

Mobile Application Development

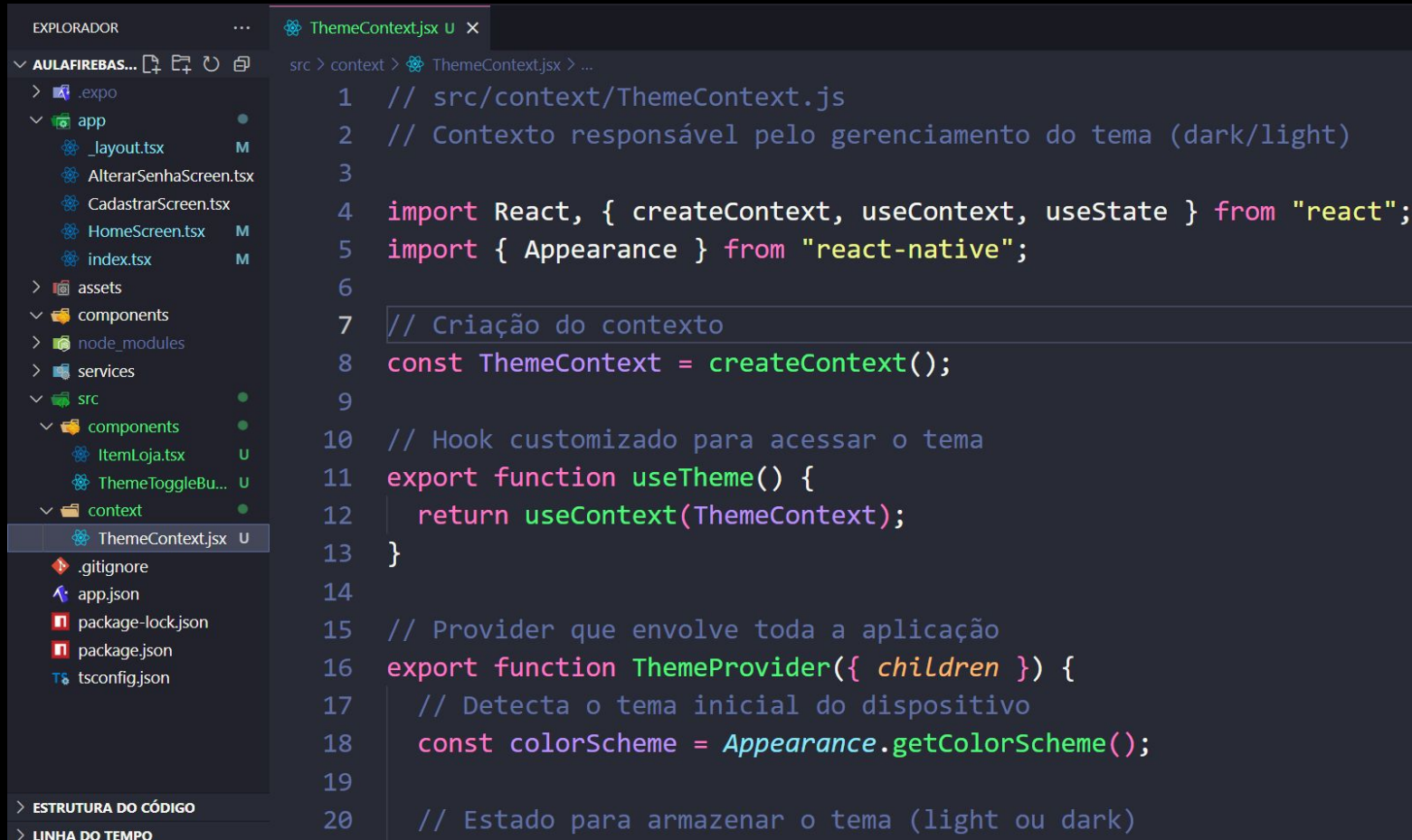
Prof. Fernando Pinéo

Tema Claro e Tema Escuro

Expo Router

Criando o arq ThemeContext.jsx

FILIP



EXPLORADOR

- AULAFIREBAS...
- .expo
- app
 - _layout.tsx
 - AlterarSenhaScreen.tsx
 - CadastrarScreen.tsx
 - HomeScreen.tsx
 - index.tsx
- assets
- components
- node_modules
- services
- src
 - components
 - ItemLoja.tsx
 - ThemeToggleBu...
 - context
 - ThemeContext.jsx
- .gitignore
- app.json
- package-lock.json
- package.json
- tsconfig.json

ESTRUTURA DO CÓDIGO

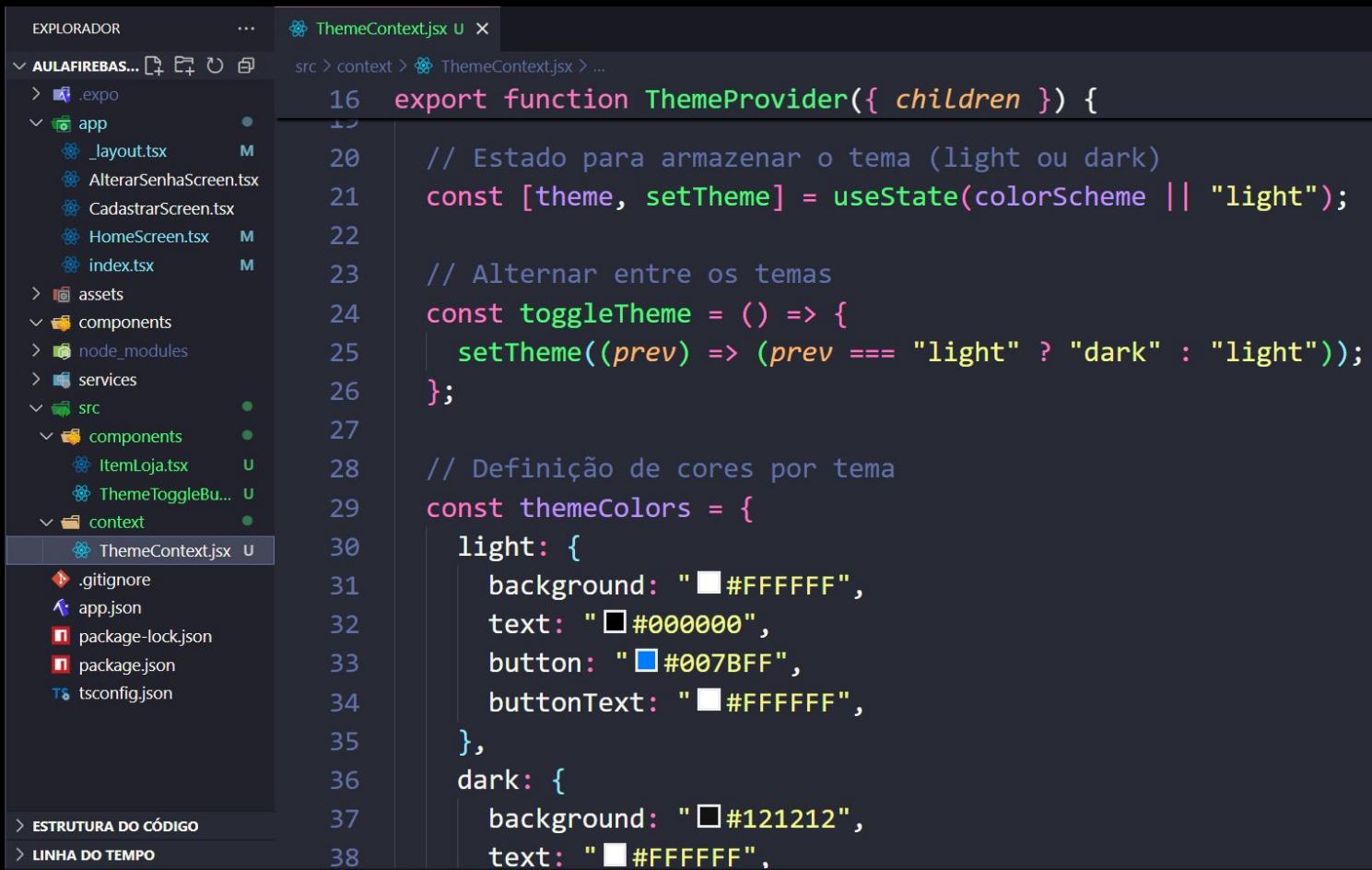
LINHA DO TEMPO

ThemeContext.jsx

```
1 // src/context/ThemeContext.js
2 // Contexto responsável pelo gerenciamento do tema (dark/light)
3
4 import React, { createContext, useContext, useState } from "react";
5 import { Appearance } from "react-native";
6
7 // Criação do contexto
8 const ThemeContext = createContext();
9
10 // Hook customizado para acessar o tema
11 export function useTheme() {
12   return useContext(ThemeContext);
13 }
14
15 // Provider que envolve toda a aplicação
16 export function ThemeProvider({ children }) {
17   // Detecta o tema inicial do dispositivo
18   const colorScheme = Appearance.getColorScheme();
19
20   // Estado para armazenar o tema (light ou dark)
```

Criando o arq ThemeContext.jsx

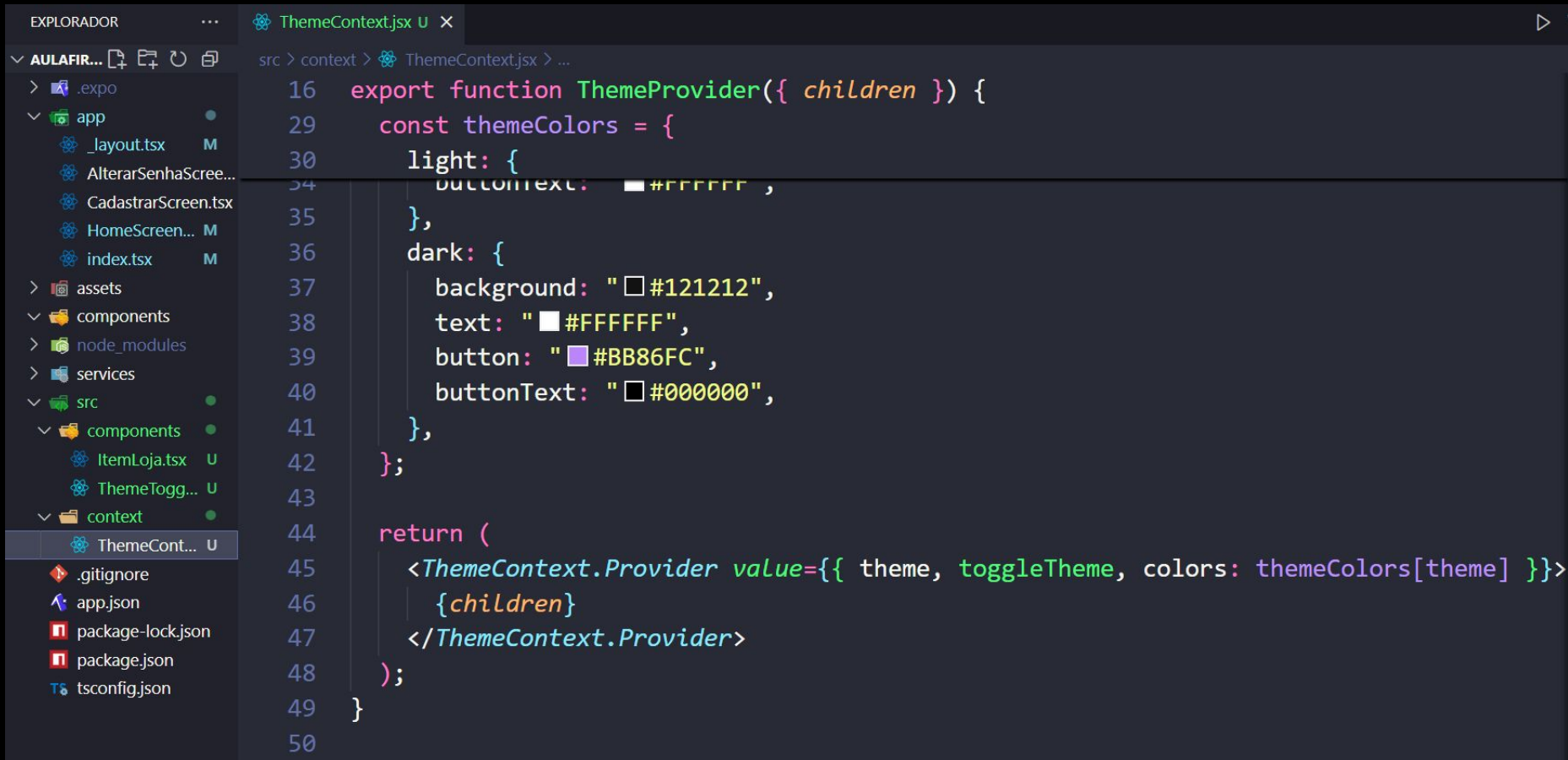
FILIP



```
16 export function ThemeProvider({ children }) {
17
18   // Estado para armazenar o tema (light ou dark)
19   const [theme, setTheme] = useState(colorScheme || "light");
20
21   // Alternar entre os temas
22   const toggleTheme = () => {
23     setTheme((prev) => (prev === "light" ? "dark" : "light"));
24   };
25
26   // Definição de cores por tema
27   const themeColors = {
28     light: {
29       background: "■ #FFFFFF",
30       text: "□ #000000",
31       button: "■ #007BFF",
32       buttonText: "■ #FFFFFF",
33     },
34     dark: {
35       background: "□ #121212",
36       text: "■ #FFFFFF",
37     }
38   };
39 }
```

Criando o arq ThemeContext.jsx

FIAP

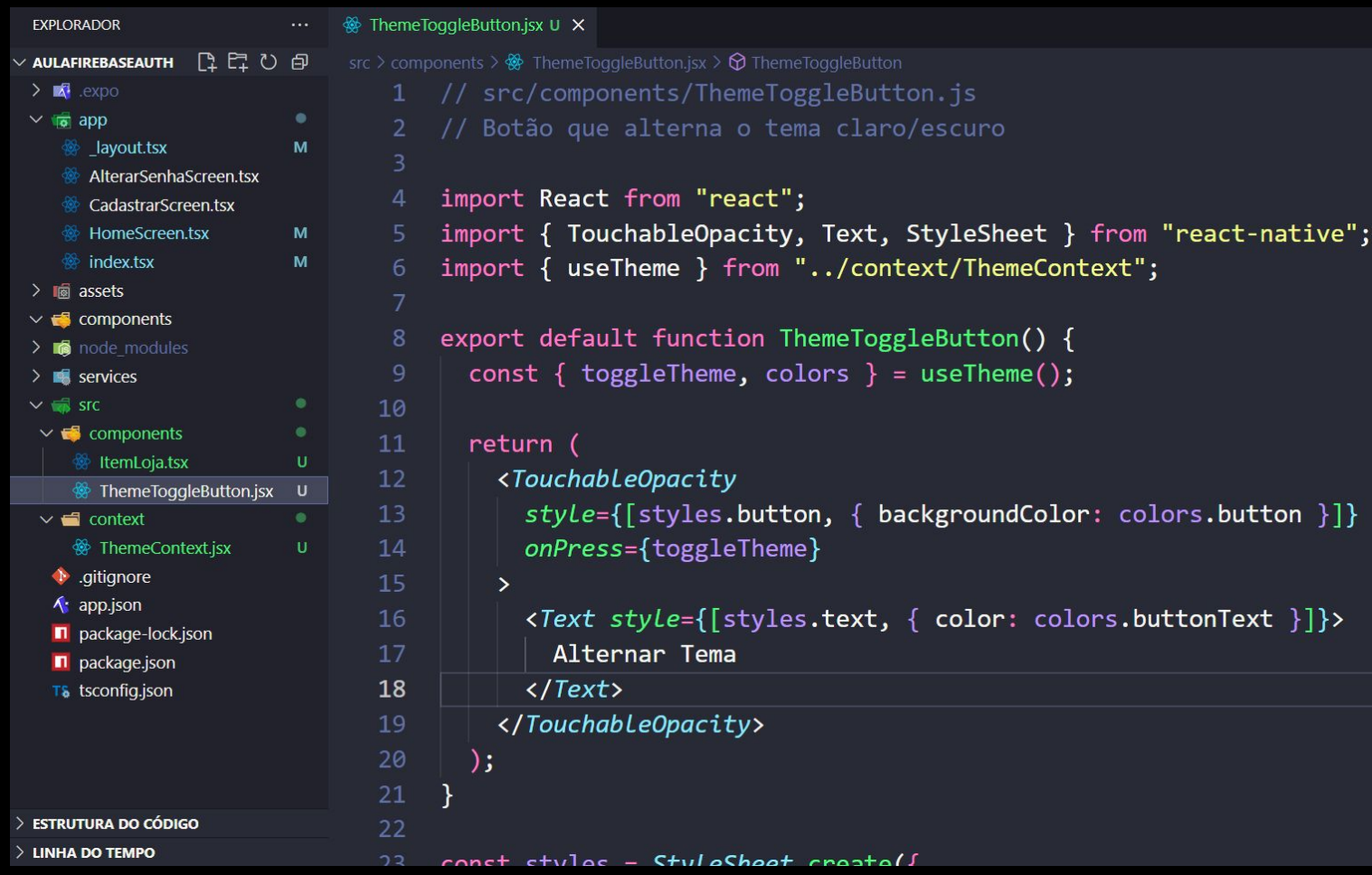


The image shows a VS Code editor window with a dark theme. On the left, the Explorer sidebar shows a project structure with folders like .expo, app, assets, components, node_modules, services, src, and context. The 'context' folder is expanded, showing 'ThemeCont... U'. The main editor area displays the code for 'ThemeContext.jsx' with line numbers 16 to 50. The code defines a 'ThemeProvider' function that exports a theme and a 'toggleTheme' function. It also defines 'themeColors' for 'light' and 'dark' themes. The 'light' theme has a white background, white text, a purple button, and a black button text. The 'dark' theme has a black background, white text, a purple button, and a white button text. The function returns a 'ThemeContext.Provider' with the theme, toggleTheme function, and the selected theme colors.

```
16 export function ThemeProvider({ children }) {
29   const themeColors = {
30     light: {
34       buttonText: "■ #FFFFFF",
35     },
36     dark: {
37       background: "□ #121212",
38       text: "■ #FFFFFF",
39       button: "■ #BB86FC",
40       buttonText: "□ #000000",
41     },
42   };
43
44   return (
45     <ThemeContext.Provider value={{ theme, toggleTheme, colors: themeColors[theme] }}>
46       {children}
47     </ThemeContext.Provider>
48   );
49 }
50
```

Criando o arq ThemeToggleButton.jsx

FILAP

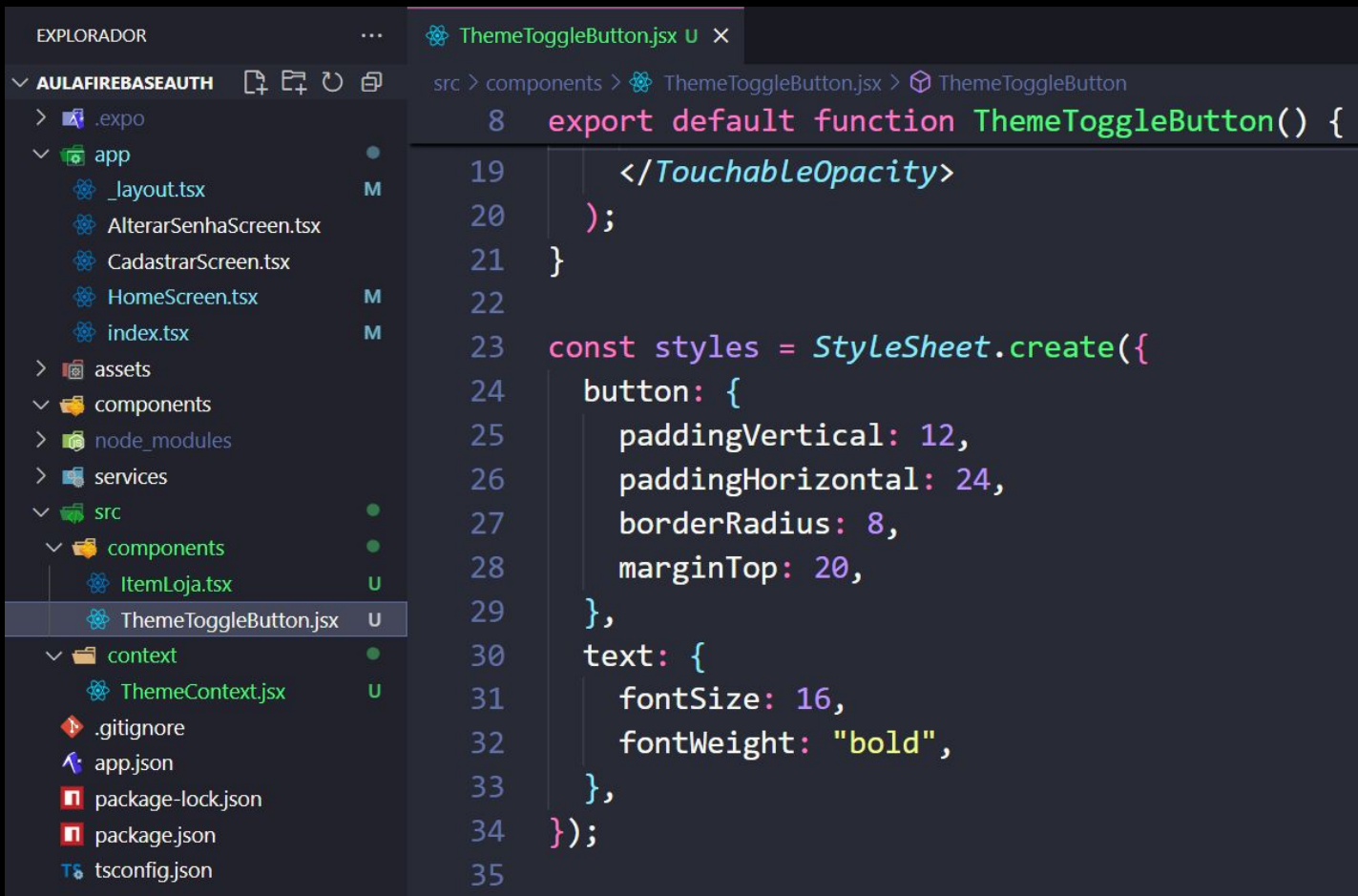


The image shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer is titled "EXPLORADOR" and shows a project structure for "AULAFIREBASEAUTH". The code editor is titled "ThemeToggleButton.jsx" and shows the following code:

```
1 // src/components/ThemeToggleButton.js
2 // Botão que alterna o tema claro/escuro
3
4 import React from "react";
5 import { TouchableOpacity, Text, StyleSheet } from "react-native";
6 import { useTheme } from "../context/ThemeContext";
7
8 export default function ThemeToggleButton() {
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 }
22
23 const styles = StyleSheet.create({
```


Criando o arq ThemeToggleButton.jsx

FILAP

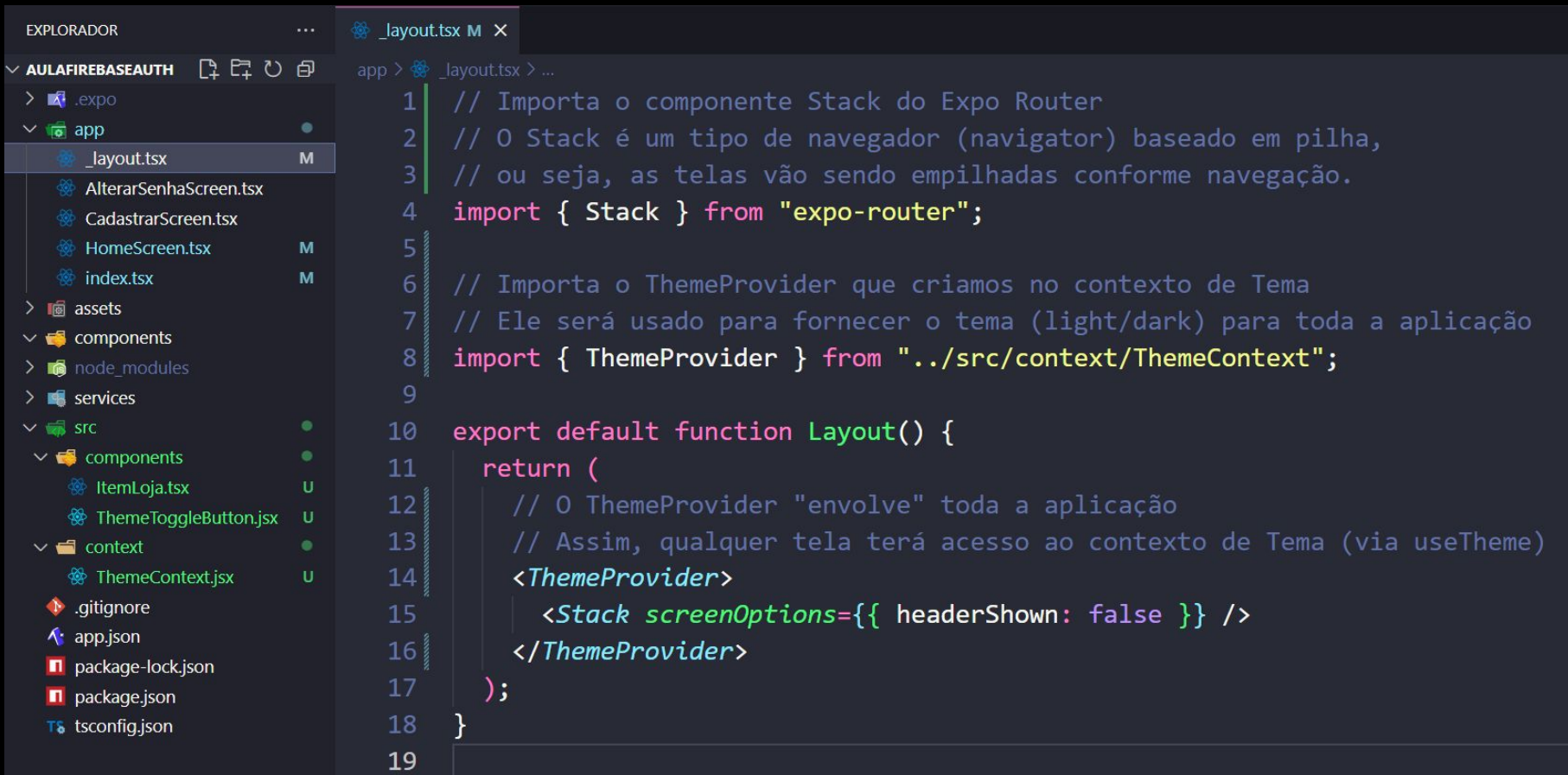


The image shows a code editor interface. On the left is the 'EXPLORADOR' (Explorer) panel showing a file tree. The 'src' directory is expanded, showing 'components' and 'context'. 'ThemeToggleButton.jsx' is selected in 'components'. On the right is the editor window for 'ThemeToggleButton.jsx', showing the following code:

```
8  export default function ThemeToggleButton() {  
19    </TouchableOpacity>  
20  };  
21 }  
22  
23 const styles = StyleSheet.create({  
24   button: {  
25     paddingVertical: 12,  
26     paddingHorizontal: 24,  
27     borderRadius: 8,  
28     marginTop: 20,  
29   },  
30   text: {  
31     fontSize: 16,  
32     fontWeight: "bold",  
33   },  
34 });  
35
```

Modificação no arq _layout.tsx

FIAP

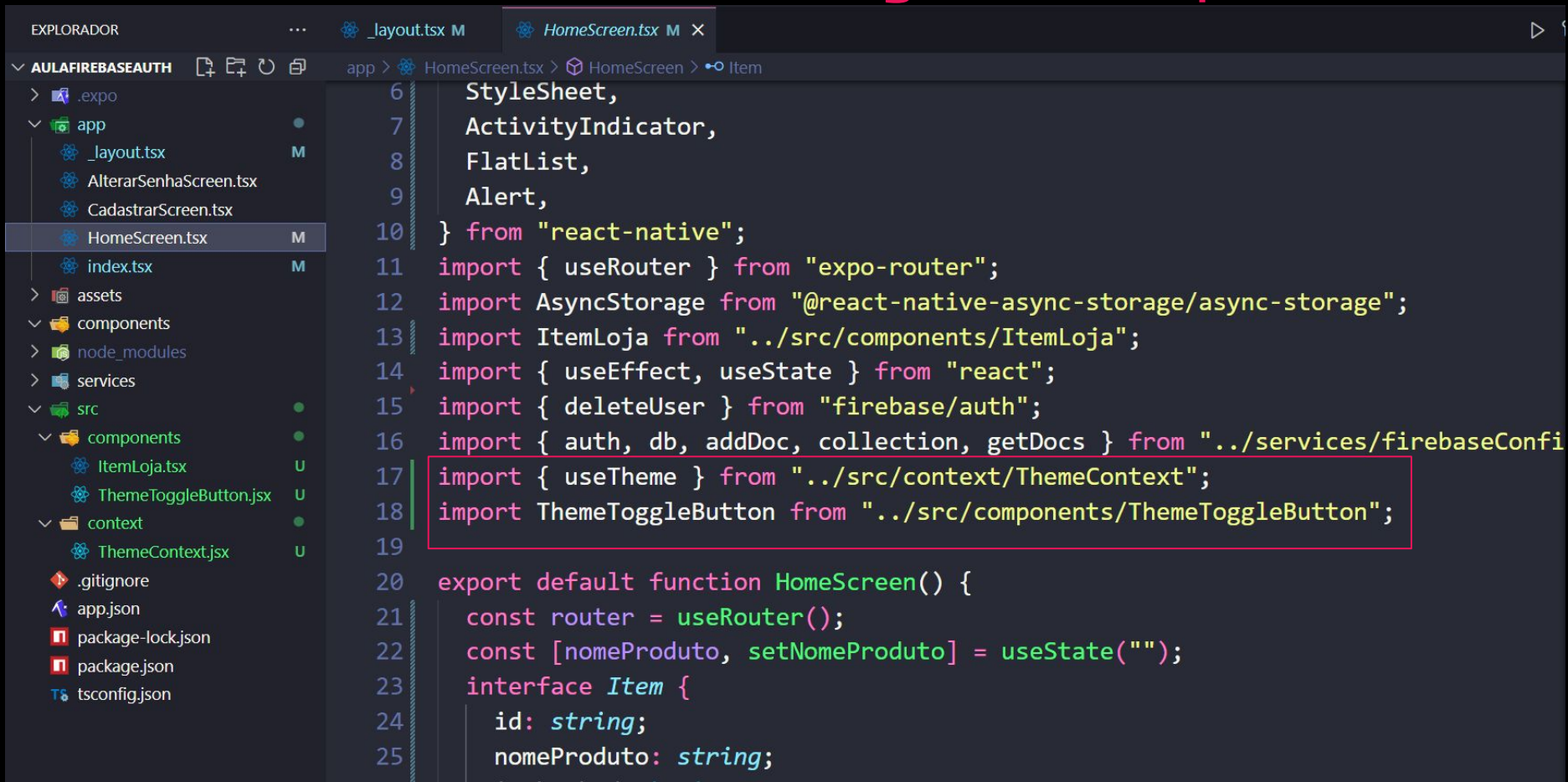


The image shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer, titled 'EXPLORADOR', shows a project structure for 'AULAFIREBASEAUTH'. The 'app' directory is expanded, showing files like '_layout.tsx' (marked with 'M'), 'AlterarSenhaScreen.tsx', 'CadastrarScreen.tsx', 'HomeScreen.tsx' (marked with 'M'), and 'index.tsx' (marked with 'M'). Other directories like 'assets', 'components', 'node_modules', 'services', and 'src' are also visible. The 'src' directory is expanded, showing 'components' (with 'ItemLoja.tsx' marked 'U' and 'ThemeToggleButton.jsx' marked 'U'), 'context' (with 'ThemeContext.jsx' marked 'U'), and other files like '.gitignore', 'app.json', 'package-lock.json', 'package.json', and 'tsconfig.json'.

The code editor on the right shows the content of '_layout.tsx'. The code is as follows:

```
1 // Importa o componente Stack do Expo Router
2 // O Stack é um tipo de navegador (navigator) baseado em pilha,
3 // ou seja, as telas vão sendo empilhadas conforme navegação.
4 import { Stack } from "expo-router";
5
6 // Importa o ThemeProvider que criamos no contexto de Tema
7 // Ele será usado para fornecer o tema (light/dark) para toda a aplicação
8 import { ThemeProvider } from "../src/context/ThemeContext";
9
10 export default function Layout() {
11   return (
12     // O ThemeProvider "envolve" toda a aplicação
13     // Assim, qualquer tela terá acesso ao contexto de Tema (via useTheme)
14     <ThemeProvider>
15       <Stack screenOptions={{ headerShown: false }} />
16     </ThemeProvider>
17   );
18 }
19
```


Na HomeScreen, realizar os seguintes imports



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like .expo, app, assets, components, node_modules, services, src, and context. The file HomeScreen.tsx is selected in the app folder. The code editor shows the content of HomeScreen.tsx, with imports highlighted in a red box. The imports are:

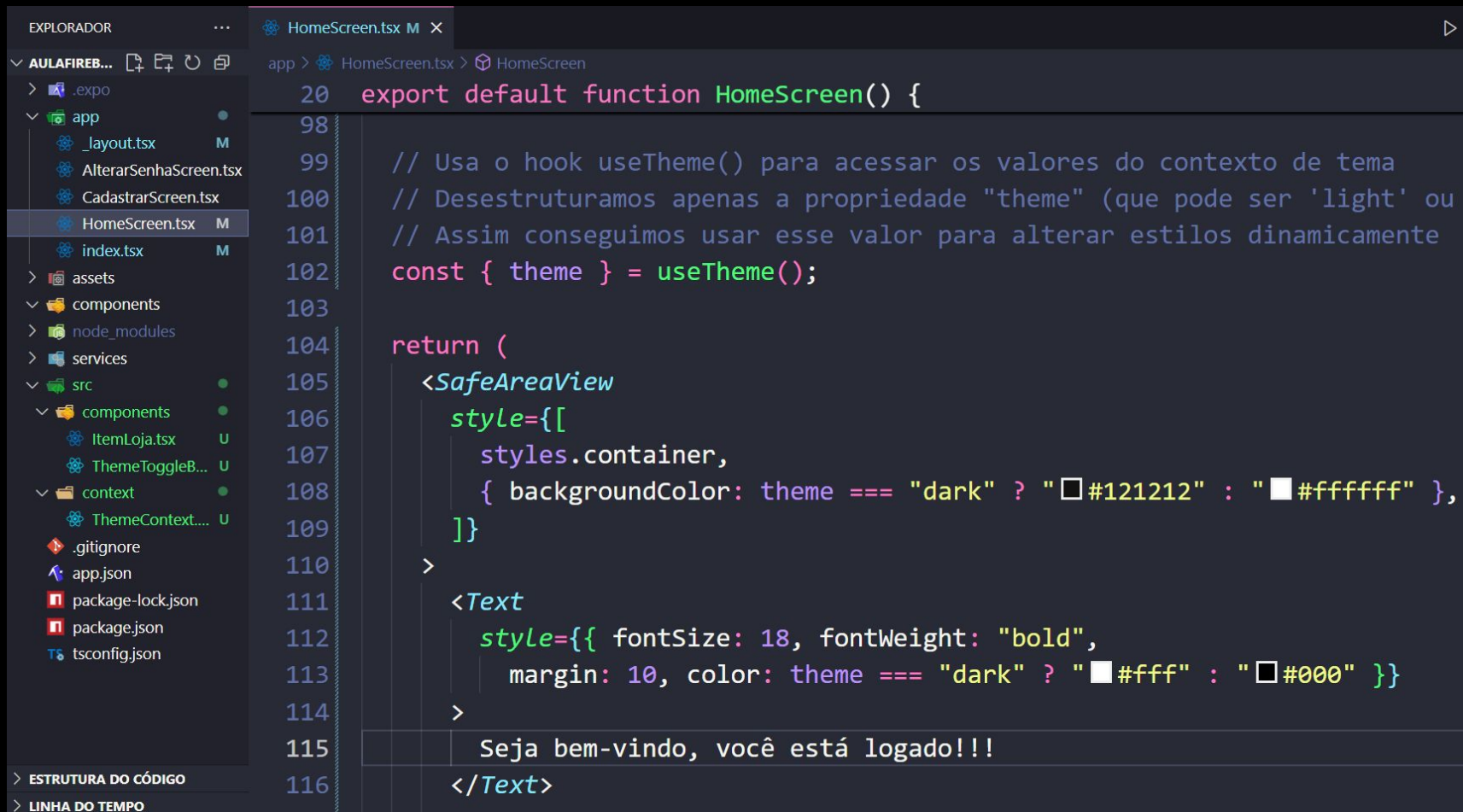
```
6 StyleSheet,  
7 ActivityIndicator,  
8 FlatList,  
9 Alert,  
10 } from "react-native";  
11 import { useRouter } from "expo-router";  
12 import AsyncStorage from "@react-native-async-storage/async-storage";  
13 import ItemLoja from "../src/components/ItemLoja";  
14 import { useEffect, useState } from "react";  
15 import { deleteUser } from "firebase/auth";  
16 import { auth, db, addDoc, collection, getDocs } from "../services/firebaseConfig";  
17 import { useTheme } from "../src/context/ThemeContext";  
18 import ThemeToggleButton from "../src/components/ThemeToggleButton";
```

The code continues with the HomeScreen function definition:

```
20 export default function HomeScreen() {  
21   const router = useRouter();  
22   const [nomeProduto, setNomeProduto] = useState("");  
23   interface Item {  
24     id: string;  
25     nomeProduto: string;
```

Alteração na Tela HomeScreen

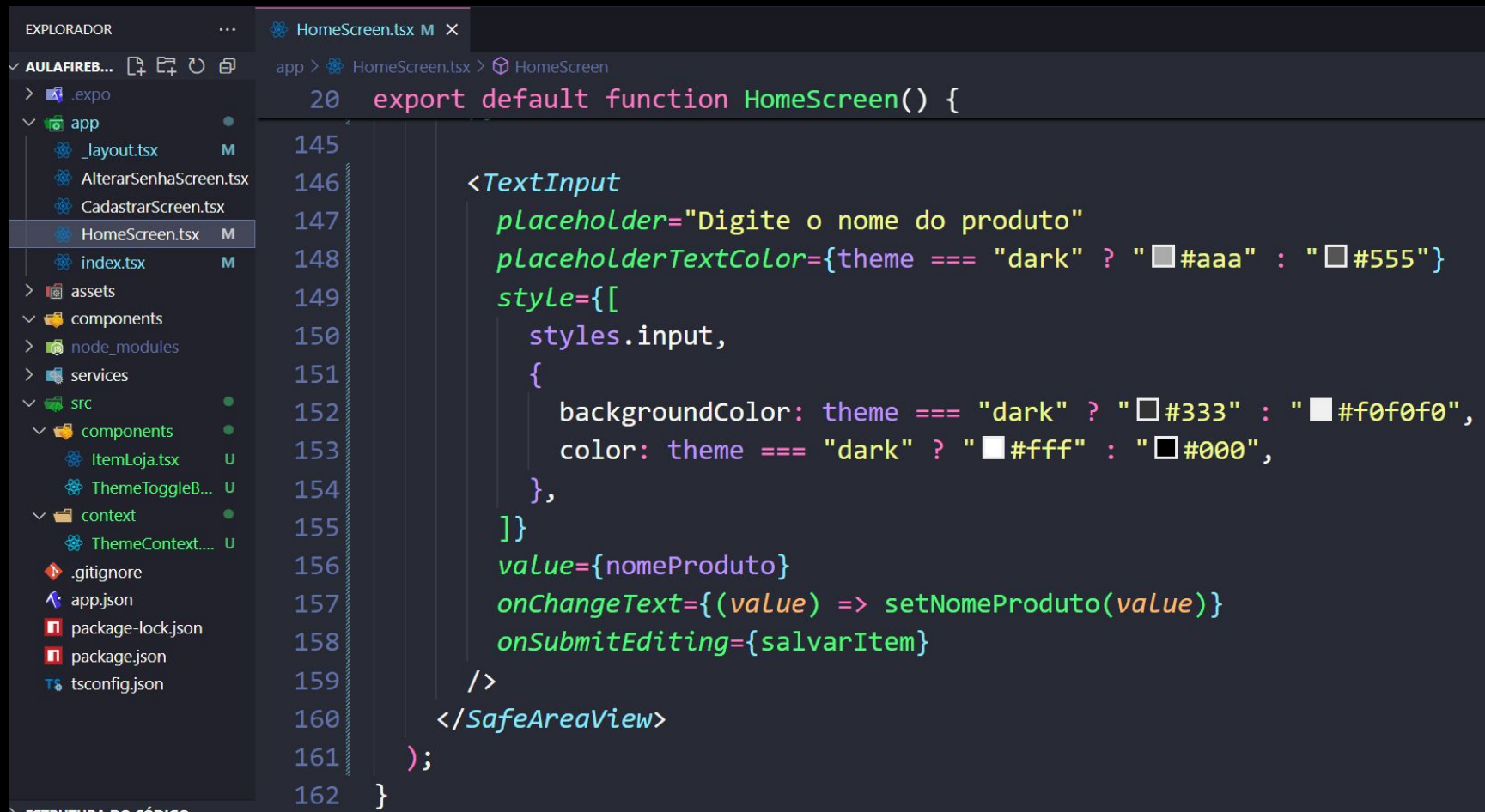
FIAP



```
20 export default function HomeScreen() {
98
99   // Usa o hook useTheme() para acessar os valores do contexto de tema
100   // Desestruturamos apenas a propriedade "theme" (que pode ser 'light' ou
101   // Assim conseguimos usar esse valor para alterar estilos dinamicamente
102   const { theme } = useTheme();
103
104   return (
105     <SafeAreaView
106       style={[
107         styles.container,
108         { backgroundColor: theme === "dark" ? "■#121212" : "■#ffffff" },
109       ]
110     >
111       <Text
112         style={{ fontSize: 18, fontWeight: "bold",
113           margin: 10, color: theme === "dark" ? "■#fff" : "■#000" }}
114       >
115         Seja bem-vindo, você está logado!!!
116       </Text>
```

Alteração na Tela HomeScreen

FIAP

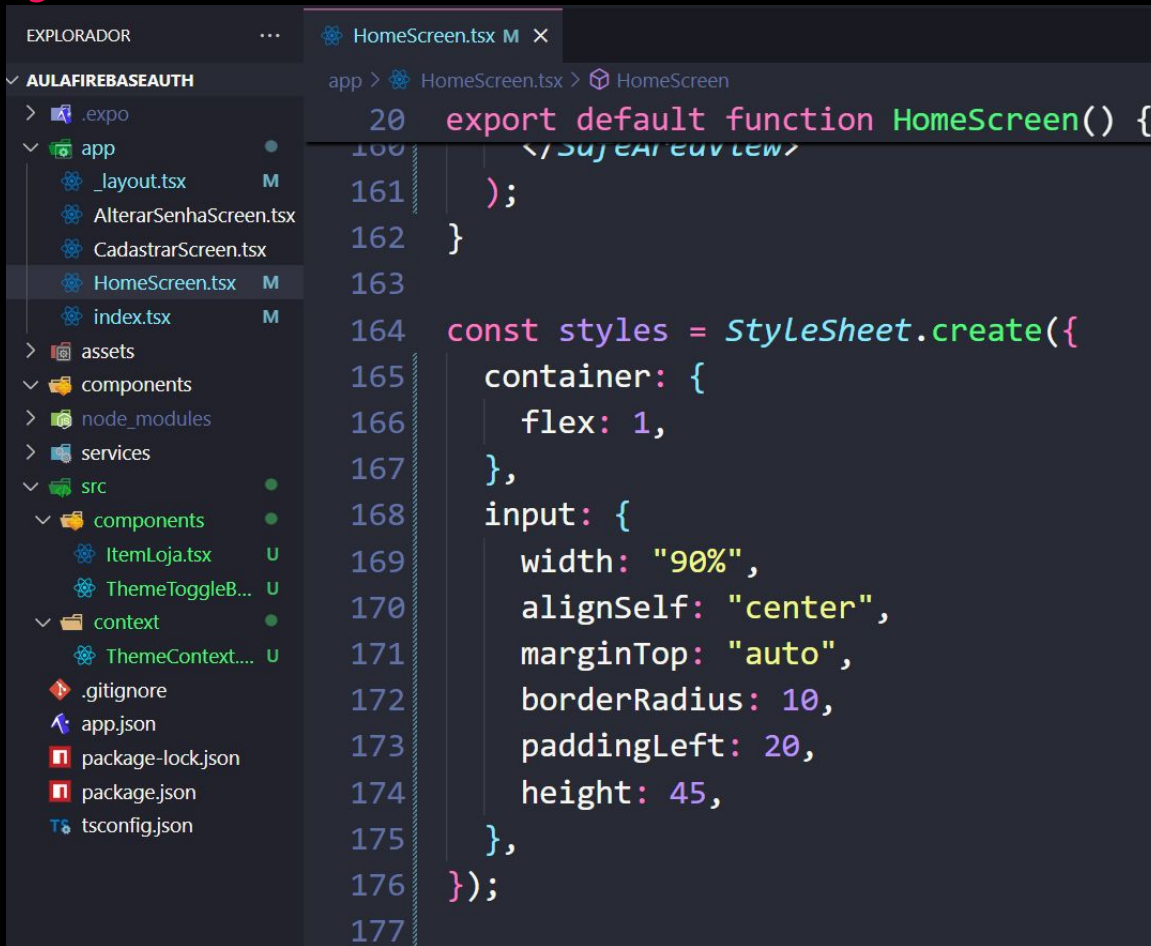


```
EXPLORADOR  ...  HomeScreen.tsx M X
AULAFIREB...  app > HomeScreen.tsx > HomeScreen
  > .expo
  > app
    _layout.tsx M
    AlterarSenhaScreen.tsx
    CadastrarScreen.tsx
    HomeScreen.tsx M
    index.tsx M
  > assets
  > components
  > node_modules
  > services
  > src
    components
      ItemLoja.tsx U
      ThemeToggleB... U
    context
      ThemeContext.... U
  > .gitignore
  > app.json
  > package-lock.json
  > package.json
  > tsconfig.json

20  export default function HomeScreen() {
145
146    <TextInput
147      placeholder="Digite o nome do produto"
148      placeholderTextColor={theme === "dark" ? "■ #aaa" : "■ #555"}
149      style={[
150        styles.input,
151        {
152          backgroundColor: theme === "dark" ? "■ #333" : "■ #f0f0f0",
153          color: theme === "dark" ? "■ #fff" : "■ #000",
154        },
155      ]}
156      value={nomeProduto}
157      onChangeText={(value) => setNomeProduto(value)}
158      onSubmitEditing={salvarItem}
159    />
160    </SafeAreaView>
161  );
162 }
```

Alteração na Tela HomeScreen

FIAP



```
EXPLORADOR    ...    HomeScreen.tsx M X
AULAFIREBASEAUTH
  > .expo
  > app
    _layout.tsx M
    AlterarSenhaScreen.tsx
    CadastrarScreen.tsx
    HomeScreen.tsx M
    index.tsx M
  > assets
  > components
  > node_modules
  > services
  > src
    components
      ItemLoja.tsx U
      ThemeToggleB... U
    context
      ThemeContext.... U
  .gitignore
  app.json
  package-lock.json
  package.json
  tsconfig.json

app > HomeScreen.tsx > HomeScreen
20  export default function HomeScreen() {
161  };
162  }
163
164  const styles = StyleSheet.create({
165    container: {
166      flex: 1,
167    },
168    input: {
169      width: "90%",
170      alignSelf: "center",
171      marginTop: "auto",
172      borderRadius: 10,
173      paddingLeft: 20,
174      height: 45,
175    },
176  });
177
```

Alterando o esquema de cores para a tela de
Login(index)

Na tela Index

FIAP

EXPLORADOR

index.tsx M X

app > index.tsx > LoginScreen

```
1 import { Link } from 'expo-router';
2 import React, { useState, useEffect } from 'react';
3 import { View, Text, TextInput, TouchableOpacity, StyleSheet, Alert } from '
4 import AsyncStorage from '@react-native-async-storage/async-storage';
5 import { useRouter } from 'expo-router';
6 import { signInWithEmailAndPassword, sendPasswordResetEmail } from 'firebase/
7 import { auth } from '../services/firebaseConfig'
8 import { useTheme } from "../src/context/ThemeContext";
9
10
11
12 export default function LoginScreen() {
13   // Usa o hook useTheme() para acessar os valores do contexto de tema
14   // Desestruturamos apenas a propriedade "theme" (que pode ser 'light' ou '
15   // Assim conseguimos usar esse valor para alterar estilos dinamicamente
16   const { theme } = useTheme();
17
18   // Estados para armazenar os valores digitados
19
```


Na tela Index

FIAP

```
index.tsx M ×
app > index.tsx > LoginScreen
12 export default function LoginScreen() {
80
81   return (
82     <View style={styles.container,
83       { backgroundColor: theme === 'dark' ? '■ #121212' : '■ #ffffff' }]}>
84       <Text style={styles.titulo,
85         { color: theme === 'dark' ? '■ #fff' : '■ #000' }]}>Realizar login</Text>
86
87
88       { /* Campo Email */ }
89       <TextInput
90         style={styles.input,
91           { backgroundColor: theme === 'dark' ? '■ #1E1E1E' : '■ #f0f0f0',
92             color: theme === 'dark' ? '■ #fff' : '■ #000' }]}
93         placeholder="E-mail"
94         placeholderTextColor={theme === 'dark' ? '■ #aaa' : '■ #555'}
95         keyboardType="email-address"
```

Na tela Index

FIAP

index.tsx M x

app > index.tsx > LoginScreen

```
12 export default function LoginScreen() {  
13     // ...  
96     autoCapitalize="none"  
97     value={email}  
98     onChangeText={setEmail}  
99     />  
100  
101     {/* Campo Senha */}  
102     <TextInput  
103         style={[styles.input,  
104             { backgroundColor: theme === 'dark' ? '■ #1E1E1E' : '■ #f0f0f0',  
105               color: theme === 'dark' ? '■ #fff' : '■ #000' }]]  
106         placeholder="Senha"  
107         placeholderTextColor={theme === 'dark' ? '■ #aaa' : '■ #555'}  
108         secureTextEntry  
109         value={senha}  
110         onChangeText={setSenha}  
111     />
```

Na tela Index

```
index.tsx M x
app > index.tsx > LoginScreen
12 export default function LoginScreen() {
111 //
112
113   {/* Botão */}
114   <TouchableOpacity style={styles.botao} onPress={handleLogin}>
115     <Text style={styles.textoBotao}>Login</Text>
116   </TouchableOpacity>
117
118   <Link href="CadastrarScreen" style={{ marginTop: 20,
119     color: theme === 'dark' ? '#00B37E' : '#00875F',
120     marginLeft: 150 }}>Cadastre-se</Link>
121
122   <Text style={{ marginTop: 20,
123     color: theme === 'dark' ? '#00B37E' : '#00875F',
124     marginLeft: 130 }} onPress={esqueceuSenha}>Esquece a senha</Text>
125 </View>
126 );
127 }
```

Na tela Index

```
index.tsx M X
app > index.tsx > LoginScreen
128
129 // Estilização
130 const styles = StyleSheet.create({
131   container: {
132     flex: 1,
133     justifyContent: 'center',
134     padding: 20,
135   },
136   titulo: {
137     fontSize: 28,
138     fontWeight: 'bold',
139     color: '■ #fff',
140     marginBottom: 30,
141     textAlign: 'center',
142   },
```

Na tela Index

FIAP

```
index.tsx M x
app > index.tsx > LoginScreen
130 const styles = StyleSheet.create({
142   },
143   input: {
144     backgroundColor: '□ #1E1E1E',
145     color: '■ #fff',
146     borderRadius: 10,
147     padding: 15,
148     marginBottom: 15,
149     fontSize: 16,
150     borderWidth: 1,
151     borderColor: '□ #333',
152   },
153   botao: {
154     backgroundColor: '■ #00B37E',
155     padding: 15,
156     borderRadius: 10,
157     alignItems: 'center',
158   },
159 }
```

Na tela Index

FIAP

index.tsx M X

app > index.tsx > LoginScreen

```
130 const styles = StyleSheet.create({  
153   botao: {  
154     backgroundColor: '■ #00B37E',  
155     padding: 15,  
156     borderRadius: 10,  
157     alignItems: 'center',  
158   },  
159   textoBotao: {  
160     color: '■ #fff',  
161     fontSize: 18,  
162     fontWeight: 'bold',  
163   },  
164 });  
165
```


Dúvidas?