erro', 'Por favor, preencha todos os campos.');
```

Resultado do Codigo Acima:



Cadastro de Fornecedores

Nome do Fornecedor

Endereço

Contato

Categoria

Descrição do Produto

Imagem Cadastrar

Fornecedore



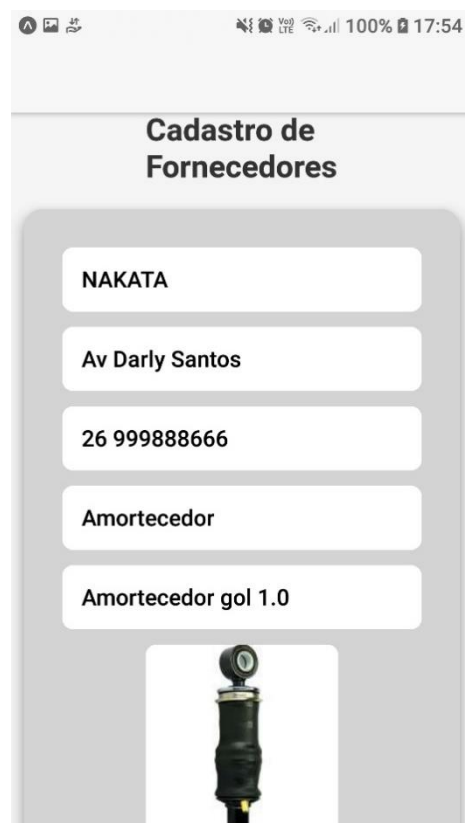
Sucesso

Fornecedor cadastrado com sucesso!

OK

Imagem Cadastrar

Fornecedore



Cadastro de Fornecedores


NAKATA

Av Darly Santos

26 999888666

Amortecedor

Amortecedor gol 1.0



Segue abaixo o Código Referente a Parte de Fornecedores Logo após serem adicionados eles são direcionados para Outra tela.

```
AppMeeting > components > FornecedoresItaja > ListaFornecedores > fornecedores.map() callback
1 import React, { useState } from 'react';
2 import {
3   Alert,
4   Image,
5   Modal,
6   ScrollView,
7   StyleSheet,
8   Text,
9   TouchableOpacity,
10  View,
11 } from 'react-native';
12
13 const ListaFornecedores = ({ fornecedores, onRemove }) => {
14   const [modalVisible, setModalVisible] = useState(false);
15   const [fornecedorSelecionado, setFornecedorSelecionado] = useState(null);
16
17   const handleDetalhes = (fornecedor) => {
18     setFornecedorSelecionado(fornecedor);
19     setModalVisible(true);
20   };
21
22   const handleDelete = (fornecedor) => {
23     Alert.alert('Confirmar exclusão', 'Você realmente deseja excluir este fornecedor?', [
24       { text: 'Cancelar', style: 'cancel' },
25       { text: 'Excluir', onPress: () => onRemove(fornecedor.id) },
26     ]);
27   };
28
29   return (
30     <View style={styles.container}>
31       <ScrollView>
32         {fornecedores.map((fornecedor) => (
33           <View key={fornecedor.id} style={styles.fornecedorContainer}>
34             <Image source={{ uri: fornecedor.imagem }} style={styles.imagemFornecedor} />
35             <View style={styles.infoContainer}>
36               <Text style={styles.nomeFornecedor}>{fornecedor.nome}</Text>
37               <Text style={styles.detalheFornecedor}>Categoria: {fornecedor.categorias}</Text>
38               <TouchableOpacity>
39                 <Text style={styles.actionButton}>
40                   <Text style={styles.actionText}>Detalhes</Text>
41                 </TouchableOpacity>
42               <TouchableOpacity>
43                 <Text style={styles.actionButton}>
44                   <Text style={styles.actionText}>Apagar</Text>
45                 </TouchableOpacity>
46             </View>
47           </View>
48         ))}
49       </ScrollView>
50
51       {fornecedorSelecionado && (
52         <Modal visible={modalVisible} animationType="slide" transparent={true}>
53           <View style={styles.modalContainer}>
54             <View style={styles.modalContent}>
55               <Image
56                 source={{ uri: fornecedorSelecionado.imagem }}
57                 style={styles.modalImage}
58               />
59               <Text style={styles.modalText}>
60                 Nome: {fornecedorSelecionado.nome}
61               </Text>
62               <Text style={styles.modalText}>
63                 Endereço: {fornecedorSelecionado.endereco}
64               </Text>
65               <Text style={styles.modalText}>
66                 Contato: {fornecedorSelecionado.contato}
67               </Text>
68               <Text style={styles.modalText}>
69                 Categoria: {fornecedorSelecionado.categorias}
70               </Text>
71               <Text style={styles.modalText}>
72                 Descrição: {fornecedorSelecionado.descricao}
73               </Text>
74               <TouchableOpacity>
75                 <Text style={styles.actionButton}>
76                   <Text style={styles.actionText}>Fechar</Text>
77                 </TouchableOpacity>
78             </View>
79           </View>
80         </Modal>
81       )}
82     </View>
83   );
84
85   const styles = StyleSheet.create({
86     container: { flex: 1, padding: 16, backgroundColor: '#F0F0F0' },
87     fornecedorContainer: {
88       flex: 1,
89       margin: 16,
90       padding: 16,
91       backgroundColor: '#FFF',
92       borderRadius: 12,
93       elevation: 3,
94     },
95     imagemFornecedor: { width: 80, height: 80, borderRadius: 50 },
96     infoContainer: { flex: 1, margin: 16 },
97     nomeFornecedor: { fontWeight: 'bold', fontSize: 16, margin: 8 },
98     detalheFornecedor: { fontSize: 14, color: '#555', margin: 4 },
99     actionButton: {
100       backgroundColor: '#007BFF',
101       padding: 10,
102       borderRadius: 8,
103       margin: 8,
104     },
105     actionText: { color: '#FFF', textAlign: 'center', fontWeight: 'bold' },
106     modalContainer: { flex: 1, justifyContent: 'center', alignItems: 'center', backgroundColor: 'rgba(0,0,0,0.5)' },
107     modalContent: {
108       width: '90%',
109       padding: 20,
110       borderRadius: 12,
111       backgroundColor: '#FFF',
112       alignItems: 'center',
113     },
114     modalImage: { width: 150, height: 150, borderRadius: 75, margin: 20 },
115     modalText: { fontSize: 16, color: '#333', margin: 10 },
116   });
117
118   export default ListaFornecedores;
119 }
```

Segue Abaixo o Resultado Referente ao Codigo Acima:

