Aula: Criando um Componente de Tabela no Expo com TypeScript

Objetivo da Aula

Aprender a criar um componente de tabela simples, utilizando o framework Expo (React Native) com TypeScript, para exibir dados de forma organizada em linhas e colunas.

Pré-requisitos

• Ter o Node.js instalado.

• Ter o Expo CLI instalado (*npm install -g expo-cli*

• Conhecimentos básicos de:

- React Native

- TypeScript

• Um editor de código (recomendado: VS Code).

Criando o Projeto no Expo

1. Criar um novo projeto:

expo init TabelaExpo

Escolha a opção "Blank (TypeScript)" para já vir configurado com TypeScript.

Acesse a pasta do projeto:

cd TabelaExpo

Execute o projeto:

expo start

O que vamos construir?

Um componente de tabela que recebe dados como props e exibe em formato de linhas e colunas, com títulos.

Estrutura do Componente

Vamos criar uma pasta chamada components:

mkdir components

Dentro dela, crie o arquivo:

Tabela.tsx

Código do Componente Tabela

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

type TabelaProps = {
  headers: string[];
  data: string[][];
};

const Tabela: React.FC<TabelaProps> = ({ headers, data }) => {
  return (
    <ScrollView horizontal>
      <View>
        <View style={styles.headerRow}>
          {headers.map((header, index) => (
            <View key={index} style={styles.cell}>
              <Text style={styles.headerText}>{header}</Text>
            </View>
          ))}
        </View>

        {data.map((row, rowIndex) => (
          <View key={rowIndex} style={styles.dataRow}>
            {row.map((cell, cellIndex) => (
              <View key={cellIndex} style={styles.cell}>
                <Text>{cell}</Text>
              </View>
            ))}
          </View>
        ))}
```

```
      </View>
    </ScrollView>
  );
};

const styles = StyleSheet.create({
  headerRow: {
    flexDirection: 'row',
    backgroundColor: '#d9d9d9',
    borderTopWidth: 1,
    borderBottomWidth: 1,
    borderColor: '#999',
  },
  dataRow: {
    flexDirection: 'row',
    borderBottomWidth: 1,
    borderColor: '#eee',
  },
  cell: {
    padding: 10,
    minWidth: 100,
    justifyContent: 'center',
    alignItems: 'center',
    borderRightWidth: 1,
    borderColor: '#ccc',
  },
  headerText: {
    fontWeight: 'bold',
  },
});
```

```
60
61 export default Tabela;
```

Usando o Componente na Tela Principal

Abra o arquivo App.tsx e substitua o conteúdo por:

```
1  import React from 'react';
2  import { SafeAreaView, StyleSheet, Text } from 'react-native';
3  import Tabela from './components/Tabela';
4
5  export default function App() {
6    const headers = ['Nome', 'Idade', 'Cidade'];
7    const data = [
8      ['Maria', '28', 'São Paulo'],
9      ['João', '35', 'Rio de Janeiro'],
10     ['Ana', '22', 'Belo Horizonte'],
11     ['Carlos', '40', 'Curitiba'],
12   ];
13
14   return (
15     <SafeAreaView style={styles.container}>
16       <Text style={styles.title}>Tabela de Pessoas</Text>
17       <Tabela headers={headers} data={data} />
18     </SafeAreaView>
19   );
20 }
21
22 const styles = StyleSheet.create({
23   container: {
24     flex: 1,
25     marginTop: 50,
26     padding: 10,
```

```
  },
  title: {
    fontSize: 24,
    fontWeight: 'bold',
    marginBottom: 20,
    textAlign: 'center',
  },
});
```

Resultado Esperado

• Uma tabela simples com cabeçalho e linhas de dados.

• A tabela pode ser rolada horizontalmente se as colunas forem muito largas.

O que aprendemos?

• Criar um componente funcional com TypeScript no React Native.

• Usar props tipadas (headers e data).

• Trabalhar com listas usando .map() para gerar elementos dinamicamente.

• Estilizar uma tabela simples com StyleSheet.

Próximos passos (Sugestões)

• Adicionar ordenação por colunas.

• Permitir rolagem vertical.

• Aplicar estilos melhores (cores, bordas, fontes).

• Inserir funcionalidades como clique na linha ou coluna.