



## **Mundo 4**

### **Missão Prática Nível 1**

**429 Polo Centro – Porto Alegre – RS**

**Desenvolvimento Full Stack – 2023.1 – 4º Semestre Letivo**

<https://github.com/Jefferson-sandoval/missao-pratica-nivel-1-mundo-4>

#### **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.



# Cadastro e Listagem de Fornecedores em React Native

---

Os códigos fornecidos criam um app simples em React Native para o cadastro e listagem de fornecedores. Vou detalhar o papel de cada componente e também explicar algumas das funções e conceitos importantes utilizados.

## 1. Index.tsx

Este componente é a tela principal do app, onde os fornecedores são cadastrados e exibidos. Ele utiliza a função `useState` do React para gerenciar a lista de fornecedores.

- `useState`: `useState([])` é usado para criar uma variável de estado chamada `suppliers`, que armazena a lista de fornecedores. O estado inicial é um array vazio.
- `addSupplier`: Esta função é responsável por adicionar um novo fornecedor à lista de fornecedores. Ela usa a função `setSuppliers` para atualizar o estado. A sintaxe `setSuppliers((prevSuppliers) => [...prevSuppliers, newSupplier])` significa que estamos pegando o estado anterior (`prevSuppliers`) e criando um novo array com todos os fornecedores antigos e o novo fornecedor (`newSupplier`) no final. Essa função é passada como uma prop para o componente `SupplierForm`, que vai chamá-la sempre que um novo fornecedor for cadastrado.

## 2. SupplierForm.js

Esse componente cuida do formulário que o usuário preenche para cadastrar um novo fornecedor. Ele usa quatro variáveis de estado para controlar os campos do formulário: nome, endereço, contato e categoria.

- `useState` para campos de formulário: Cada campo tem seu próprio `useState`. Por exemplo, `const [name, setName] = useState("")` cria uma variável `name` para armazenar o valor do campo "Nome do Fornecedor" e a função `setName` para atualizá-la conforme o usuário digita.
- `handleSubmit`: Essa função é chamada quando o botão "Adicionar Fornecedor" é pressionado. Ela cria um objeto `newSupplier` com os valores dos campos e chama a função `onAddSupplier`, que foi passada de `Index.tsx`. Depois disso, os campos do formulário são limpos para o próximo cadastro.

## 3. SupplierList.js

Esse componente exibe a lista de fornecedores cadastrados. Ele utiliza o componente `FlatList` do React Native para renderizar os fornecedores.



- **FlatList:** É uma forma eficiente de renderizar longas listas em React Native. O FlatList recebe os dados (nesse caso, a lista de fornecedores) e uma função `renderItem`, que define como cada item da lista deve ser exibido.
- **KeyExtractor:** O FlatList precisa de uma chave única para cada item, então aqui usamos `keyExtractor={({item, index}) => index.toString()}` para garantir que cada fornecedor tenha uma chave baseada no índice da lista. Idealmente, seria melhor usar um ID único para cada fornecedor, mas como isso não está presente, o índice da lista serve para este propósito.

## 4. App.js

Este componente é o ponto de entrada do app. Ele basicamente junta todos os outros componentes e fornece a lógica para adicionar e listar fornecedores.

- **Estado centralizado:** O App.js gerencia o estado dos fornecedores (usando `useState`), e passa tanto a função `addSupplier` quanto a lista de fornecedores para os componentes `SupplierForm` e `SupplierList`. Isso cria uma boa separação de responsabilidades: o App.js mantém o estado, enquanto os componentes filhos cuidam de exibir o formulário e a lista.

## Explicações de Funções e Conceitos

- **`useState`:** Uma função do React que permite adicionar estado a componentes funcionais. Ele retorna um valor atual do estado e uma função para atualizá-lo. Exemplo: `const [name, setName] = useState("")` — aqui, `name` é o valor atual e `setName` é a função usada para atualizar esse valor.
- **`setSuppliers((prevSuppliers) => [...prevSuppliers, newSupplier])`:** Essa linha atualiza o estado `suppliers`. A função `prevSuppliers` representa o estado atual. Ao usar o `spread operator (...)`, estamos copiando todos os fornecedores anteriores e adicionando o novo fornecedor no final.
- **FlatList:** Um componente para renderizar listas de maneira eficiente em React Native. Ele só renderiza os itens visíveis na tela, o que melhora o desempenho em listas grandes.
- **`handleSubmit`:** Essa função pega os valores dos campos do formulário, cria um objeto `newSupplier`, e o passa de volta para o componente pai (via `onAddSupplier`), atualizando a lista de fornecedores.

## Conclusão

Cada parte do app está bem dividida e clara. A função `useState` é usada para controlar tanto os campos do formulário quanto a lista de fornecedores, e o FlatList permite uma renderização eficiente da lista. O fluxo é simples: o usuário adiciona um fornecedor, que é armazenado no estado e exibido na lista. Com isso, o app cumpre bem o objetivo de cadastrar fornecedores e mostrar essa lista de forma organizada.