

Trabalhando com Firebase (Firestore)

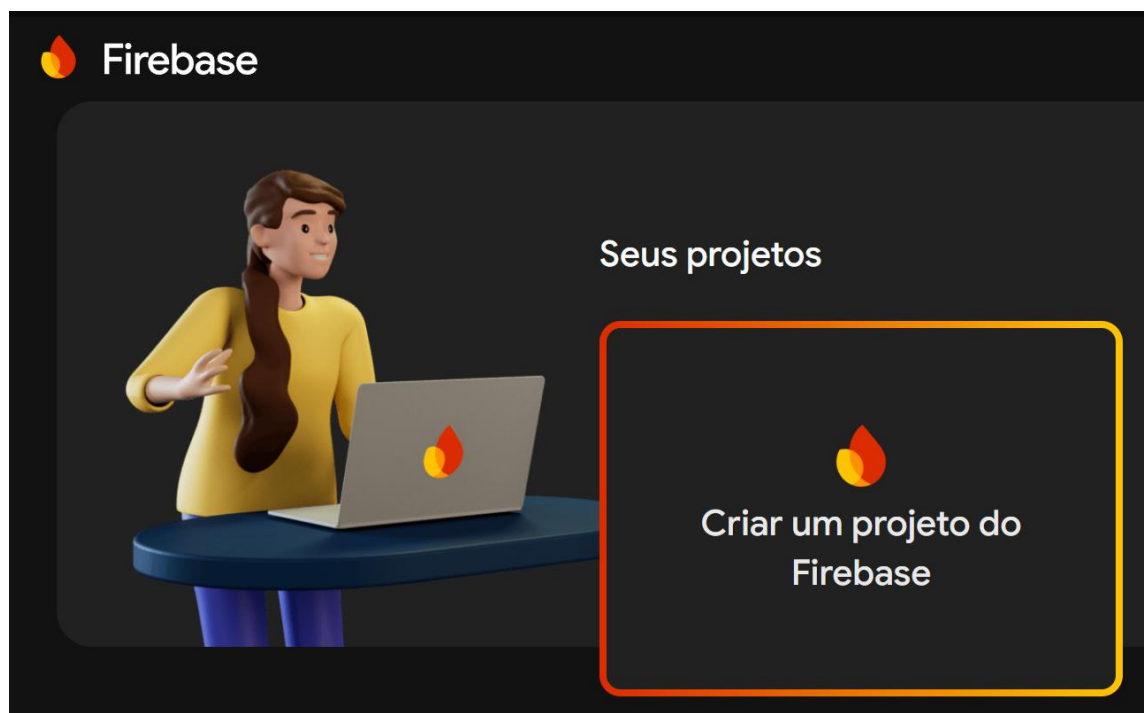
Por João Siles

Olá! Nesse tutorial vamos aprender a ler dados no **Firestore** pertencente ao pacote **Firebase**. Segundo o Google o **Firebase** é uma plataforma de desenvolvimento de aplicações web e móveis da Google, que fornece infraestrutura de **back-end** e ferramentas para construir, lançar e monitorar aplicações de forma rápida e eficiente. É como um kit de ferramentas completo que te permite focar no que realmente importa: a experiência do usuário, deixando a parte mais complexa de configuração e gestão para o **Firebase**. Uma das soluções dele é o banco de dados **NoSQL** chamado **Firestore**.

Vamos acessar o site firebase.google.com e com a nossa conta Google vamos logar e clicar na opção **Go to console**.



Clique na opção **Criar um projeto do Firebase**




Dê um nome a projeto (siga o nome sugerido na imagem)

× Criar um projeto

Vamos começar nomeando o projeto[?]

Nome do projeto

meu-primeiro-firebase

 meu-primeiro-firebase-b1c2b

Já tem um projeto do Google Cloud?

[Adicionar o Firebase ao projeto do Google Cloud](#)

Continuar

Avance nas etapas e desmarque a opção do Google Analytics.

× Criar um projeto

Google Analytics para seu projeto do Firebase

O Google Analytics é uma solução de análise ilimitada e gratuita. Com ele, é possível segmentar, gerar relatórios e muito mais nos seguintes produtos: Firebase Crashlytics, Cloud Messaging, Mensagens no app, Configuração remota, Teste A/B e Cloud Functions.

O Google Analytics ativa:

× Teste A/B ?

× Segmentação de usuários em produtos do Firebase ?

× Registros de navegação estrutural no Crashlytics ?

× Gatilhos do Cloud Functions com base em eventos ?

× Geração de relatórios ilimitada gratuita ?



Ativar o Google Analytics neste projeto
Recomendado

[Anterior](#)

[Criar projeto](#)

Uma vez criado seu projeto você será redirecionado ao painel de controle do **Firebase**. Nesse painel vamos clicar no símbolo `</>` para cadastrar um projeto genérico. Repare que existem outros botões para projetos **Android**, **IOS**, **Flutter** e **Unity** para a criação em seus devidos projetos.



Uma vez que clicamos no botão preencha com os seguintes dados sugeridos:

×

Adicionar o Firebase ao seu app da Web

1

Registrar app

Apelido do app ?

meu-primeiro-firebase

☐

Configure também o **Firebase Hosting** para este app. [Saiba mais](#)

O Hosting pode ser configurado a qualquer momento. Faça isso a quando quiser sem custos financeiros.

Registrar app

2

Adicionar o SDK do Firebase

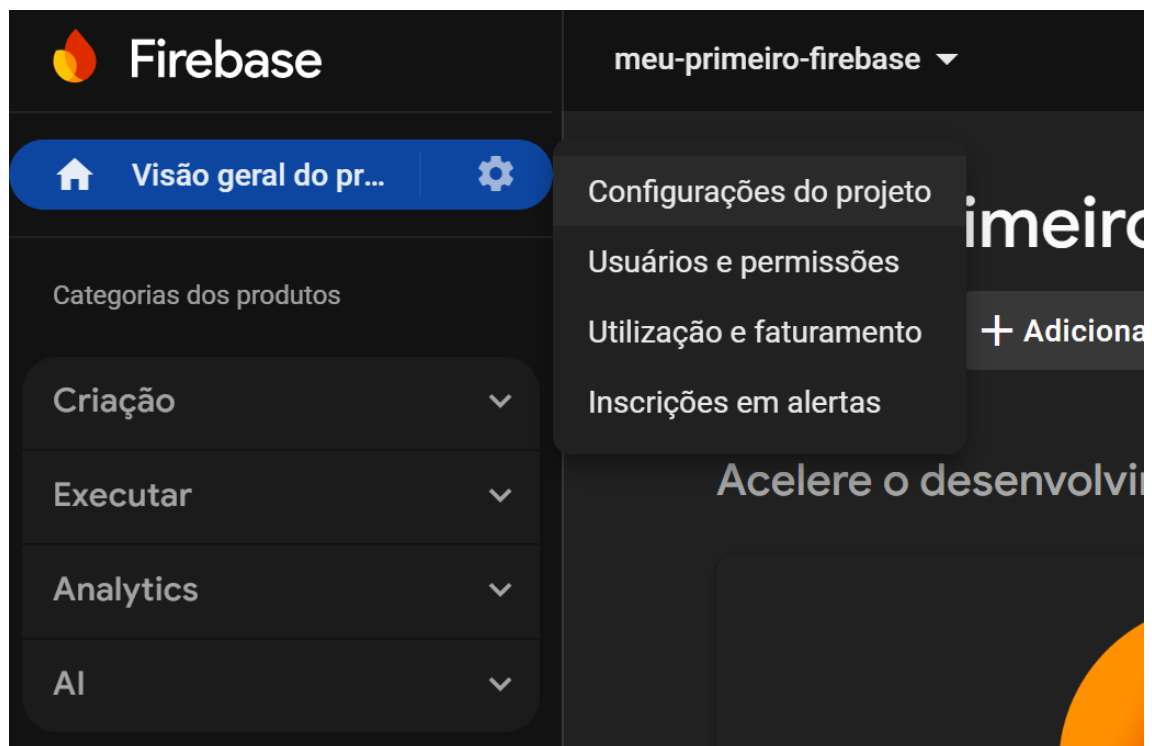
Como resultado ele vai te dar duas coisas: a primeira é a linha de instalação do firebase em um projeto **node** que é **npm install firebase** (execute ela no seu projeto expo **resetado** com o comando **npm run reset-project**), a segunda é um objeto de inicialização com os dados da sua aplicação. Ele é parecido com isso:

```

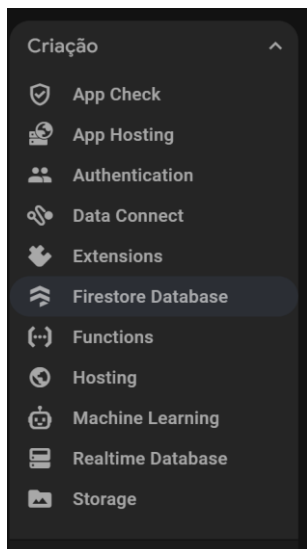
1 // Import the functions you need from the SDKs you need
2 import { initializeApp } from "firebase/app";
3 // TODO: Add SDKs for Firebase products that you want to use
4 // https://firebase.google.com/docs/web/setup#available-libraries
5
6 // Your web app's Firebase configuration
7 const firebaseConfig = {
8   apiKey: "fdfsfdsglksdf;ljk;sdlfkg;sldf",
9   authDomain: "meu-primeiro-firebase-b1c2b.firebaseio.com",
10  projectId: "meu-primeiro-firebase-b1c2b",
11  storageBucket: "meu-primeiro-firebase-b1c2b.firebaseio.com",
12  messagingSenderId: "3453453453453",
13  appId: "1:325434854385093485034580439"
14 };
15
16 // Initialize Firebase
17 const app = initializeApp(firebaseConfig);

```

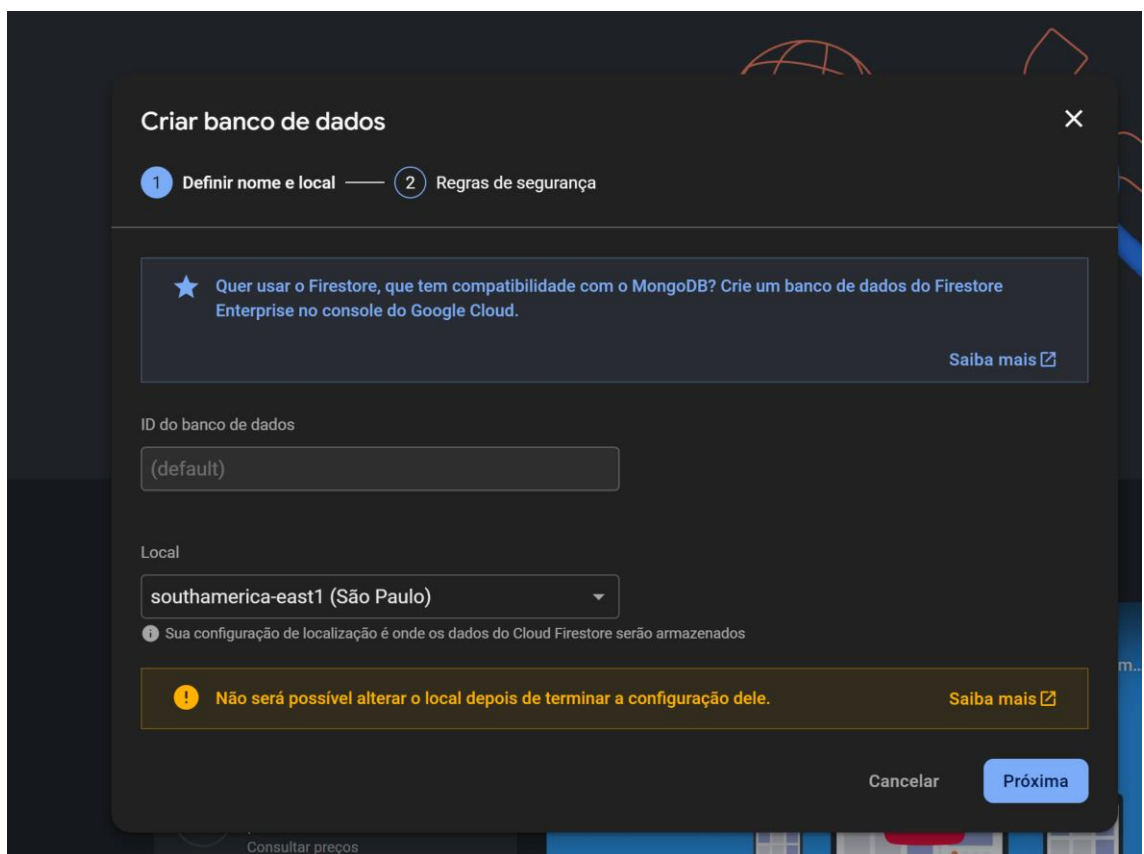
Esse código pode ser recuperado mais tarde nas seguintes opções:



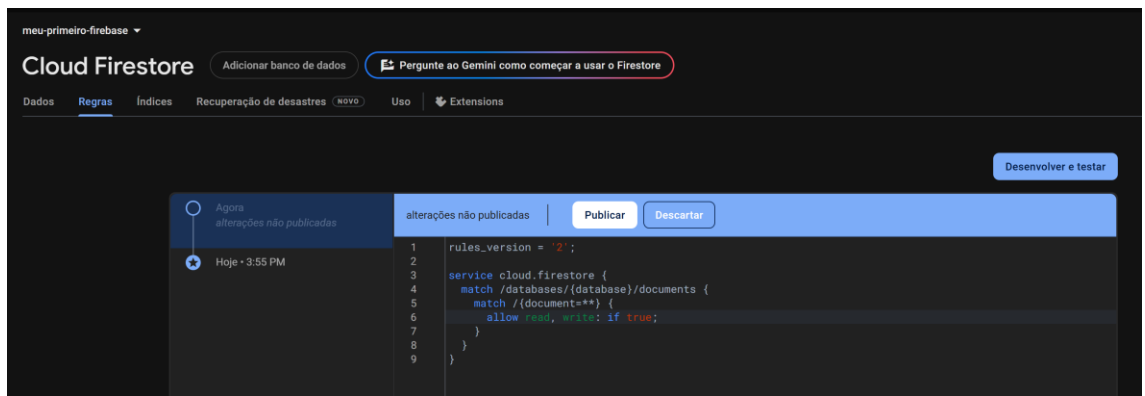
De volta ao painel de controle clique em **Criação** > **Firestore Database**.



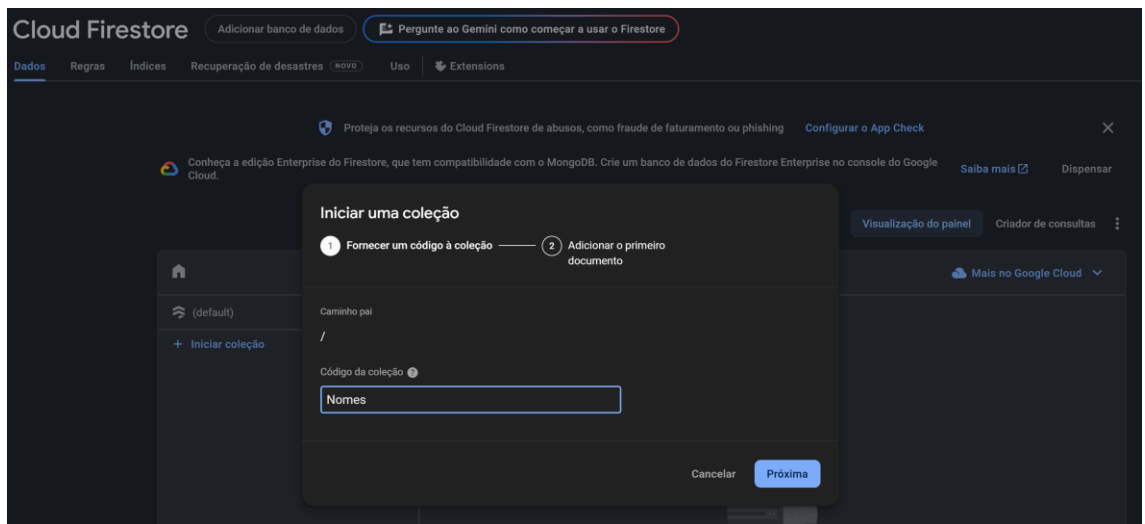
Clique em **Criar banco de dados** e na janela que aparecer escolha o servidor mais próximo igual na imagem:



Avance as etapas e clique em **Criar**. Na tela que aparecer clique em **Regras** substitua no código onde está escrito **false** por **true** e clique em publicar, isso vai permitir que possamos executar todas as operações de **CRUD** ao qual por padrão ele protege no **ambiente de produção**.



Volte para **Dados** e clique em **Iniciar coleção**. Em bancos de dados **NoSQL**, uma coleção é como uma "caixa" que guarda vários documentos. Esses documentos são parecidos com arquivos de texto em formato **JSON**, onde ficam armazenadas as informações. Diferente das tabelas em bancos de dados tradicionais (relacionais), os documentos dentro de uma mesma coleção não precisam ter o mesmo formato. Isso significa que cada documento pode ter campos diferentes, o que dá mais liberdade para organizar os dados de forma flexível. Preencha os dados como na imagem:



E também com os seguintes dados (o **ID do documento** é preenchido ao clicar em **Código automático**):

Iniciar uma coleção



Fornecer um código à coleção

2

Adicionar o primeiro documento

Caminho pai do documento

/Nomes

ID do documento ?

B79q3ulh7GqN0RsJaQF9

Campo

Tipo

Nome

= string



String

João

Campo

Tipo

Sobrenome

= string



String

Siles

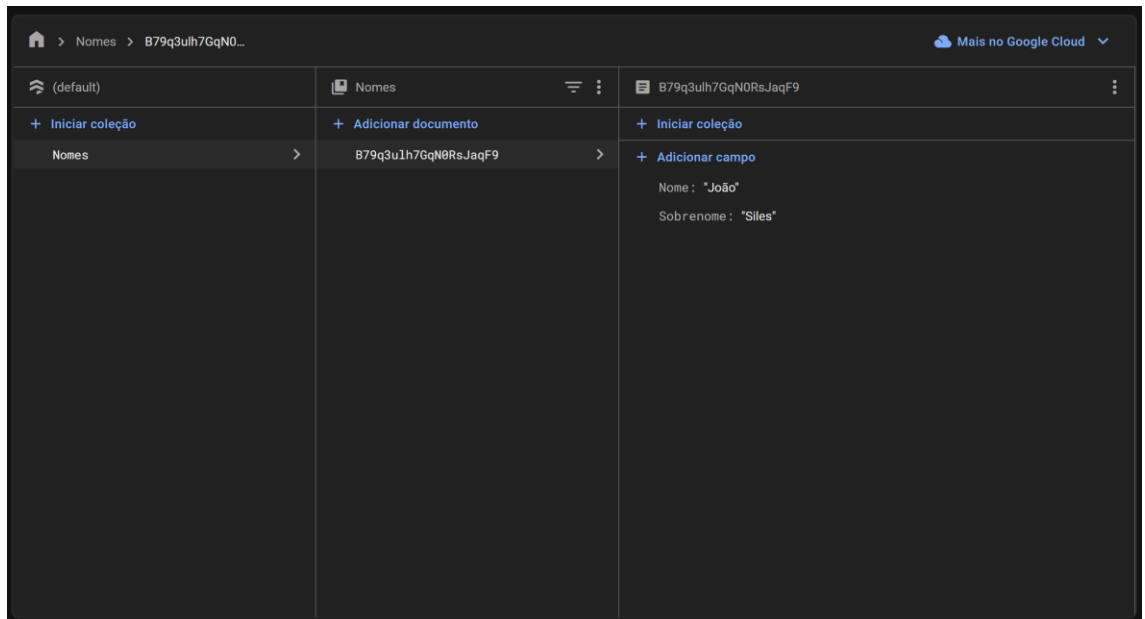


Adicionar campo

Cancelar

Salvar

Clicar em **Adicionar documento** permite criar mais registros. Um **documento** em **NoSQL** é como um arquivo que guarda informações organizadas em pares de "**chave**" e "**valor**".



No seu projeto **Expo** criado (e resetado) digite o seguinte código no arquivo **index.tsx**:

```

1  import firebase from 'firebase/compat/app';
2  import 'firebase/compat/firestore';
3
4  const firebaseConfig = {
5    apiKey: "",
6    authDomain: "",
7    projectId: "",
8    storageBucket: "",
9    messagingSenderId: "",
10   appId: ""
11 };
12
13
14  firebase.initializeApp(firebaseConfig);
15
16
17  import React, { useEffect, useState } from 'react';
18  import { FlatList, Text, View } from 'react-native';
19
20  export default function App() {
21     const [nomes, setNomes] = useState([]);
22
23     useEffect(() => {
24         const fetchData = async () => {
25             const nomesCollection = firebase.firestore().collection('Nomes');
26             const snapshot = await nomesCollection.get();
27
28             const data = [];
29             snapshot.forEach((doc) => {
30                 data.push({ id: doc.id, ...doc.data() });
31             });
32
33             setNomes(data);
34         };
35
36         fetchData();
37     }, []);
38
39     return (
40         <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
41             <Text>Lista de Nomes:</Text>
42             <FlatList
43                 data={nomes}
44                 keyExtractor={(item) => item.id}
45                 renderItem={({ item }) => (
46                     <View>
47                         <Text>{item.Nome} {item.Sobrenome}</Text>
48                     </View>
49                 )}
50             />
51         </View>
52     );
53 }
54

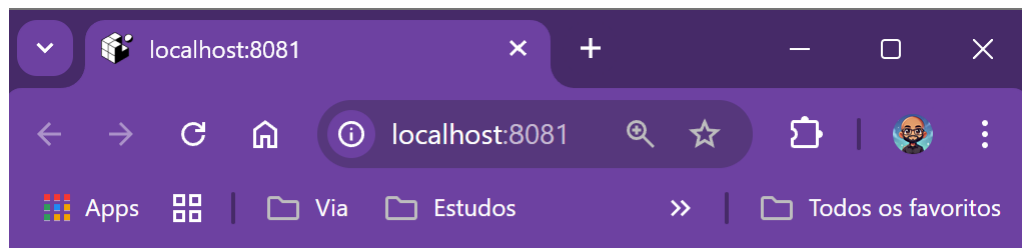
```

Resumidamente o código faz o seguinte:

- Conecta ao Firebase usando as informações do projeto (linhas 1 a 14).
- Declara o estado de **nomes** usando **UseState**.
 - **useState([])** diz que a caixa começa **vazia**.

- **useState** é usado para guardar informações que mudam enquanto o app está rodando.
- No exemplo que você mandou:
- **nomes** é como uma caixa onde guardamos os nomes.
- **setNomes** é a função que coloca novos nomes dentro da caixa.
- Com o **useEffect** acessamos a coleção "**Nomes**" no banco de dados Firestore.
 - Esse código busca os nomes no **Firebase** e guarda na caixa (**nomes**) com **setNomes**.
 - Pense no **useEffect** como um aviso para o React: "assim que o app começar, faça isso aqui".
- Para mais tipos de queries consulte a documentação (<https://firebase.google.com/docs/firestore/query-data/queries?hl=pt-br>)

Ao executar o projeto a seguinte tela deverá aparecer:



index

Lista de Nomes:

Barra Junior

Lais Soares

João Siles