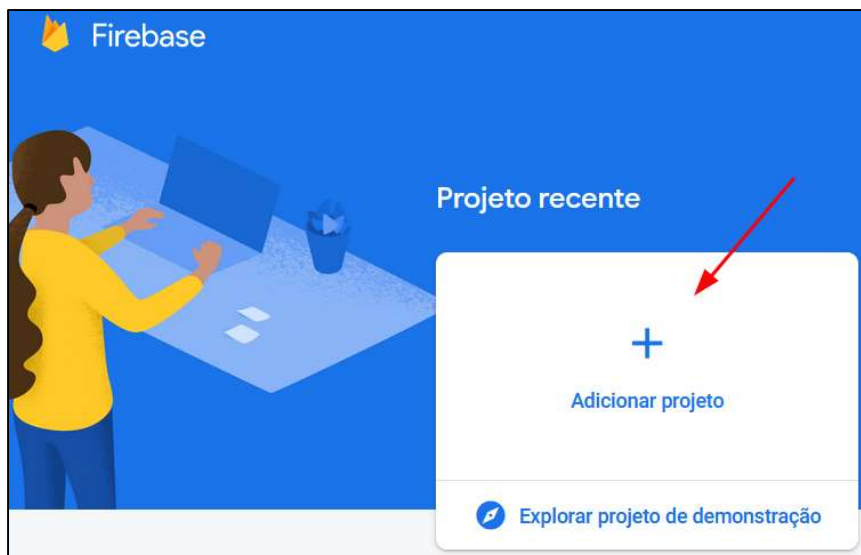


Criando uma aplicação utilizando React-Native com Firebase

Preparando o Ambiente no Firestore

Crie um novo projeto no console de administração do firebase acessando <https://console.firebase.google.com/> estando logado com sua conta google:



Nomeie o projeto à ser criado:



Adicione uma nova aplicação “web” ao projeto criado:



Nomeie a aplicação

×

Adicionar o Firebase ao seu app da Web

1

Registrar app

Apelido do app ?

☐ Also set up **Firebase Hosting** for this app. [Saiba mais](#)

Hosting can also be set up later. There is no cost to get started anytime.

Registrar app

2

Adicionar o SDK do Firebase

Volte para o console de administração:

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK:

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyDM6yte4JWw2Zhahf-QG6-H9jKoIc5H1L0",
  authDomain: "exemplorn-8556c.firebaseio.com",
  databaseURL: "https://exemplorn-8556c-default-rtdb.firebaseio.com",
  projectId: "exemplorn-8556c",
  storageBucket: "exemplorn-8556c.appspot.com",
  messagingSenderId: "72535558741",
  appId: "1:72535558741:web:8bd4019f741e60ba7e22dc"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Observação: ao usar esse método, utilize o [SDK modular do JavaScript](#), que ajuda a reduzir o tamanho do SDK.

Saiba mais sobre o Firebase para Web: [Vamos começar](#), [Referência da API Web SDK](#), [Amstras](#)

Continuar no console

Adicione um serviço firestore


ExemploRN

Plano Spark

1 app + Adicionar app


Escolha um produto para adicionar ao app

Armazene e sincronize dados de app em milissegundos



Authentication

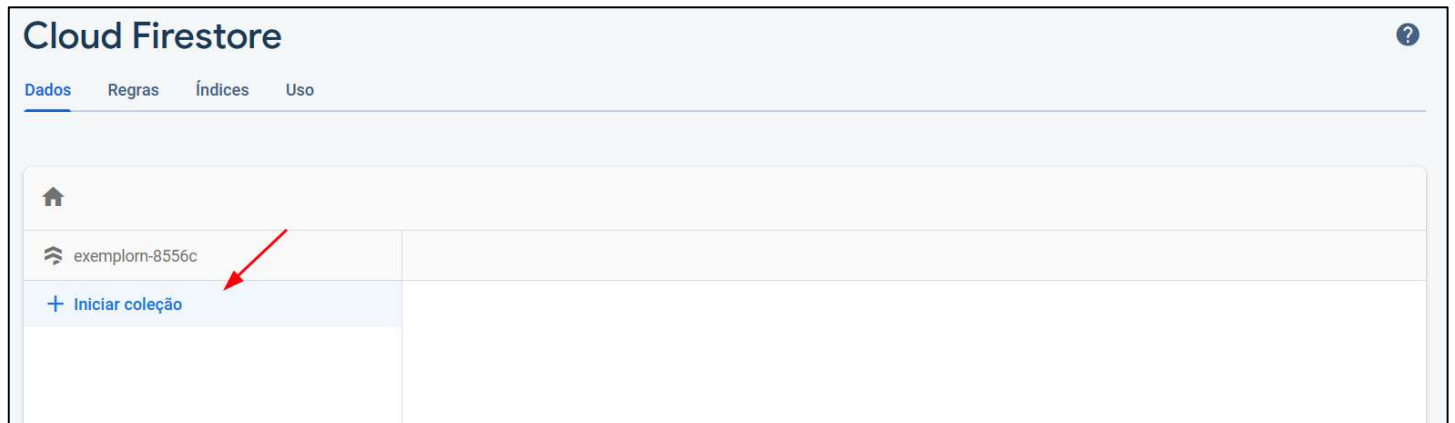
Autenticar e gerenciar usuários



Cloud Firestore

Atualizações em tempo real, consultas eficientes e escalonamento automático

Caso seja a primeira base criada sera solicitado o local do serviço. Após instanciar o serviço do firestore, inicie uma nova coleção:



Nomeie a coleção como “tarefas”:

The screenshot shows a dialog box titled 'Iniciar uma coleção'. It has two steps: '1 Fornecer um código à coleção' and '2 Adicionar o primeiro documento'. The 'Caminho pai' field is empty. The 'Código da coleção' field is filled with 'tarefas'. At the bottom right, there are 'Cancelar' and 'Próxima' buttons. A red arrow points to the 'Próxima' button.

Selecione para gerar um ID automático:

The screenshot shows the same dialog box, but now the 'ID do documento' field is set to 'Código automático'. A red arrow points to the 'Código automático' button. Below this, there's a table for document fields. The table has columns for 'Campo', 'Tipo', and 'Valor'. The first row is empty, and there's a '+' button to add more fields. At the bottom right, there are 'Cancelar' and 'Salvar' buttons.

Campo	Tipo	Valor

Adicione o campo “rotulo” com valor “Primeira Tarefa”:

Iniciar uma coleção

1 ✓ Fornecer um código à coleção — 2 Adicionar o primeiro documento

Caminho pai do documento
/tarefas

ID do documento ?
efDiAQRWzs68YtSQzDX1

Campo	Tipo	Valor
rotulo	= string	Primeira Tarefa

+ Adicionar campo

Cancelar Salvar

Vamos adicionar um novo documento:

Dados Regras Índices Uso

home > tarefas > efDiAQRWzs68...

exemplom-8556c	tarefas	efDiAQRWzs68YtSQzDX1
+ Iniciar coleção	+ Adicionar documento	+ Iniciar coleção
tarefas >	efDiAQRWzs68YtSQzDX1 >	+ Adicionar campo
		rotulo: "Primeira Tarefa"

Selecione para gerar o ID automático:

Adicionar um documento

Caminho pai
/tarefas

ID do documento ?

Código automático

! Obrigatório

Campo	Tipo	Valor
	= string	

+ Adicionar campo

Cancelar Salvar

Adicione o campo "rotulo" com valor "Segunda Tarefa":

Adicionar um documento

Caminho pai
/tarefas

ID do documento ⓘ
DqOdfyqIZwHuTzVil3Lu

Campo	Tipo	Valor
rotulo	= string	Segunda Tarefa

+ Adicionar campo

Cancelar Salvar

Desta maneira já temos 2 registros no firestore:

Dados	Regras	Índices	Uso
home > tarefas > DqOdfyqIZwHuT.			
exemplorn-8556c	tarefas	DqOdfyqIZwHuTzVil3Lu	
+ Iniciar coleção	+ Adicionar documento	+ Iniciar coleção	
tarefas	DqOdfyqIZwHuTzVil3Lu	+ Adicionar campo	rotulo: "Segunda Tarefa"
	efDiAQRWzs68YtSQzDX1		

Criando a Aplicação React Native

Crie uma nova aplicação React Native com Expo:

```
→ react-native-firebase npx create-expo-app notes-expofb
```

Navegue até a pasta do projeto recém-criado:

```
→ react-native-firebase cd notes-expofb
```

Adicione a biblioteca "Safe Area Context":

```
→ notes-expofb git:(master) npx expo-cli install react-native-safe-area-context
```

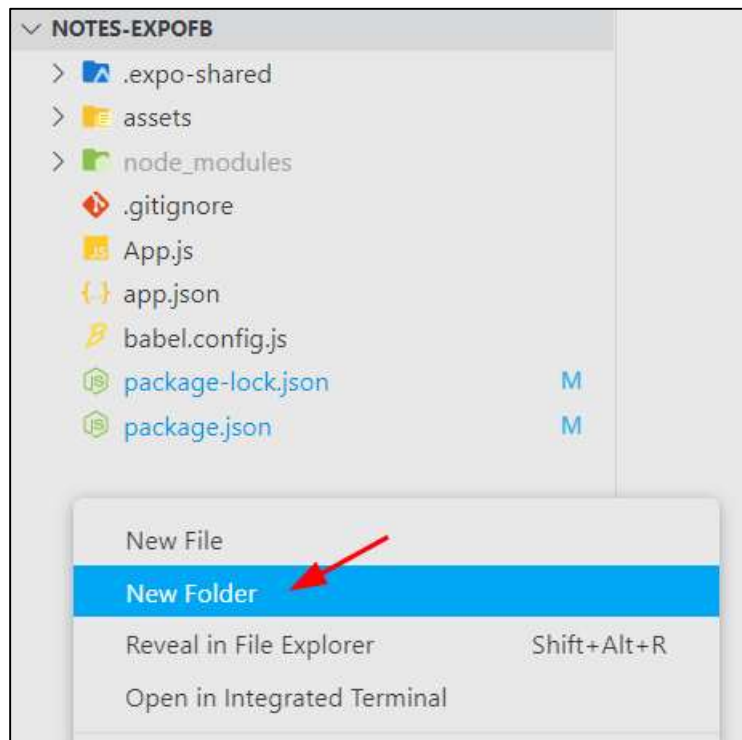
Adicione a biblioteca do firebase:

```
→ notes-expofb git:(master) npx expo-cli install firebase@9.6.11
```

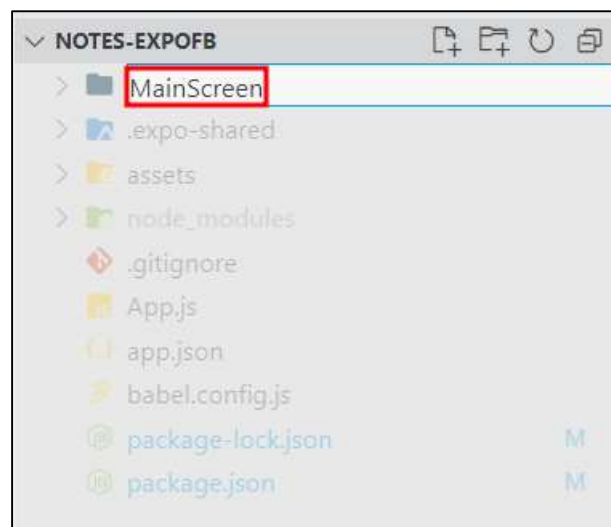
Abra o projeto criado no Visual Studio Code:

```
→ notes-expofb git:(master) code .
```

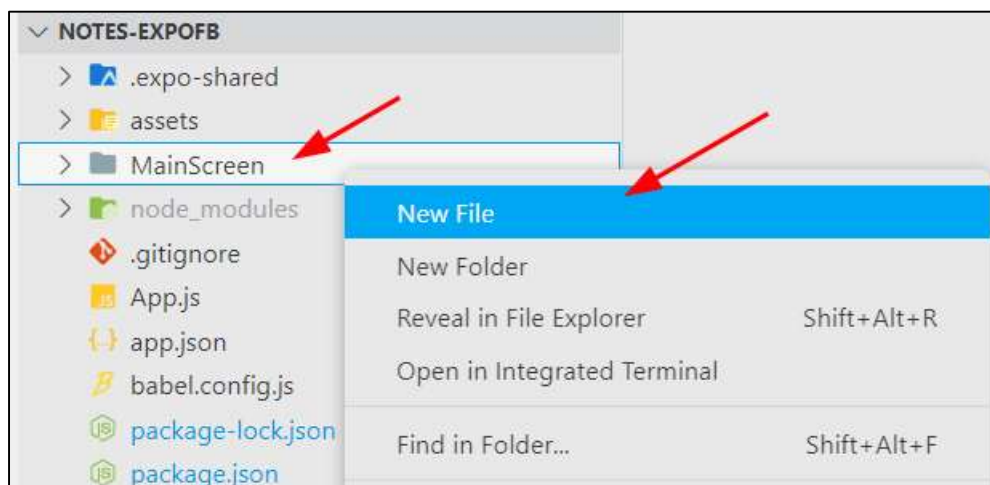
Crie uma nova pasta:



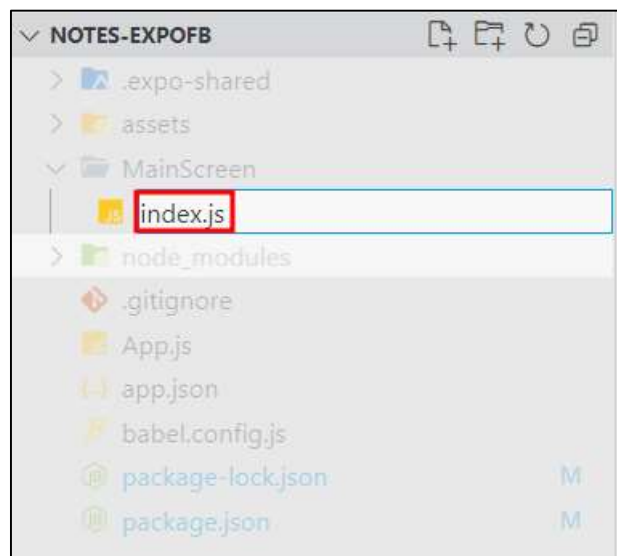
Nomeie como "MainScreen":



Crie um novo arquivo na pasta criada:



Nomeie como "index.js":



Adicione a listagem a seguir no arquivo recém-criado:

```
MainScreen > index.js > ...
1  import { View, Text } from "react-native";
2
3  const MainScreen = () => {
4    return (
5      <View>
6        <Text>Testando</Text>
7      </View>
8    )
9  }
10
11 export default MainScreen
```

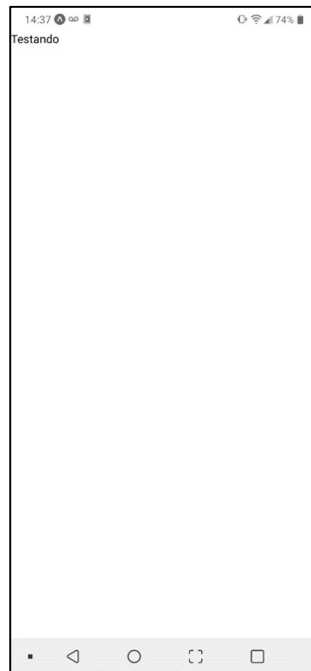
Edite o arquivo "App.js" para ficar como a imagem a seguir:

```
App.js > ...
1  import { StatusBar } from 'expo-status-bar';
2  import { SafeAreaProvider, SafeAreaView } from 'react-native-safe-area-context';
3  import MainScreen from './MainScreen'
4
5  export default function App() {
6    return (
7      <SafeAreaProvider>
8        <SafeAreaView>
9          <MainScreen />
10        </SafeAreaView>
11        <StatusBar style="auto" />
12      </SafeAreaProvider>
13    );
14  }
```

Execute a aplicação:

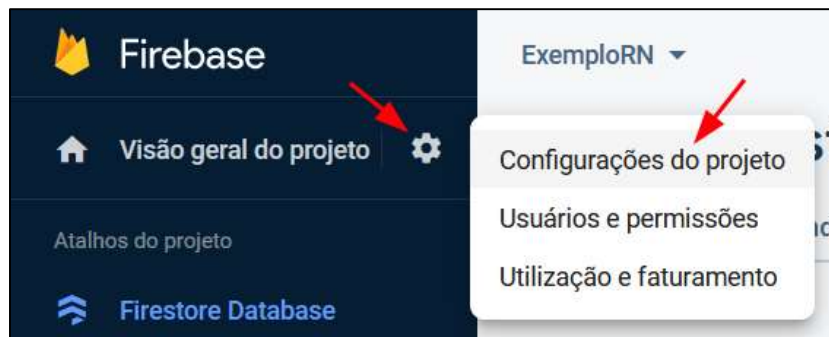
```
→ notes-expofb git:(master) npm start
```

Executando a aplicação no expo deve ser apresentada a tela como a imagem a seguir:

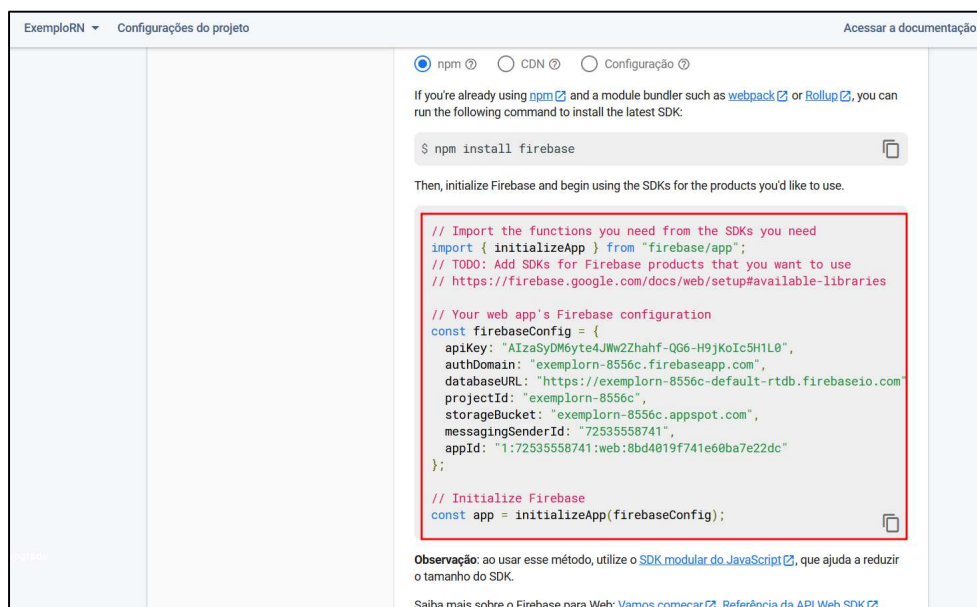


Integrando ao Firebase

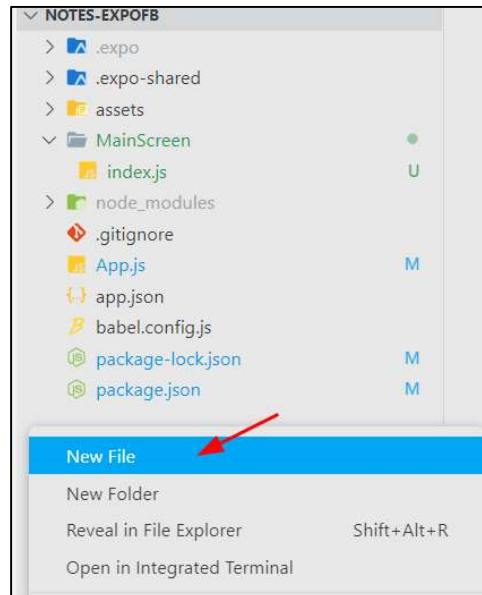
Acesse as configurações do projeto no console do firebase:



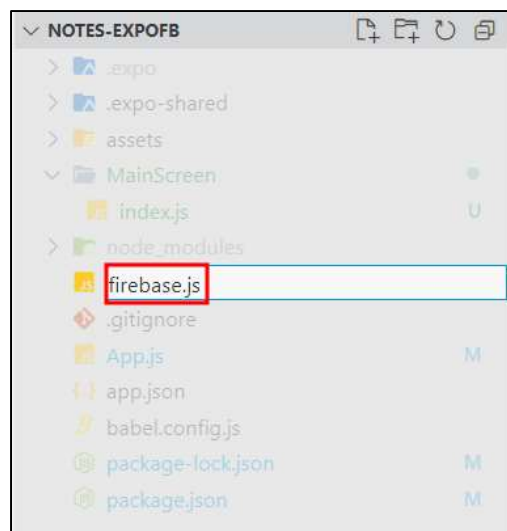
Role a página até encontrar as informações do projeto. Copie as informações:



Crie um arquivo:



Nomeie como "firebase.js"



Cole as informações copiadas do projeto firebase no arquivo recém-criado:

```
firebase.js > ...
1  import { initializeApp } from "firebase/app";
2  // TODO: Add SDKs for Firebase products that you want to use
3  // https://firebase.google.com/docs/web/setup#available-libraries
4
5  // Your web app's Firebase configuration
6  const firebaseConfig = {
7    apiKey: "AIzaSyDM6yte4JWw2Zhahf-QG6-H9jKoIc5H1L0",
8    authDomain: "exemplorn-8556c.firebaseio.com",
9    databaseURL: "https://exemplorn-8556c-default-rtdb.firebaseio.com",
10   projectId: "exemplorn-8556c",
11   storageBucket: "exemplorn-8556c.appspot.com",
12   messagingSenderId: "72535558741",
13   appId: "1:72535558741:web:8bd4019f741e60ba7e22dc"
14 };
15
16 // Initialize Firebase
17 const app = initializeApp(firebaseConfig);
```

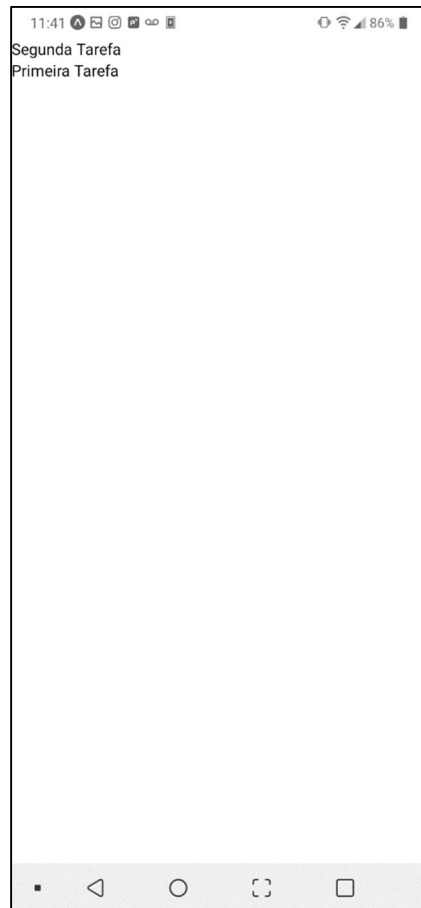
Adicione as linhas em destaque na imagem a seguir:

```
35 firebase.js > ...
1  import { initializeApp } from "firebase/app";
2  import { getFirestore } from 'firebase/firestore'
3
4  const firebaseConfig = {
5    apiKey: "AIzaSyDM6yte4JWw2Zhahf-QG6-H9jKoIc5H1L0",
6    authDomain: "exemplorn-8556c.firebaseio.com",
7    databaseURL: "https://exemplorn-8556c-default-rtdb.firebaseio.com",
8    projectId: "exemplorn-8556c",
9    storageBucket: "exemplorn-8556c.appspot.com",
10    messagingSenderId: "72535558741",
11    appId: "1:72535558741:web:8bd4019f741e60ba7e22dc"
12  };
13
14  const app = initializeApp(firebaseConfig);
15  const firestore = getFirestore()
16
17  export { firestore }
```

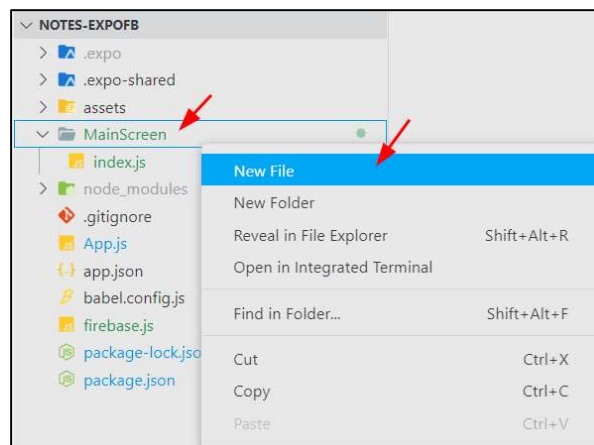
Edite o arquivo "MainScreen/index.js" como na listagem a seguir:

```
MainScreen > index.js > ...
1  import { View, Text, FlatList } from "react-native";
2  import { useEffect, useState } from "react";
3  import { query, collection, getDocs } from "firebase/firestore"
4  import { firestore } from '../firebase'
5
6  const MainScreen = () => {
7    const [ tarefas, setarefas ] = useState([])
8
9    async function loadTarefas() {
10      try {
11        const q = query(collection(firestore, "tarefas"))
12        const queryResult = await getDocs(q)
13        let items = []
14        queryResult.forEach((doc) => {
15          items.push({
16            id: doc.id,
17            rotulo: doc.data().rotulo
18          })
19        })
20        setarefas(items)
21      } catch (error) {
22        console.log(error)
23      }
24    }
25
26    useEffect(() => {
27      loadTarefas()
28    }, [])
29
30    return (
31      <View>
32        <FlatList data={tarefas} renderItem={({ item }) => <Text>{item.rotulo}</Text>}.keyExtractor={item => item.id} />
33      </View>
34    )
35  }
36
37  export default MainScreen
```

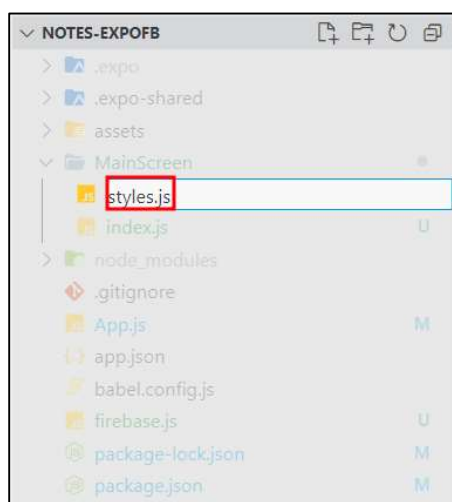
Observe que a aplicação já está consultando as informações no firebase:



Crie um novo arquivo em “MainScreen”:



Nomeie como “styles.js”:



Insira o conteúdo da listagem a seguir no arquivo recém-criado:

```
MainScreen > styles.js > ...
1  import { StyleSheet } from "react-native";
2
3  const styles = StyleSheet.create({
4    container: {
5      padding: 15,
6      backgroundColor: '#fff',
7    },
8    input: {
9      fontSize: 30,
10   },
11   buttonContainer: {
12     alignItems: "center",
13     justifyContent: "center",
14     flexDirection: "row"
15   },
16   button: {
17     padding: 5,
18     flex: 1,
19   },
20   task: {
21     fontSize: 30
22   }
23 });
24
25 export { styles }
```

Edite o arquivo "MainScreen/index.js" como na imagem a seguir:

```
MainScreen > index.js > [0] MainScreen > saveTarefa
1  import { View, Text, FlatList, TextInput, Button } from "react-native";
2  import { useEffect, useState } from "react";
3  import { query, collection, getDocs, addDoc } from "firebase/firestore"
4  import { firestore } from '../firebase'
5
6  import { styles } from './styles'
7
8  const MainScreen = () => {
9    const [ tarefas, setTarefas ] = useState([])
10   const [ tarefaEmEdicao, setTarefaEmEdicao ] = useState({ id: 0, rotulo: '' })
11
12   async function loadTarefas() {
13     try {
14       const q = query(collection(firestore, "tarefas"))
15       const queryResult = await getDocs(q)
16       let items = []
17       queryResult.forEach((doc) => {
18         items.push({
19           id: doc.id,
20           rotulo: doc.data().rotulo
21         })
22       })
23       setTarefas(items)
24     } catch (error) {
25       console.log(error)
26     }
27   }
28
29   async function limpaTarefaEmEdicao() {
30     setTarefaEmEdicao({ id: 0, rotulo: '' })
31   }
32
33   async function saveTarefa() {
34     if(tarefaEmEdicao.id === 0) {
35       await addDoc(collection(firestore, "tarefas"), {rotulo: tarefaEmEdicao.rotulo})
36     }
37     loadTarefas()
38     limpaTarefaEmEdicao()
39   }
40
41   useEffect(() => {
42     loadTarefas()
43   }, [])
44
45   return (
46     <View style={styles.container}>
47       <View>
48         <Text>Tarefa:</Text>
49         <TextInput style={styles.input} value={tarefaEmEdicao.rotulo} placeholder="Digite uma nova tarefa"
50           onChangeText={text => setTarefaEmEdicao({id: tarefaEmEdicao.id, rotulo: text})} />
51       </View>
52       <View style={styles.buttonContainer}>
53         <View style={styles.button}>
54           <Button title="Limpar" onPress={() => limpaTarefaEmEdicao()} />
55         </View>
56         <View style={styles.button}>
57           <Button title="Salvar" onPress={() => saveTarefa()} />
58         </View>
59       </View>
60       <FlatList data={tarefas} renderItem={({ item }) => <Text>{item.rotulo}</Text> keyExtractor={item => item.id} />
61     </View>
62   )
63 }
64
65 export default MainScreen
```


Edite o arquivo "MainScreen/index.js" como na imagem a seguir para inserir a função de editar uma tarefa:

```
MainScreen > index.js > MainScreen > saveTarefa
1  import { View, Text, FlatList, TextInput, Button, TouchableHighlight } from "react-native";
2  import { useEffect, useState } from "react";
3  import { query, collection, getDocs, addDoc, doc, setDoc } from "firebase/firestore"
4  import { firestore } from "../firebase"
5
6  import { styles } from './styles'
7
8  const MainScreen = () => {
9    const [ tarefas, setTarefas ] = useState([])
10   const [ tarefaEmEdicao, setTarefaEmEdicao ] = useState({ id: 0, rotulo: '' })
11
12   async function loadTarefas() {
13     try {
14       const q = query(collection(firestore, "tarefas"))
15       const queryResult = await getDocs(q)
16       let items = []
17       queryResult.forEach((doc) => {
18         items.push({
19           id: doc.id,
20           rotulo: doc.data().rotulo
21         })
22       })
23       setTarefas(items)
24     } catch (error) {
25       console.log(error)
26     }
27   }
28
29   async function limpaTarefaEmEdicao() {
30     setTarefaEmEdicao({ id: 0, rotulo: '' })
31   }
32
33   async function saveTarefa() {
34     if(tarefaEmEdicao.id === 0) {
35       await addDoc(collection(firestore, "tarefas"), {rotulo: tarefaEmEdicao.rotulo})
36     } else {
37       await setDoc(doc(firestore, "tarefas", tarefaEmEdicao.id), {rotulo: tarefaEmEdicao.rotulo}, {merge: true})
38       console.log(tarefaEmEdicao.id)
39     }
40     loadTarefas()
41     limpaTarefaEmEdicao()
42   }
43
44   useEffect(() => {
45     loadTarefas()
46   }, [])
47
48   return (
49     <View style={styles.container}>
50       <View>
51         <Text>Tarefa:</Text>
52         <TextInput style={styles.input} value={tarefaEmEdicao.rotulo} placeholder="Digite uma nova tarefa"
53           onChangeText={text => setTarefaEmEdicao({id: tarefaEmEdicao.id, rotulo: text})} />
54       </View>
55       <View style={styles.buttonContainer}>
56         <View style={styles.button}>
57           <Button title="Limpar" onPress={() => limpaTarefaEmEdicao()} />
58         </View>
59         <View style={styles.button}>
60           <Button title="Salvar" onPress={() => saveTarefa()} />
61         </View>
62       </View>
63       <FlatList data={tarefas} renderItem={({ item }) =>
64         <TouchableHighlight onPress={() => setTarefaEmEdicao(item)}>
65           <Text style={styles.task} >{item.rotulo}</Text>
66         </TouchableHighlight> } keyExtractor={item => item.id} />
67     </View>
68   )
69 }
70
71 export default MainScreen
```


Edite o arquivo "MainScreen/index.js" como na imagem a seguir para inserir a função de remover uma tarefa:

```
1 import { View, Text, FlatList, TextInput, Button, TouchableHighlight, Alert } from "react-native";
2 import { useEffect, useState } from "react";
3 import { query, collection, getDocs, addDoc, doc, setDoc, deleteDoc } from "firebase/firestore"
4 import { firestore } from "../firebase"
5
6 import { styles } from './styles'
7
8 const MainScreen = () => {
9   const [ tarefas, setTarefas ] = useState([])
10   const [ tarefaEmEdicao, setTarefaEmEdicao ] = useState({ id: 0, rotulo: '' })
11
12   async function loadTarefas() {
13     try {
14       const q = query(collection(firestore, "tarefas"))
15       const queryResult = await getDocs(q)
16       let items = []
17       queryResult.forEach((doc) => {
18         items.push({
19           id: doc.id,
20           rotulo: doc.data().rotulo
21         })
22       })
23       setTarefas(items)
24     } catch (error) {
25       console.log(error)
26     }
27   }
28
29   async function limpaTarefaEmEdicao() {
30     setTarefaEmEdicao({ id: 0, rotulo: '' })
31   }
32
33   async function saveTarefa() {
34     if(tarefaEmEdicao.id === 0) {
35       await addDoc(collection(firestore, "tarefas"), {rotulo: tarefaEmEdicao.rotulo})
36     } else {
37       await setDoc(doc(firestore, "tarefas", tarefaEmEdicao.id), {rotulo: tarefaEmEdicao.rotulo}, {merge: true})
38       console.log(tarefaEmEdicao.id)
39     }
40     loadTarefas()
41     limpaTarefaEmEdicao()
42   }
43
44   function deleteTarefa(tarefa) {
45     Alert.alert('Remover Tarefa', "Tem certeza que deseja remover a tarefa: '" + tarefa.rotulo + "'", [
46       { text: 'Cancelar' },
47       {
48         text: 'Remover',
49         onPress: async () => {
50           try {
51             await deleteDoc(doc(firestore, "tarefas", tarefa.id))
52           } catch (error) {
53             console.log(error)
54           }
55         }
56       }
57     ])
58   }
59
60   useEffect(() => {
61     loadTarefas()
62   }, [])
63
64   return (
65     <View style={styles.container}>
66       <View>
67         <Text>Tarefa:</Text>
68         <TextInput style={styles.input} value={tarefaEmEdicao.rotulo} placeholder="Digite uma nova tarefa"
69           onChangeText={text => setTarefaEmEdicao({id: tarefaEmEdicao.id, rotulo: text})} />
70       </View>
71       <View style={styles.buttonContainer}>
72         <View style={styles.button}>
73           <Button title="Limpar" onPress={() => limpaTarefaEmEdicao()} />
74         </View>
75         <View style={styles.button}>
76           <Button title="Salvar" onPress={() => saveTarefa()} />
77         </View>
78       </View>
79       <FlatList data={tarefas} renderItem={({ item }) =>
80         <TouchableHighlight onPress={() => setTarefaEmEdicao(item)} onLongPress={() => deleteTarefa(item)}>
81           <Text style={styles.task}>{item.rotulo}</Text>
82         </TouchableHighlight> } keyExtractor={item => item.id} />
83     </View>
84   )
85 }
86
87 export default MainScreen
```