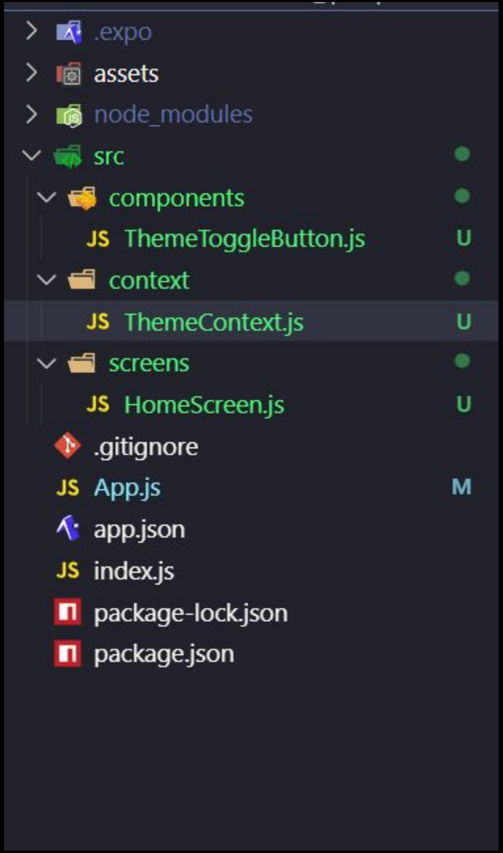
FIAP

Mobile Application Development

Prof. Fernando Pinéo

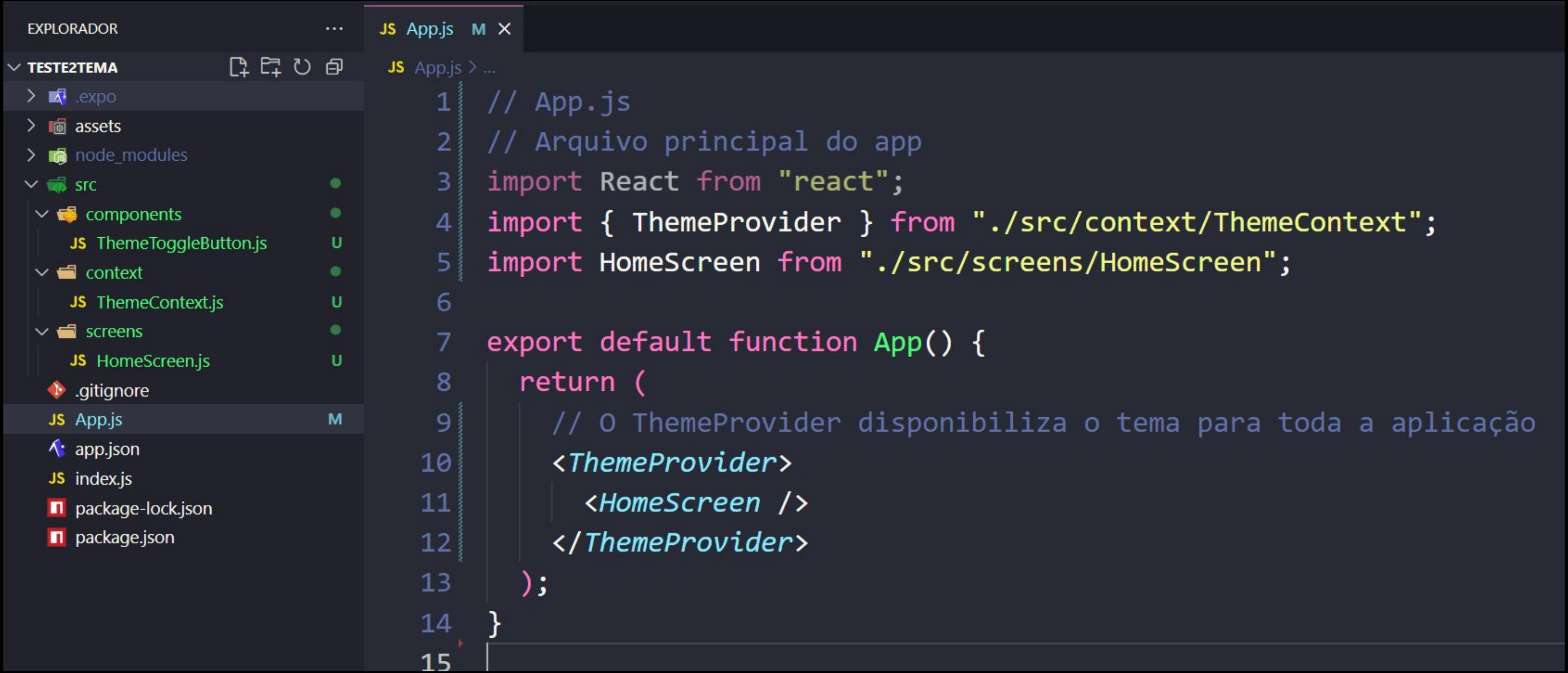
Tema Claro e Tema Escuro

Estrutura do projeto



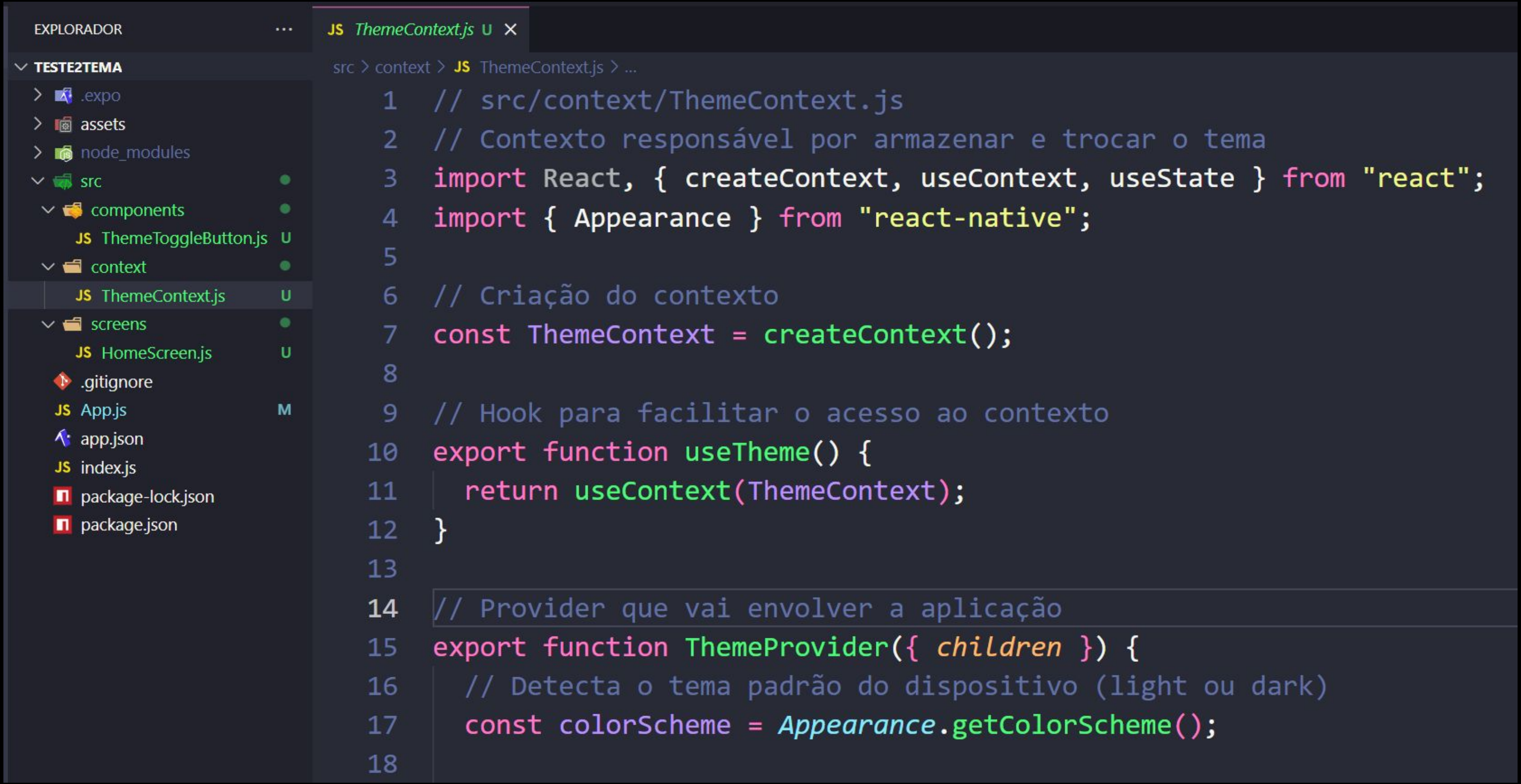
```
> .expo
> assets
> node_modules
v src                          ●
  v components                 ●
      ThemeToggleButton.js     U
  v context                    ●
      ThemeContext.js          U
  v screens                    ●
      HomeScreen.js            U
  .gitignore
  App.js            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

No arq App.js



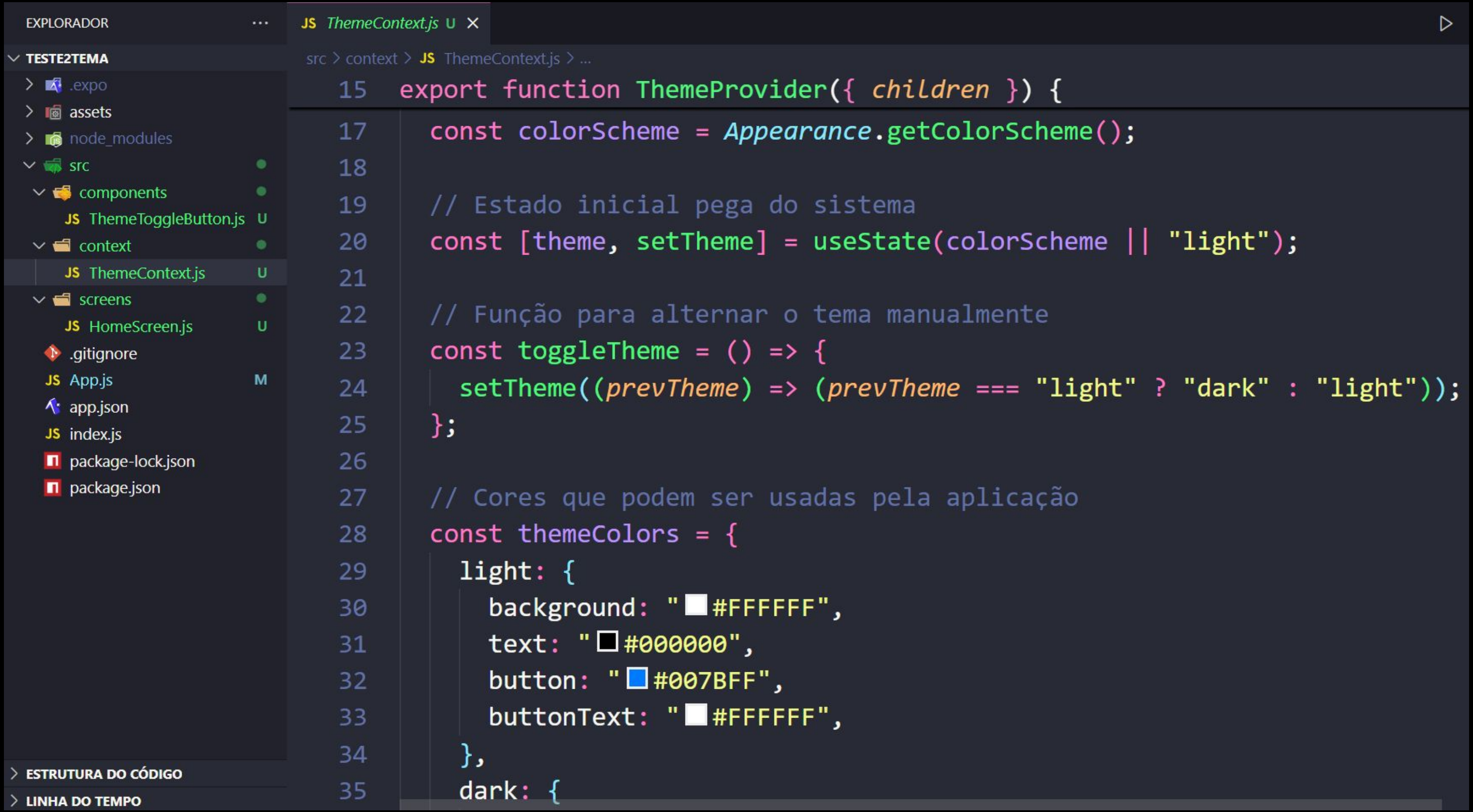
```javascript
// App.js
// Arquivo principal do app
import React from "react";
import { ThemeProvider } from "./src/context/ThemeContext";
import HomeScreen from "./src/screens/HomeScreen";

export default function App() {
  return (
    // O ThemeProvider disponibiliza o tema para toda a aplicação
    <ThemeProvider>
      <HomeScreen />
    </ThemeProvider>
  );
}

```

No arq ThemeContext



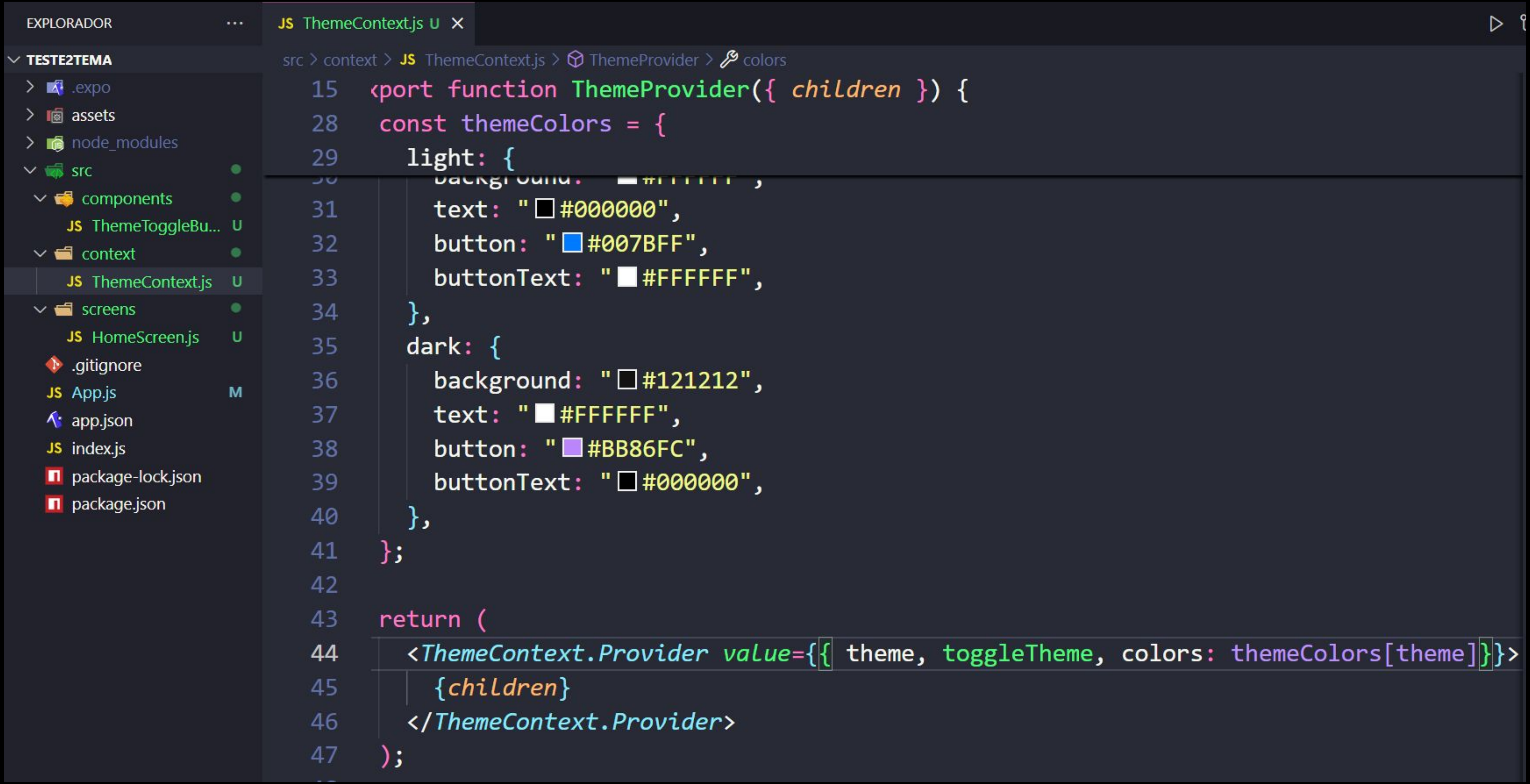
```javascript
// src/context/ThemeContext.js
// Contexto responsável por armazenar e trocar o tema
import React, { createContext, useContext, useState } from "react";
import { Appearance } from "react-native";

// Criação do contexto
const ThemeContext = createContext();

// Hook para facilitar o acesso ao contexto
export function useTheme() {
  return useContext(ThemeContext);
}

// Provider que vai envolver a aplicação
export function ThemeProvider({ children }) {
  // Detecta o tema padrão do dispositivo (light ou dark)
  const colorScheme = Appearance.getColorScheme();

```

No arq ThemeContext



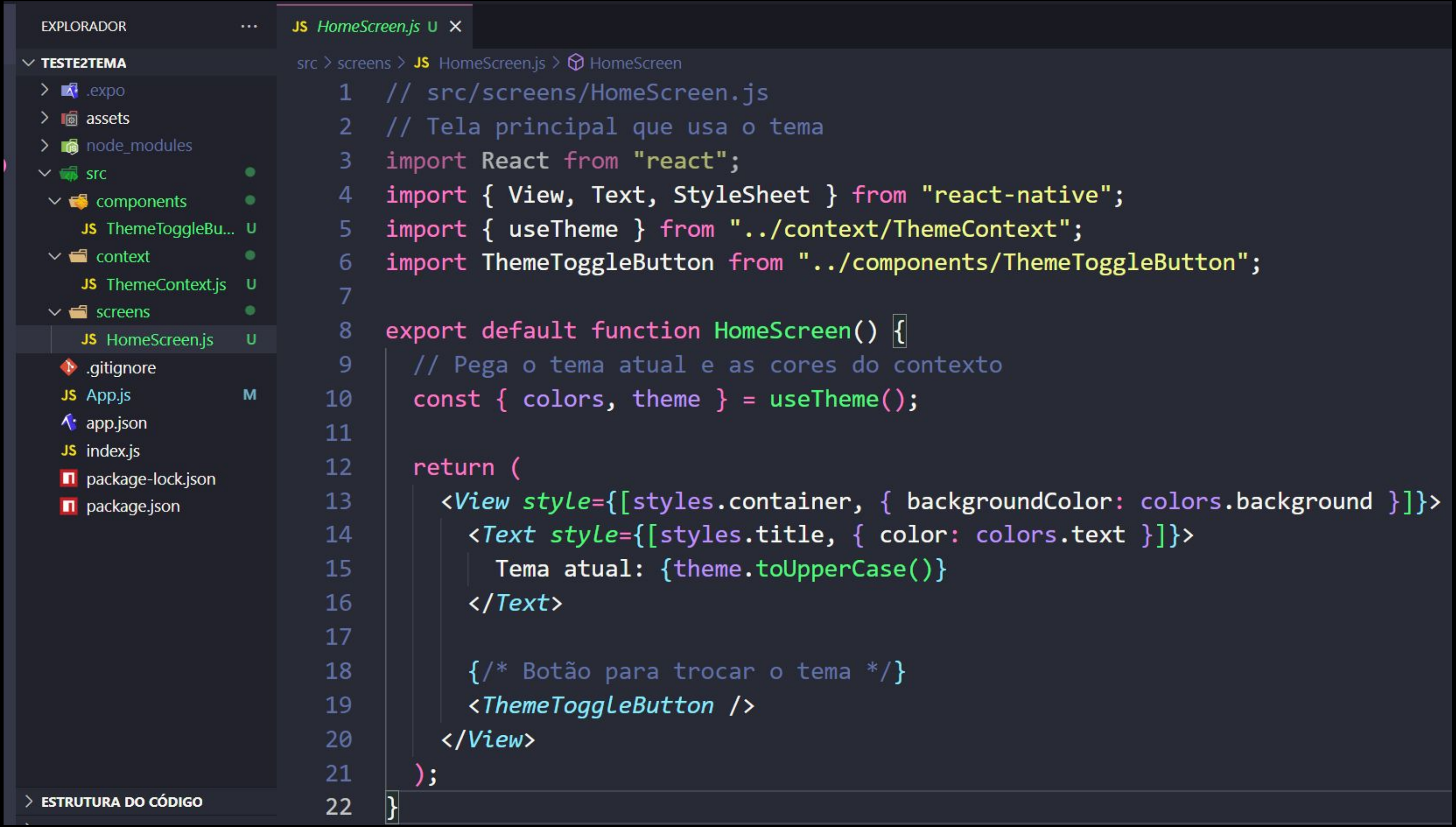
```javascript
export function ThemeProvider({ children }) {
  const colorScheme = Appearance.getColorScheme();

  // Estado inicial pega do sistema
  const [theme, setTheme] = useState(colorScheme || "light");

  // Função para alternar o tema manualmente
  const toggleTheme = () => {
    setTheme((prevTheme) => (prevTheme === "light" ? "dark" : "light"));
  };

  // Cores que podem ser usadas pela aplicação
  const themeColors = {
    light: {
      background: "#FFFFFF",
      text: "#000000",
      button: "#007BFF",
      buttonText: "#FFFFFF",
    },
    dark: {
```

No arq ThemeContext



```
‹port function ThemeProvider({ children }) {
 const themeColors = {
   light: {
     background: "#FFFFFF",
     text: "#000000",
     button: "#007BFF",
     buttonText: "#FFFFFF",
   },
   dark: {
     background: "#121212",
     text: "#FFFFFF",
     button: "#BB86FC",
     buttonText: "#000000",
   },
 };

 return (
   <ThemeContext.Provider value={{ theme, toggleTheme, colors: themeColors[theme]}}>
     {children}
   </ThemeContext.Provider>
 );
```

Tela HomeScreen.js



EXPLORADOR

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

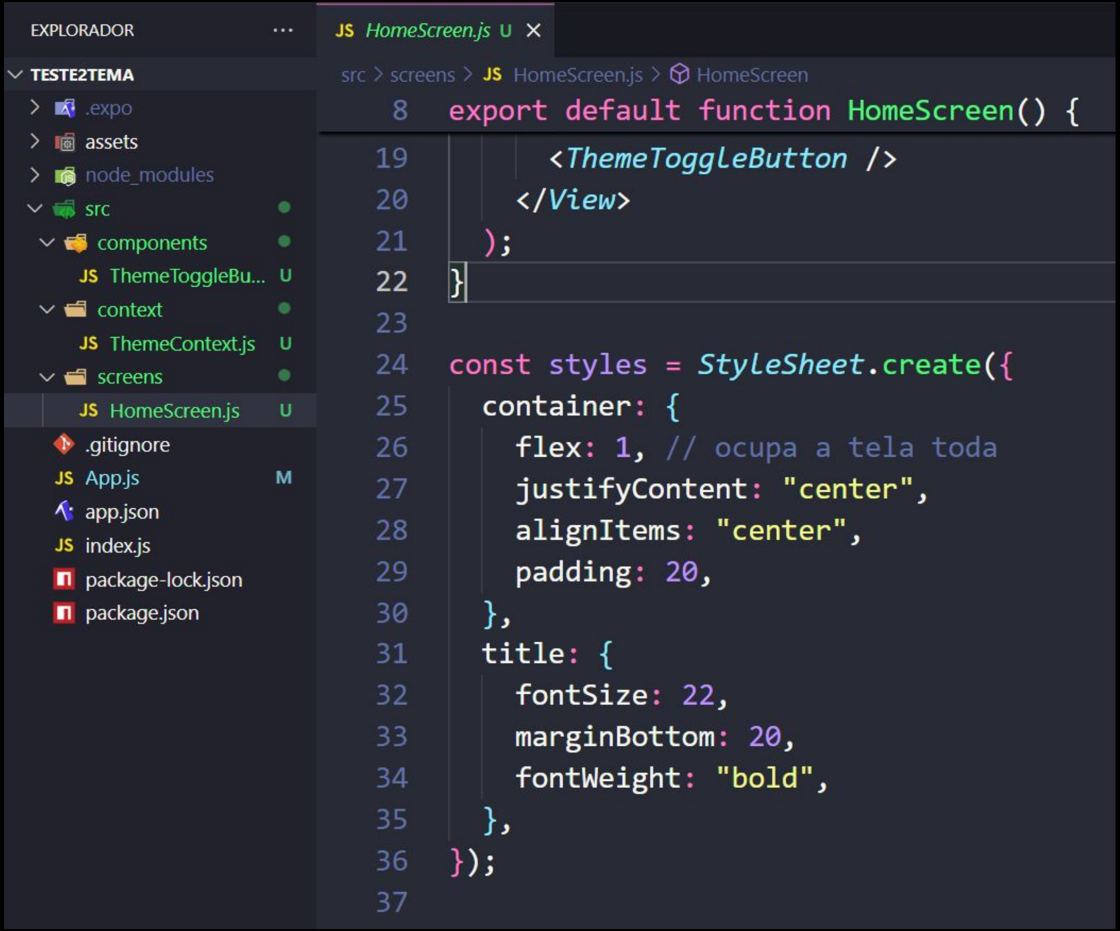
ESTRUTURA DO CÓDIGO

HomeScreen.js U

src > screens > HomeScreen.js > HomeScreen

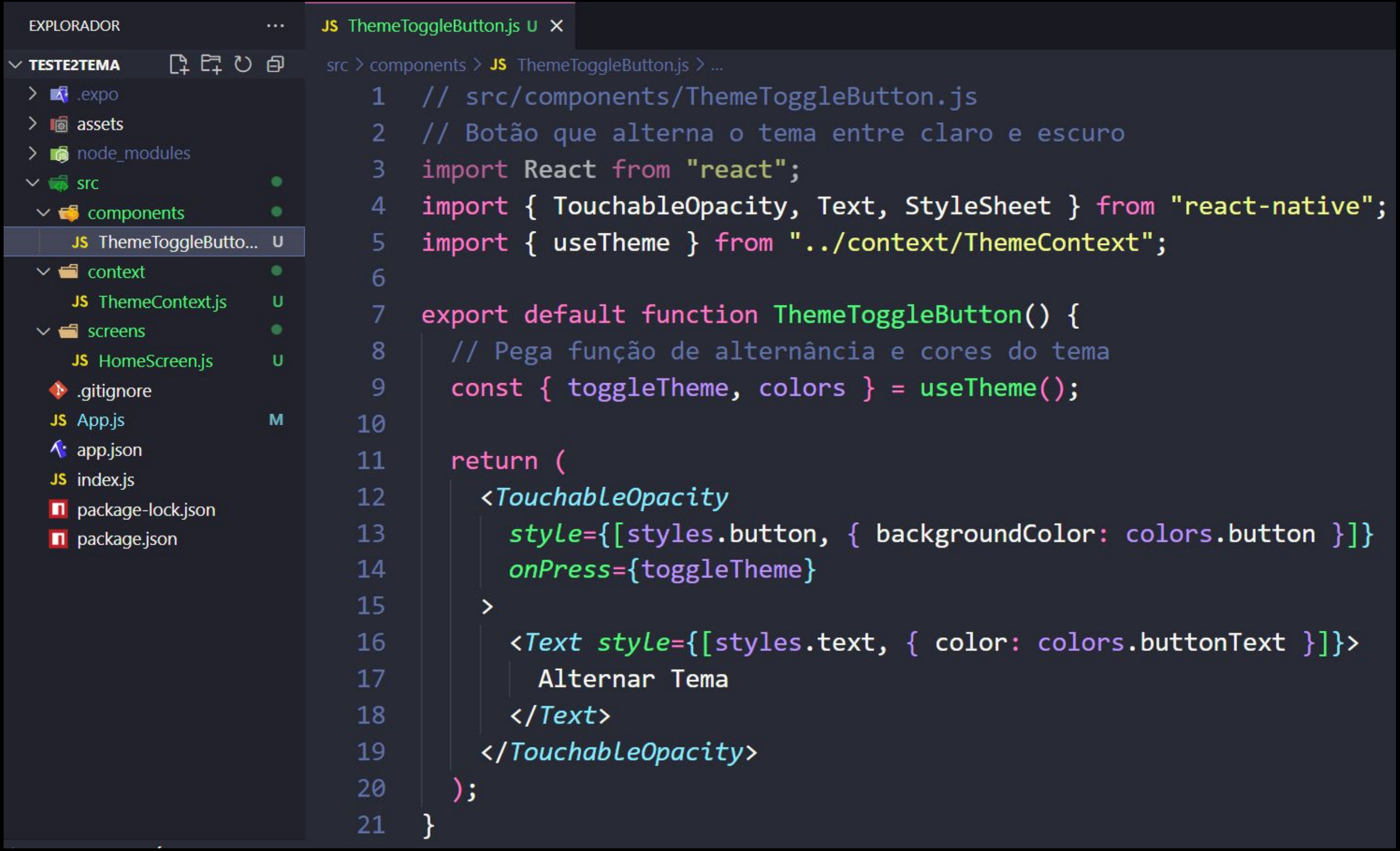
```javascript
// src/screens/HomeScreen.js
// Tela principal que usa o tema
import React from "react";
import { View, Text, StyleSheet } from "react-native";
import { useTheme } from "../context/ThemeContext";
import ThemeToggleButton from "../components/ThemeToggleButton";

export default function HomeScreen() {
  // Pega o tema atual e as cores do contexto
  const { colors, theme } = useTheme();

  return (
    <View style={[styles.container, { backgroundColor: colors.background }]}>
      <Text style={[styles.title, { color: colors.text }]}>
        Tema atual: {theme.toUpperCase()}
      </Text>

      {/* Botão para trocar o tema */}
      <ThemeToggleButton />
    </View>
  );
}
```

Tela HomeScreen.js



```
export default function HomeScreen() {
      <ThemeToggleButton />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1, // ocupa a tela toda
    justifyContent: "center",
    alignItems: "center",
    padding: 20,
  },
  title: {
    fontSize: 22,
    marginBottom: 20,
    fontWeight: "bold",
  },
});
```

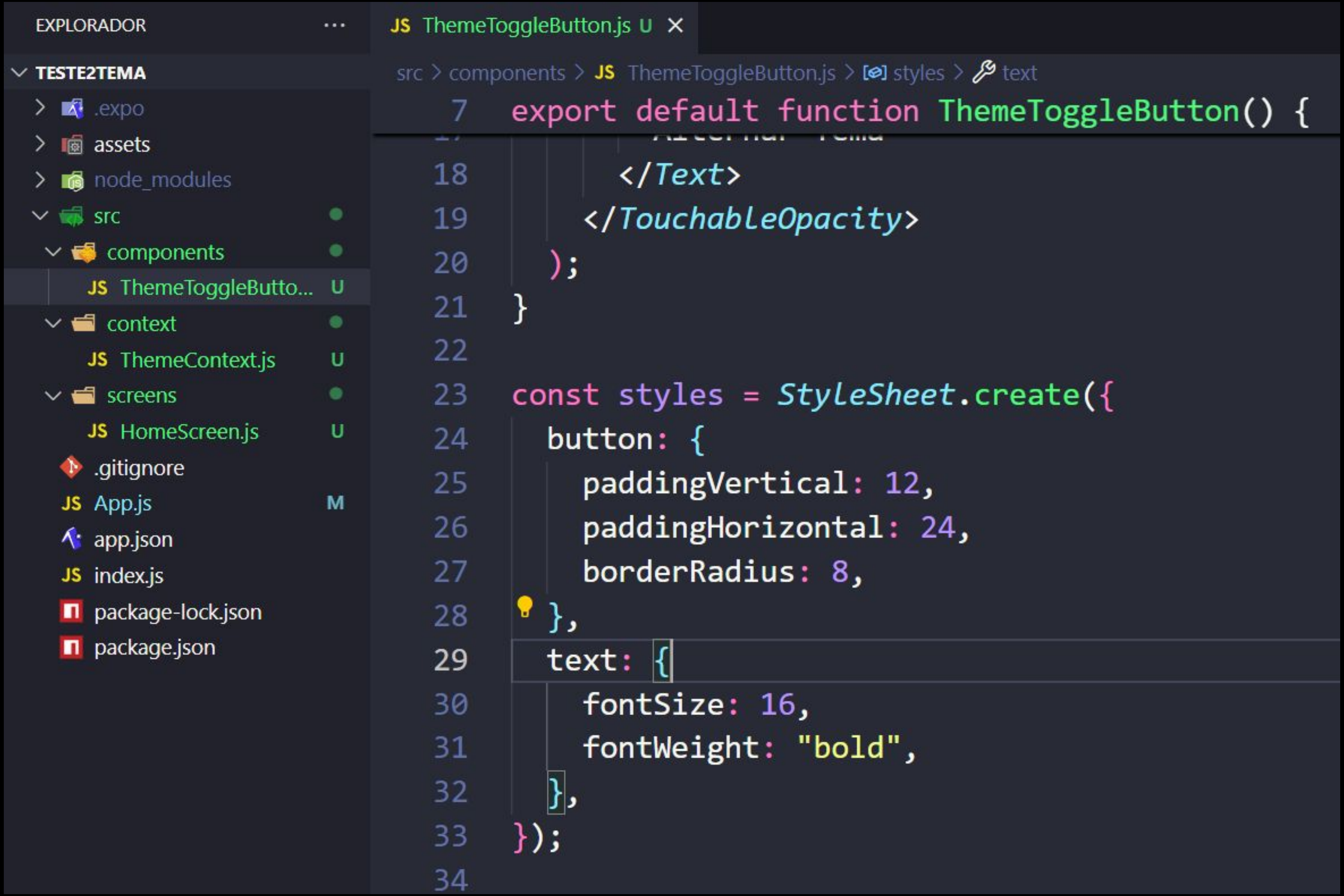
ThemeToggleButton.js



```javascript
// src/components/ThemeToggleButton.js
// Botão que alterna o tema entre claro e escuro
import React from "react";
import { TouchableOpacity, Text, StyleSheet } from "react-native";
import { useTheme } from "../context/ThemeContext";

export default function ThemeToggleButton() {
  // Pega função de alternância e cores do tema
  const { toggleTheme, colors } = useTheme();

  return (
    <TouchableOpacity
      style={[styles.button, { backgroundColor: colors.button }]}
      onPress={toggleTheme}
    >
      <Text style={[styles.text, { color: colors.buttonText }]}>
        Alternar Tema
      </Text>
    </TouchableOpacity>
  );
}
```

ThemeToggleButton.js

FIAP



EXPLORADOR ···
TESTE2TEMA
- .expo
- assets
- node_modules
- src
  - components
    - ThemeToggleButto... U
  - context
    - ThemeContext.js U
  - screens
    - HomeScreen.js U
- .gitignore
- App.js M
- app.json
- index.js
- package-lock.json
- package.json

ThemeToggleButton.js U ×

src > components > ThemeToggleButton.js > styles > text

```
export default function ThemeToggleButton() {
        </Text>
      </TouchableOpacity>
    );
}

const styles = StyleSheet.create({
  button: {
    paddingVertical: 12,
    paddingHorizontal: 24,
    borderRadius: 8,
  },
  text: {
    fontSize: 16,
    fontWeight: "bold",
  },
});
```

Dúvidas?