

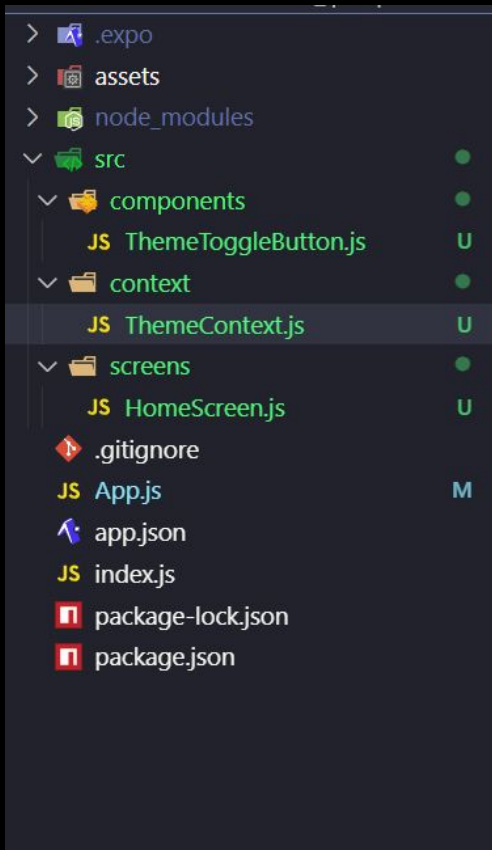
Mobile Application Development

Prof. Fernando Pinéo

Tema Claro e Tema Escuro

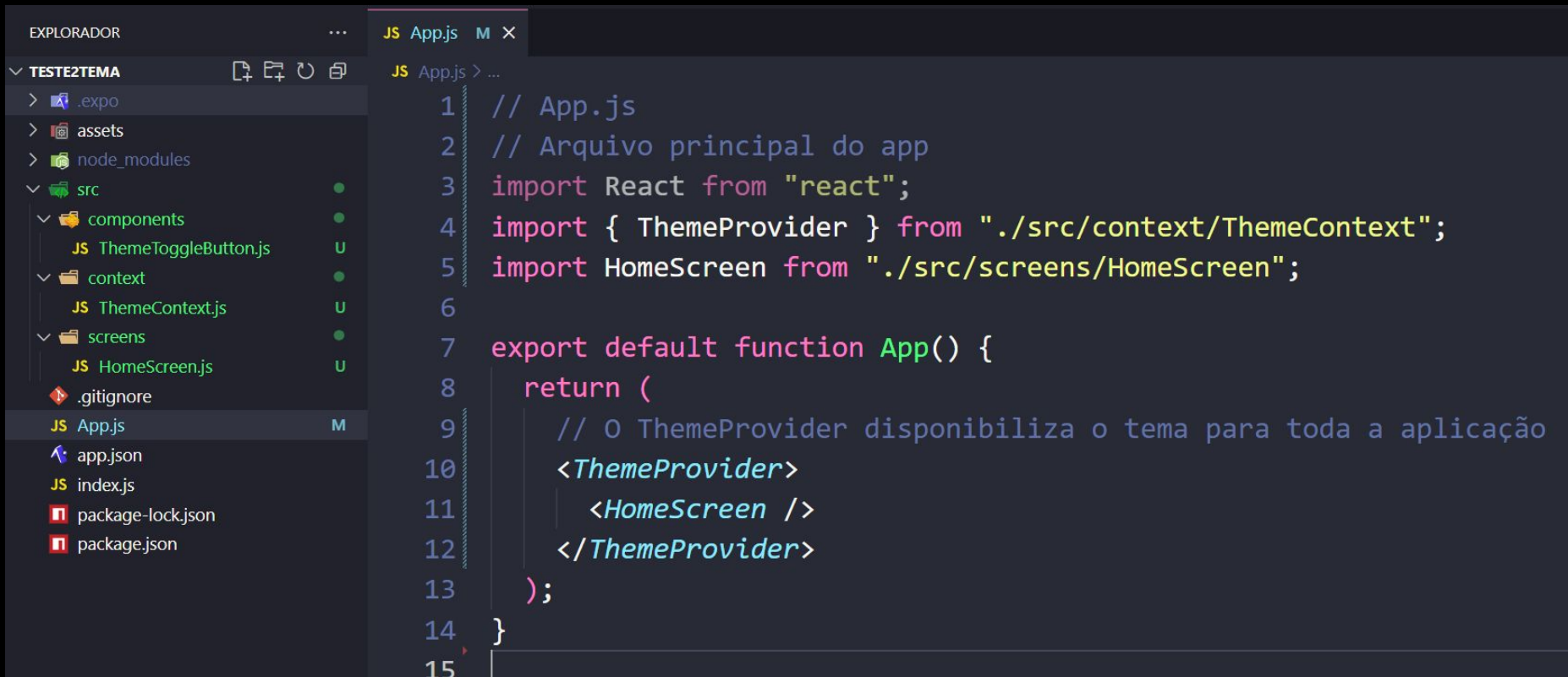
Estrutura do projeto

FIAP



No arq App.js

FIAP

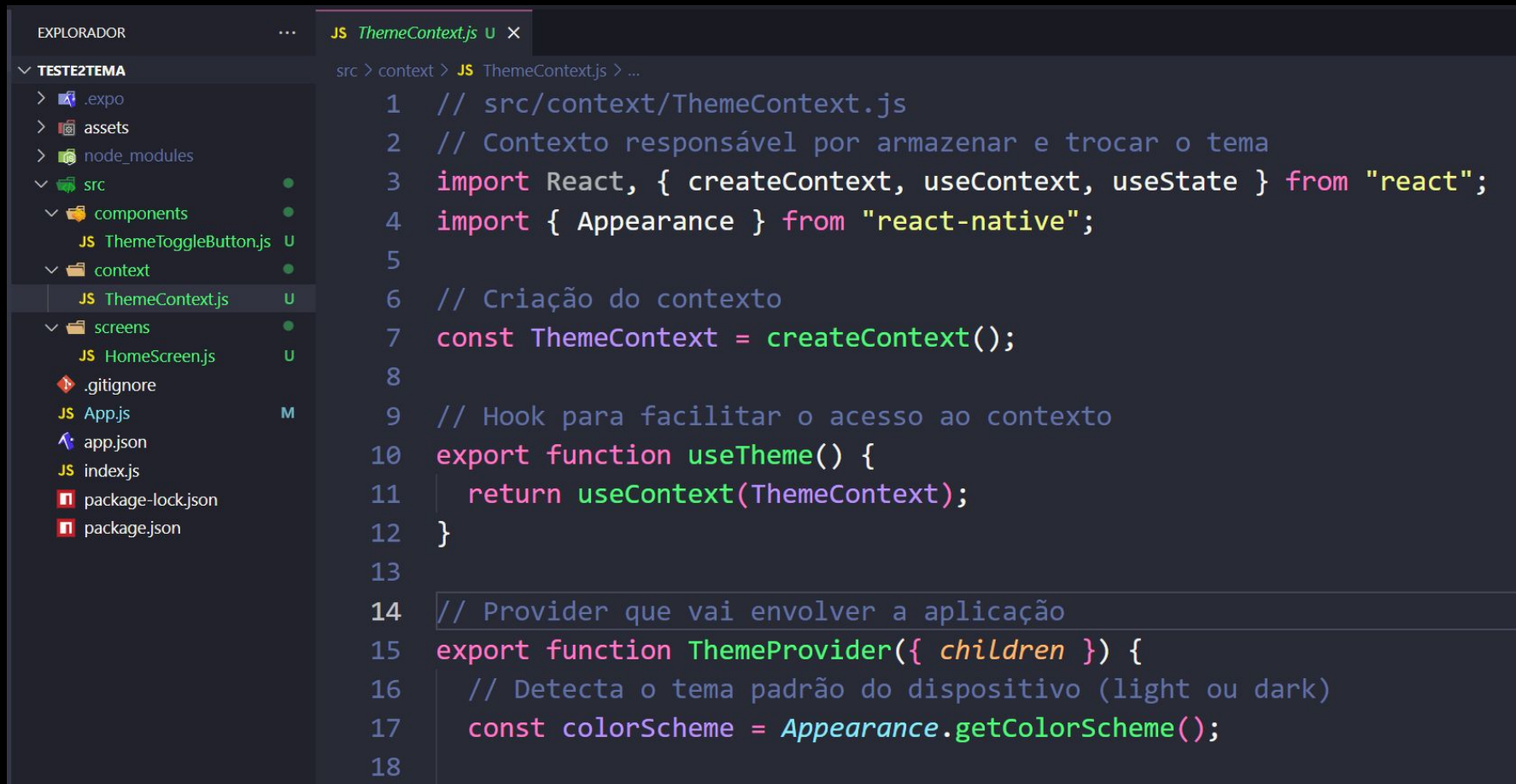


The image shows a code editor interface with a dark theme. On the left is the 'EXPLORADOR' (Explorer) sidebar showing a project structure for 'TESTE2TEMA'. The 'src' directory is expanded, showing subdirectories 'components', 'context', and 'screens', each containing a JavaScript file. The 'App.js' file is selected and highlighted. The main editor area shows the code for 'App.js', which imports React, ThemeProvider, and HomeScreen, and exports a default function App() that returns a JSX element wrapping HomeScreen with ThemeProvider.

```
1 // App.js
2 // Arquivo principal do app
3 import React from "react";
4 import { ThemeProvider } from "../src/context/ThemeContext";
5 import HomeScreen from "../src/screens/HomeScreen";
6
7 export default function App() {
8   return (
9     // O ThemeProvider disponibiliza o tema para toda a aplicação
10    <ThemeProvider>
11      <HomeScreen />
12    </ThemeProvider>
13  );
14 }
15
```

No arq ThemeContext

FIAP



The image shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer, titled 'EXPLORADOR', shows a project named 'TESTE2TEMA' with the following structure:

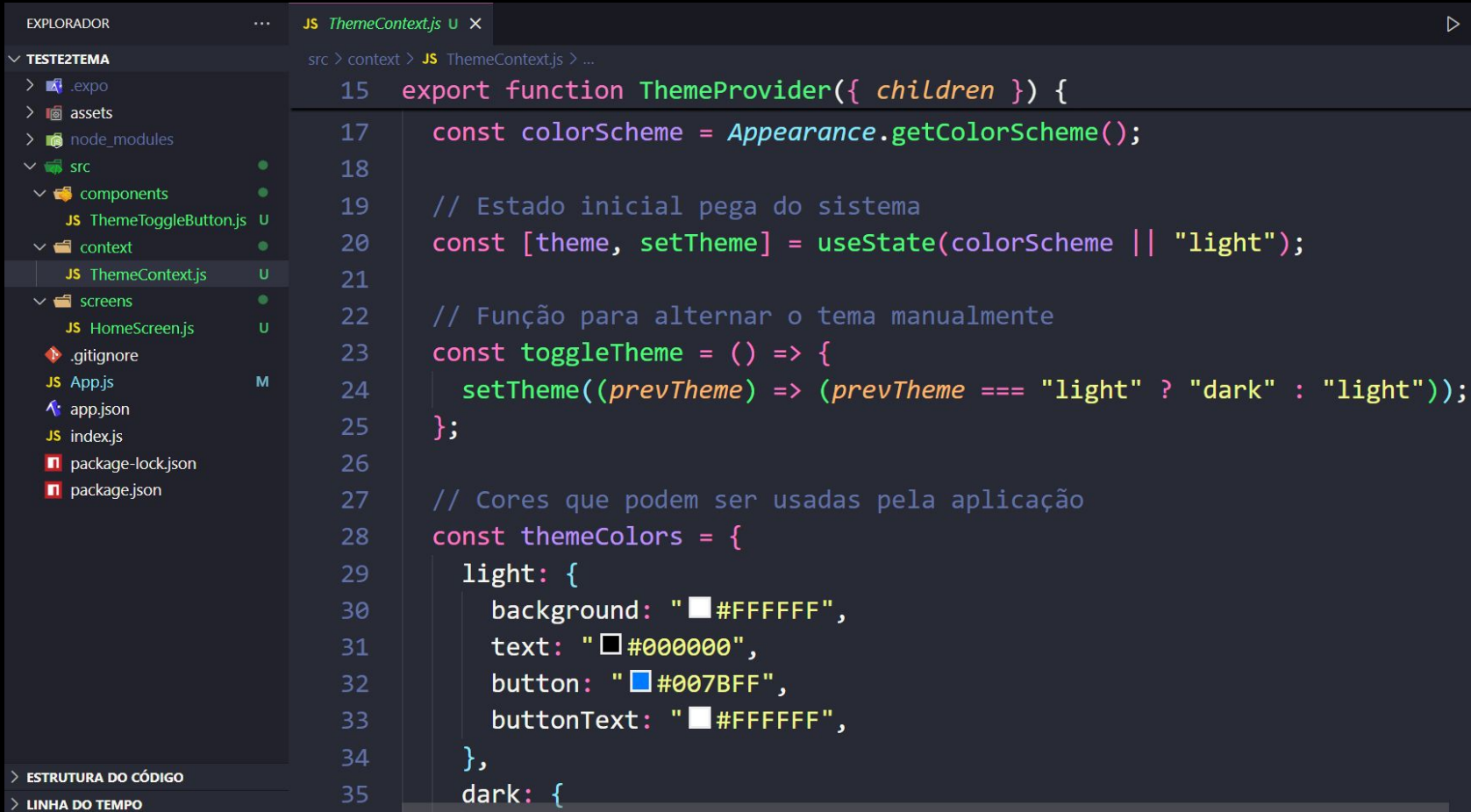
- > .expo
- > assets
- > node_modules
- ▼ src
 - ▼ components
 - JS ThemeToggleButton.js U
 - ▼ context
 - JS ThemeContext.js U
 - ▼ screens
 - JS HomeScreen.js U
- .gitignore
- JS App.js M
- app.json
- JS index.js
- package-lock.json
- package.json

The code editor shows the content of 'JS ThemeContext.js' with the following code:

```
1 // src/context/ThemeContext.js
2 // Contexto responsável por armazenar e trocar o tema
3 import React, { createContext, useContext, useState } from "react";
4 import { Appearance } from "react-native";
5
6 // Criação do contexto
7 const ThemeContext = createContext();
8
9 // Hook para facilitar o acesso ao contexto
10 export function useTheme() {
11   return useContext(ThemeContext);
12 }
13
14 // Provider que vai envolver a aplicação
15 export function ThemeProvider({ children }) {
16   // Detecta o tema padrão do dispositivo (light ou dark)
17   const colorScheme = Appearance.getColorScheme();
18 }
```

No arq ThemeContext

FIAP



```
15 export function ThemeProvider({ children }) {
17   const colorScheme = Appearance.getColorScheme();
18
19   // Estado inicial pega do sistema
20   const [theme, setTheme] = useState(colorScheme || "light");
21
22   // Função para alternar o tema manualmente
23   const toggleTheme = () => {
24     setTheme((prevTheme) => (prevTheme === "light" ? "dark" : "light"));
25   };
26
27   // Cores que podem ser usadas pela aplicação
28   const themeColors = {
29     light: {
30       background: "■ #FFFFFF",
31       text: "□ #000000",
32       button: "■ #007BFF",
33       buttonText: "■ #FFFFFF",
34     },
35     dark: {
```

No arq ThemeContext

FLANP

EXPLORADOR

TESTE2TEMA

- > .expo
- > assets
- > node_modules
- src
 - components
 - JS ThemeToggleBu... U
 - context
 - JS ThemeContext.js U
 - screens
 - JS HomeScreen.js U
- .gitignore
- JS App.js M
- app.json
- index.js
- package-lock.json
- package.json

JS ThemeContext.js U X

- src > context > JS ThemeContext.js > ThemeProvider > colors

```
15 <port function ThemeProvider({ children }) {
28   const themeColors = {
29     light: {
30       background: "#FFFFFF",
31       text: "□ #000000",
32       button: "■ #007BFF",
33       buttonText: "■ #FFFFFF",
34     },
35     dark: {
36       background: "□ #121212",
37       text: "■ #FFFFFF",
38       button: "■ #BB86FC",
39       buttonText: "□ #000000",
40     },
41   };
42
43   return (
44     <ThemeContext.Provider value={{ theme, toggleTheme, colors: themeColors[theme] }}>
45       {children}
46     </ThemeContext.Provider>
47   );
```

Tela HomeScreen.js

FIAP

EXPLORADOR

TESTE2TEMA

- > .expo
- > assets
- > node_modules
- ▼ src
 - ▼ components
 - JS ThemeToggleBu... U
 - ▼ context
 - JS ThemeContext.js U
 - ▼ screens
 - JS HomeScreen.js U
- ▼ .gitignore
- JS App.js M
- app.json
- index.js
- package-lock.json
- package.json

JS HomeScreen.js U

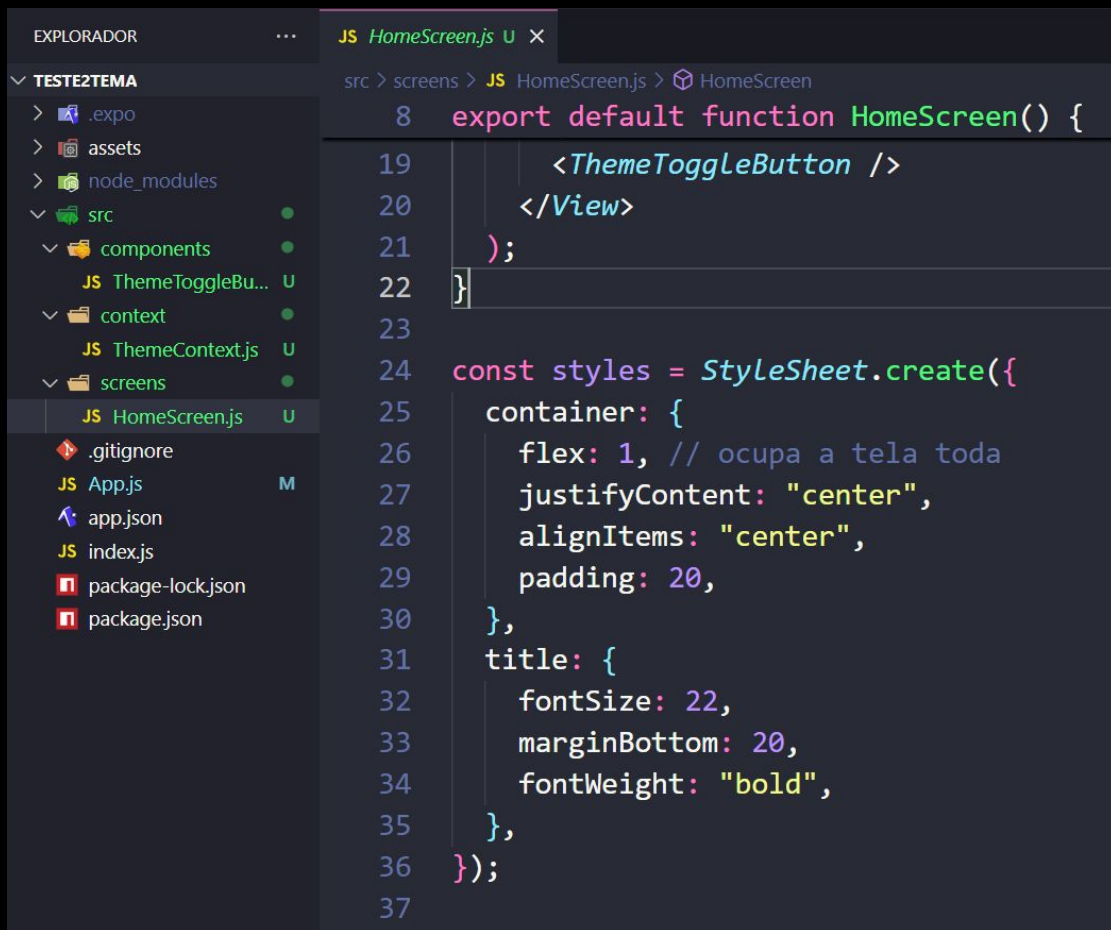
src > screens > JS HomeScreen.js > HomeScreen

```
1 // src/screens/HomeScreen.js
2 // Tela principal que usa o tema
3 import React from "react";
4 import { View, Text, StyleSheet } from "react-native";
5 import { useTheme } from "../context/ThemeContext";
6 import ThemeToggleButton from "../components/ThemeToggleButton";
7
8 export default function HomeScreen() {
9   // Pega o tema atual e as cores do contexto
10   const { colors, theme } = useTheme();
11
12   return (
13     <View style={[styles.container, { backgroundColor: colors.background }]}>
14       <Text style={[styles.title, { color: colors.text }]}>
15         Tema atual: {theme.toUpperCase()}
16       </Text>
17
18       {/* Botão para trocar o tema */}
19       <ThemeToggleButton />
20     </View>
21   );
22 }
```

ESTRUTURA DO CÓDIGO

Tela HomeScreen.js

FIAP



```
EXPLORADOR  ...  JS HomeScreen.js U X
TESTE2TEMA
  > .expo
  > assets
  > node_modules
  > src
    > components
      JS ThemeToggleBu... U
    > context
      JS ThemeContext.js U
    > screens
      JS HomeScreen.js U
  .gitignore
  JS App.js M
  app.json
  index.js
  package-lock.json
  package.json

src > screens > JS HomeScreen.js > HomeScreen
8  export default function HomeScreen() {
19    <ThemeToggleButton />
20    </View>
21  );
22 }
23
24 const styles = StyleSheet.create({
25   container: {
26     flex: 1, // ocupa a tela toda
27     justifyContent: "center",
28     alignItems: "center",
29     padding: 20,
30   },
31   title: {
32     fontSize: 22,
33     marginBottom: 20,
34     fontWeight: "bold",
35   },
36 });
37
```

ThemeToggleButton.js

FLANP

EXPLORADOR

TESTE2TEMA

.expo

assets

node_modules

src

components

JS ThemeToggleButto... U

context

JS ThemeContext.js U

screens

JS HomeScreen.js U

.gitignore

JS App.js M

app.json

index.js

package-lock.json

package.json

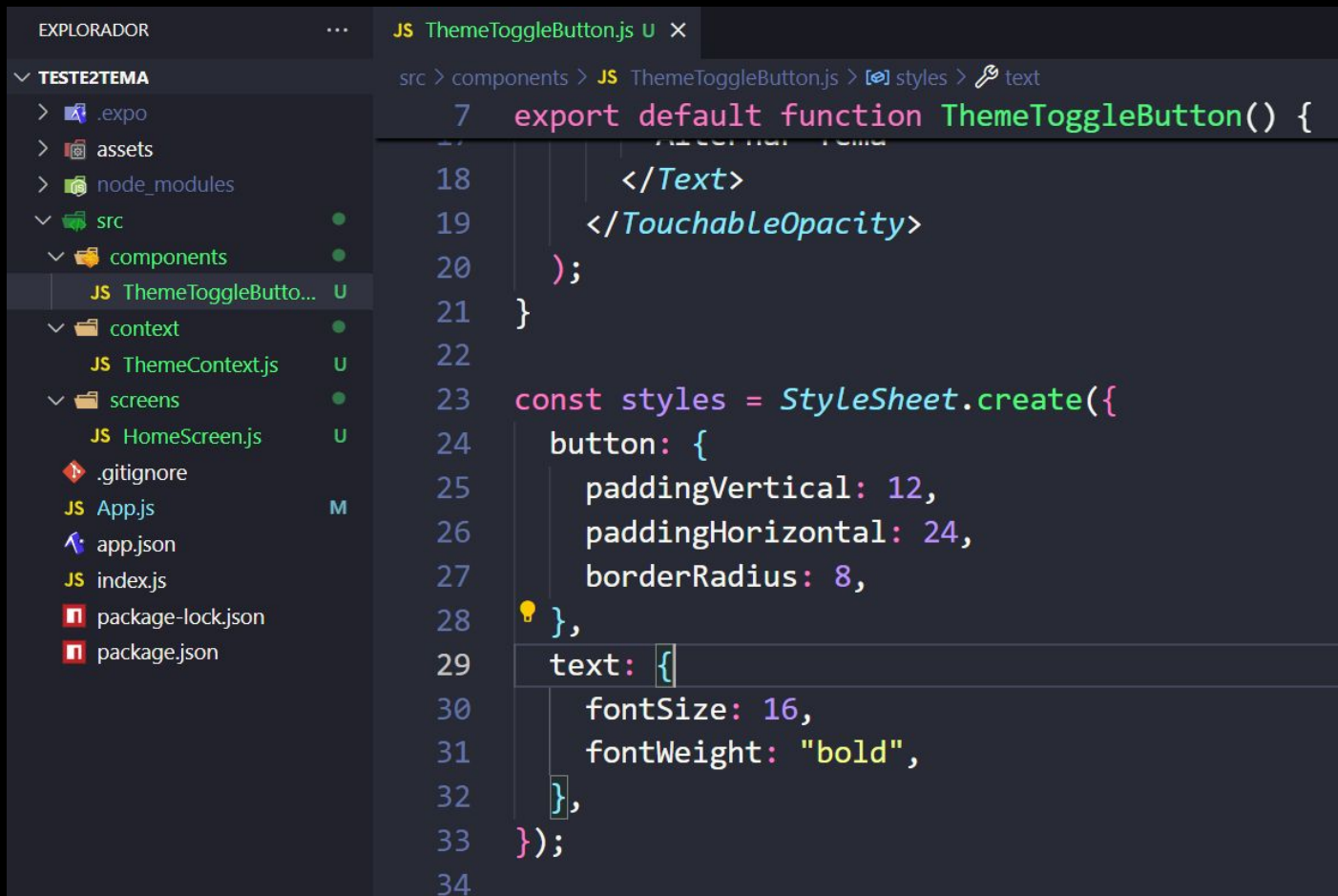
JS ThemeToggleButton.js U

src > components > JS ThemeToggleButton.js > ...

```
1 // src/components/ThemeToggleButton.js
2 // Botão que alterna o tema entre claro e escuro
3 import React from "react";
4 import { TouchableOpacity, Text, StyleSheet } from "react-native";
5 import { useTheme } from "../context/ThemeContext";
6
7 export default function ThemeToggleButton() {
8   // Pega função de alternância e cores do tema
9   const { toggleTheme, colors } = useTheme();
10
11   return (
12     <TouchableOpacity
13       style={[styles.button, { backgroundColor: colors.button }]}
14       onPress={toggleTheme}
15     >
16       <Text style={[styles.text, { color: colors.buttonText }]}>
17         Alternar Tema
18       </Text>
19     </TouchableOpacity>
20   );
21 }
```

ThemeToggleButton.js

FLAP



```
EXPLORADOR  ...  JS ThemeToggleButton.js U X
v TESTE2TEMA
  > .expo
  > assets
  > node_modules
  v src
    v components
      JS ThemeToggleButto... U
    v context
      JS ThemeContext.js U
    v screens
      JS HomeScreen.js U
  .gitignore
  JS App.js M
  app.json
  JS index.js
  package-lock.json
  package.json

src > components > JS ThemeToggleButton.js > styles > text
7  export default function ThemeToggleButton() {
17    </Text>
18    </TouchableOpacity>
19  );
20  }
21
22
23  const styles = StyleSheet.create({
24    button: {
25      paddingVertical: 12,
26      paddingHorizontal: 24,
27      borderRadius: 8,
28    },
29    text: {
30      fontSize: 16,
31      fontWeight: "bold",
32    },
33  });
34
```

Dúvidas?