```
import React, { useState } from "react";
import { Text, Button, StyleSheet, SafeAreaView } from 'react-native'

/*  CICLO DE VIDA DOS ESTADOS EM REACT:

    ESSE CONCEITO É IMPORTANTÍSSIMO DE SER ENTENDIDO!

    No react é possível usar dados passados por através dos componentes e usá-los tanto para processamento (em uma determinada função), quanto para gerar novas renderizações na tela.

    COMO ISSO ACONTECE?

    Ao receber um valor passado por propriedade, o React consegue aguardar esse valor para que ele possa ser usado e modificado segundo o nosso desejo e ainda ser renderizado na tela.

    DÁ PRA FAZER ISSO DIRETAMENTE?

    Não! O React não permite que peguemos o valor passado o modifiquemos e ainda mostremos isso na tela. Até é possível atribuir esse valor a uma variável e operar sobre uma função internamente, mas renderizar o valor na tela não.

    COMO PODEMOS FAZER PARA RENDERIZAR ENTÃO?

    Isso só é possível com o uso de Hooks.

    O QUE SÃO HOOKS?

    Hooks ou Ganchos, trazem justamente essa ideia de "puxar" funcionalidades especiais do React, esses Hooks são conhecidos pelo prefixo "use" e temos vários deles. O mais famosinho é o "useState".
    Essas funcionalidades estão atreladas ao ciclo de vida de um componente, a uma mudança de estado e outras funcionalidades do React.

    COMO FUNCIONA O USESTATE?

    O useState é um Hook (que funciona como um método podemos dizer assim) que é capaz de pegar um valor passado e retornar um array com dois valores onde:
        0 é igual ao valor passado
```

```
        1 é igual ao valor passado depois de ter sido operado por uma callback

        OBS: geralmente essas callbacks recebem o prefixo "set" (setFuncaoAlgumaCoisa)

    Vamos ver dois exemplos abaixo onde: no 1º tentamos modificar a renderização sem a utilização de um Hook e no segundo usamos o
useState para gerar uma modificação na tela.

*/

//USO SEM O HOOK
//Essa função exibe um número que inicia com 100 e tem 2 botões que deveriam aumentar ou diminuir o valor...
export function SemUseState(props) {

    let num = props.inicial //Veja que pegamos o valor passado e atribuímos sobre a variável

    const inc = () => {
        console.warn(num) //Como ela não vai mostrar valores sendo mudados, usamos um warn para ver que realmente os valores estão
sendo mudados internamente, mas não podem ser renderizados...
        num++
    }

    const dec = () => {
        console.warn(num)
        num--
    }

    return (
        <SafeAreaView style={styles.container}>
            <Text style={styles.titulo}>Uso sem o useState</Text>
            <Text style={styles.texto}>{num}</Text>
            {/* Note que o valor não muda */}
            <Button style={styles.botao} title="+" onPress={inc}/>
            <Button style={styles.botao} title="-" onPress={dec}/>
        </SafeAreaView>
    )
}
```

```jsx
//EXEMPLO USANDO HOOKS
//Essa função exibe um número que inicia com 100 e tem 2 botões que aumentam ou diminuem o valor...
export function ComUseState(props) {

    //Aqui estamos usando o Hook useState que antes foi importado da biblioteca react
    //Veja a usabilidade:
    //1º useState recebe como parâmetro o valor passado via componente
    //2º o resultado será um array com 2 valores, que destruturizamos em 2 variáveis
    //3º a primeira variável é sempre o valor atual
    //4º a segunda variável é uma função que será a única que poderá modificar o valor da primeira variável
    //5º quando o valor da variável 1 é modificado por atravé da função da variável 2 o valor da variável 1 é atualizado, se ele for operado novamente por através da função da variável 2, ele vai ser atualizado de novo, e assim sucessivamente.
    const [num, setNumero] = useState(props.inicial)

    const inc = () => {
        //Note aqui que nossa função agora recebe a função setNumero que é a única capaz de modificar o valor guardado dentro de um estado.
        setNumero(num + 1) //Não é possível fazer autoincremento ou decremento
    }
    const dec = () => {
        setNumero(num - 1) //Não é possível fazer autoincremento ou decremento
    }

    return (
        <SafeAreaView style={styles.container}>
            <Text style={styles.titulo}>Uso com o useState</Text>
            <Text style={styles.texto}>{num}</Text>
            <Button style={styles.botao} title="+" onPress={inc}/>
            <Button style={styles.botao} title="-" onPress={dec}/>
        </SafeAreaView>
    )
}

const styles = StyleSheet.create({
```

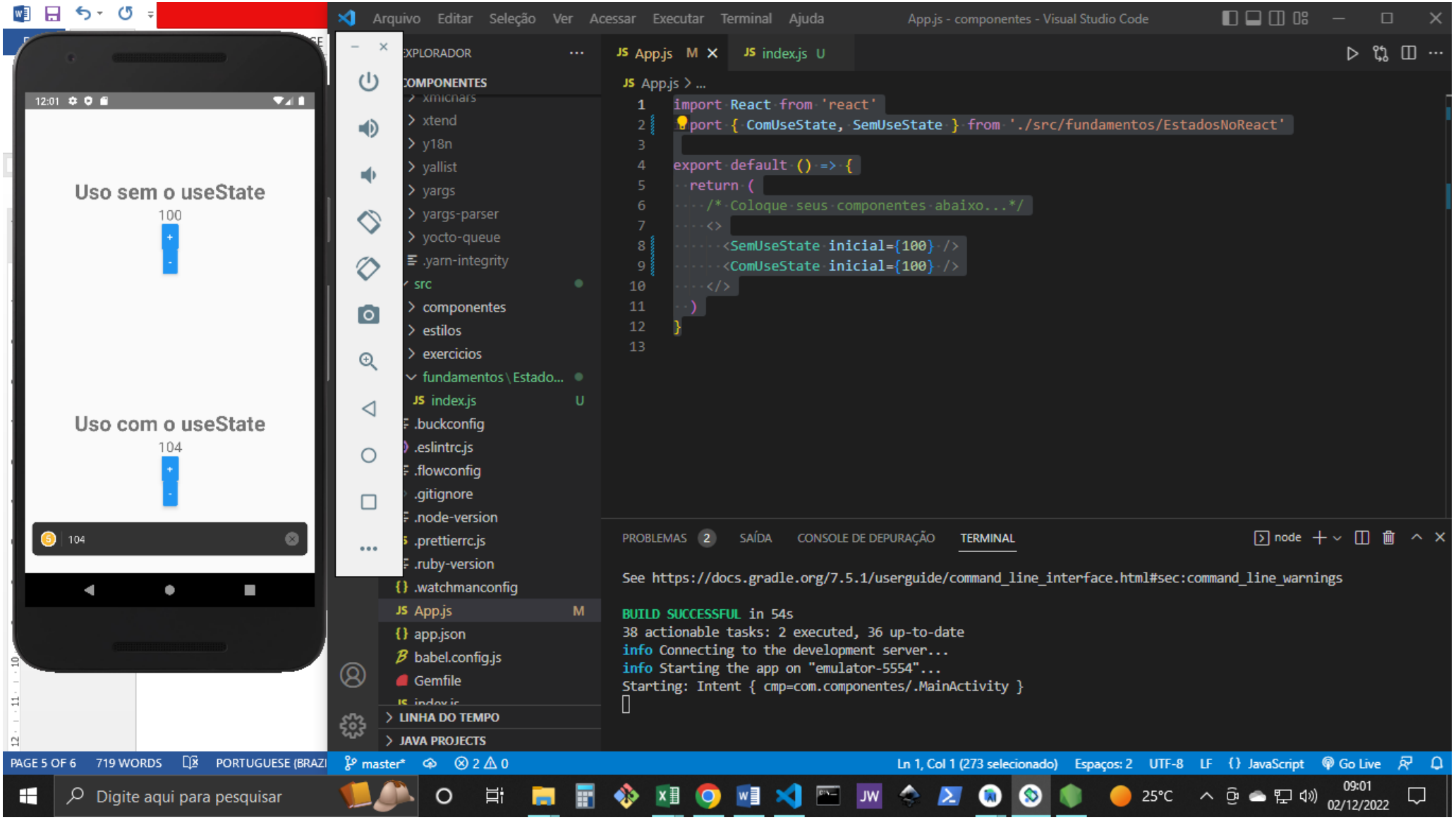
```
    titulo: {
        fontSize: 30,
        fontWeight: 'bold',
    },
    container: {
        alignItems: 'center',
        flex: 1,
        justifyContent: 'center',
        padding: 10,
        margin: 20,
    },
    texto: {
        fontSize: 20,
    },
})
```

ARQUIVO APP.JS

```
import React from 'react'
import { ComUseState, SemUseState } from './src/fundamentos/EstadosNoReact'

export default () => {
  return (
    /* Coloque seus componentes abaixo...*/
    <>
      <SemUseState inicial={100} />
      <ComUseState inicial={100} />
    </>
  )
}
```

RESULTADO DEPOIS DE APERTAR OS BOTÕES “+” 4 VEZES...



RESULTADO DEPOIS DE APERTAR OS BOTÕES “-“ 4 VEZES...