**Utilizando Firebase**

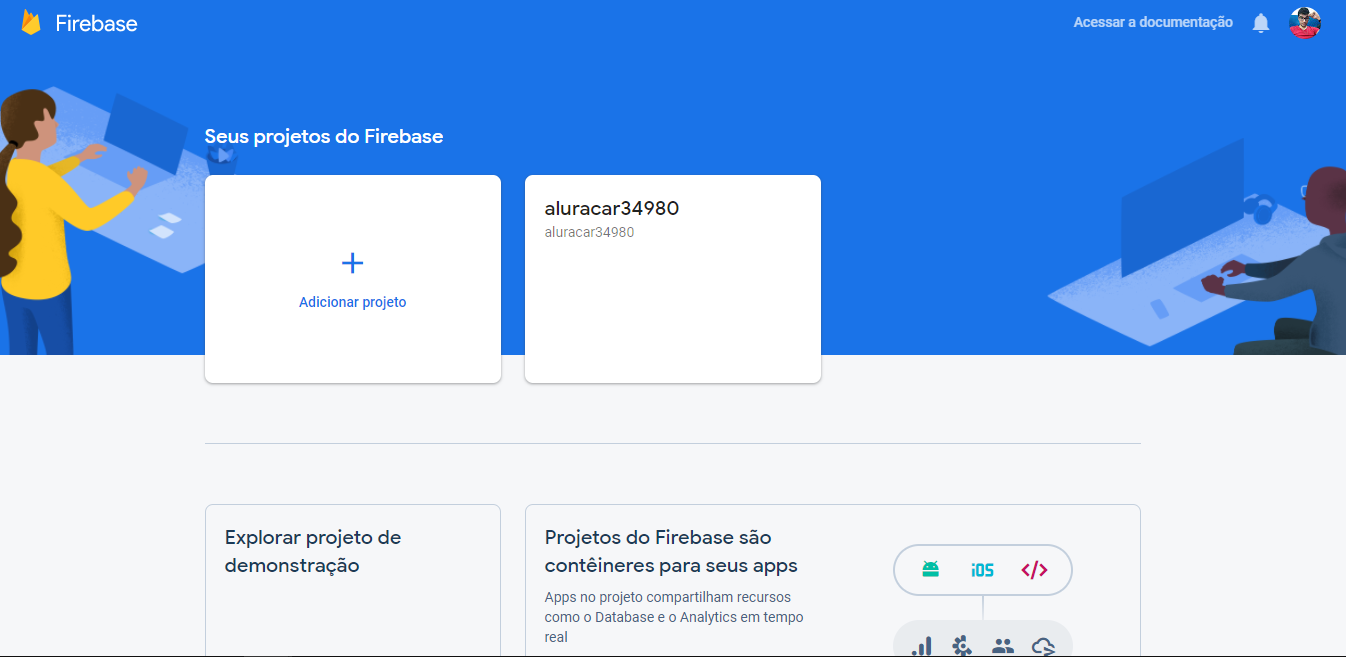
- Criando um webserver de teste

Crie uma conta no firebase : <https://firebase.google.com/>

Clicar em Ir para Console

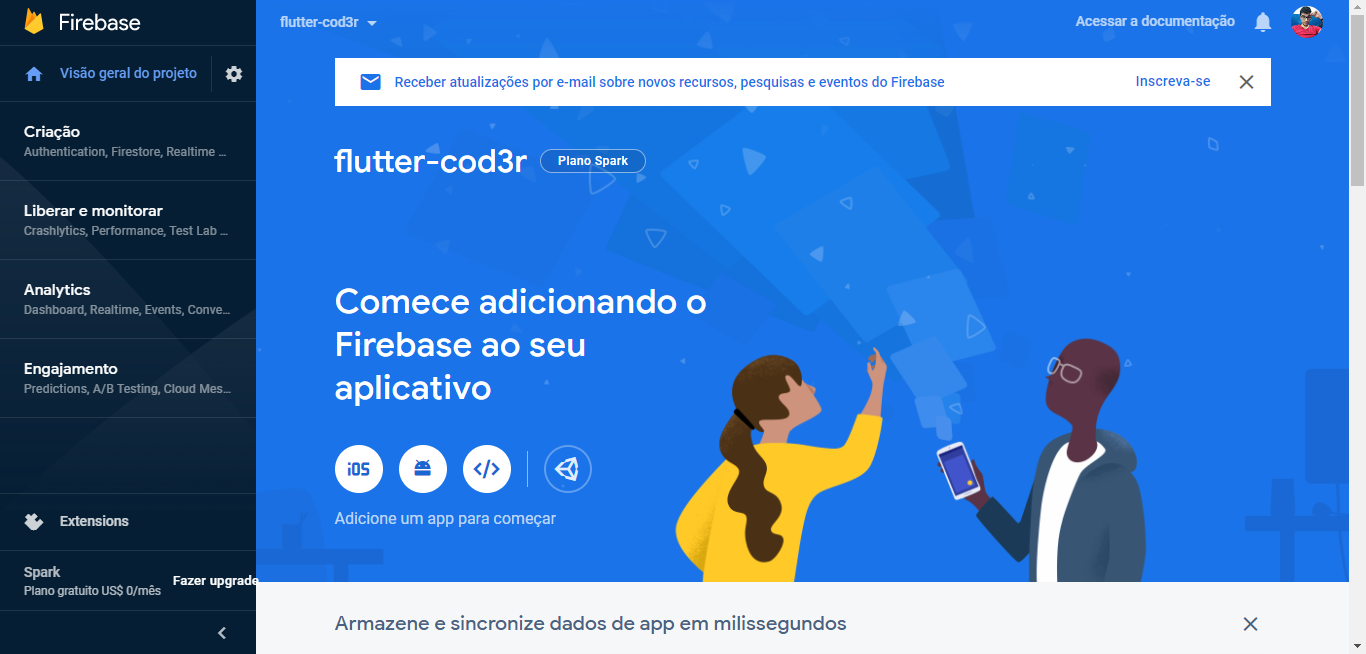


Clique em Adicionar projeto



Preencha os dados, se quiser use o google analitics

Após criar o projeto o acesse.

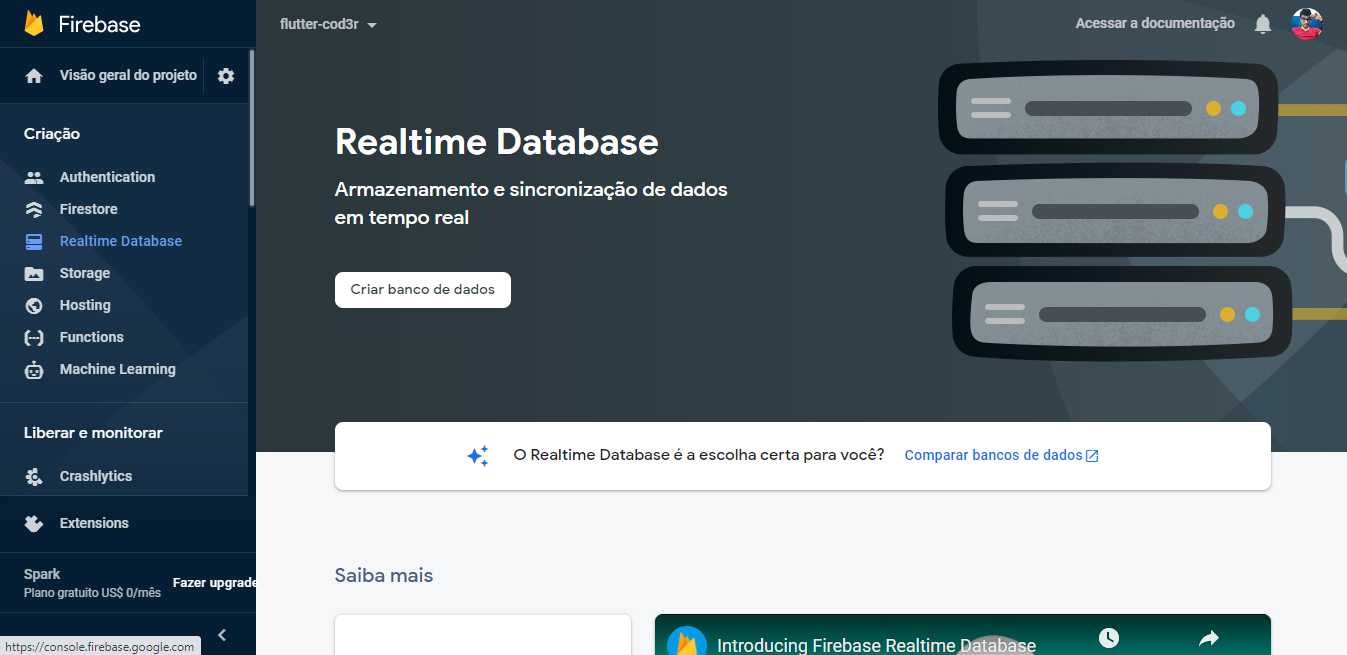


**Armazenando Dados (API Teste) - Realtime**

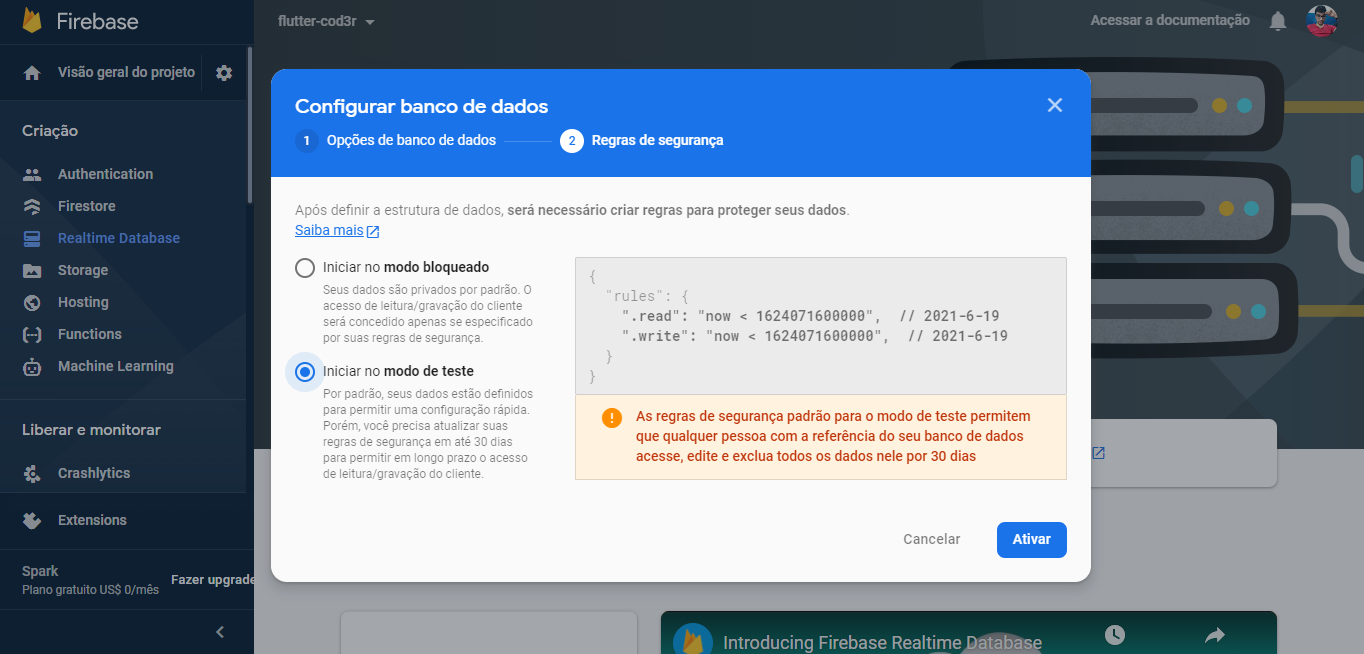
Selecione realtime database



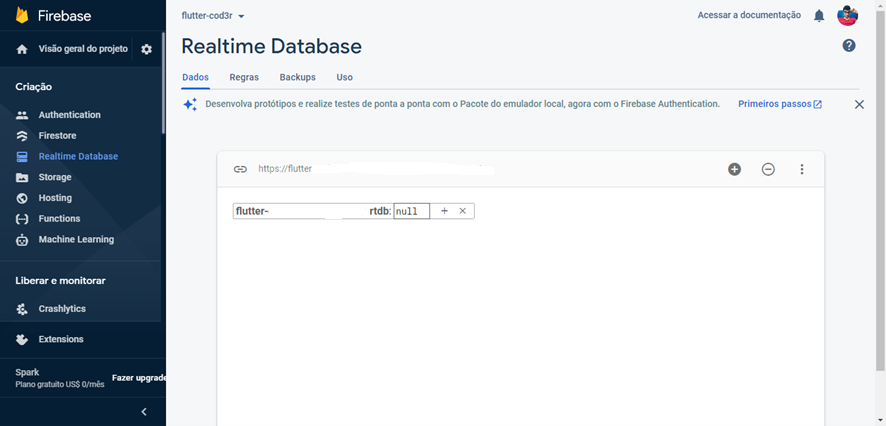
Clique em criar banco de dados



Inicie no modo teste



Dessa forma já fica disponível a url para acessar esse banco de dados pela aplicação



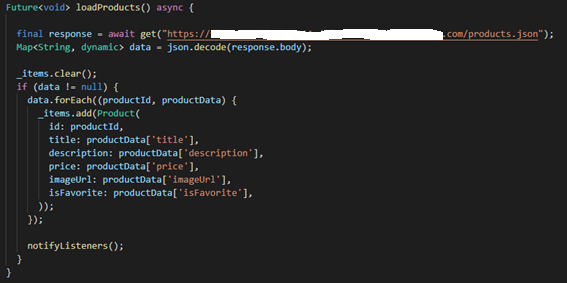
Agora você pode efetuar chamadas http (get, post, put etc...) para essa url, e automaticamente o firebase irá salvar e manipular os dados.

Obs\* o final da chamada deve terminar com um .json, como o exemplo a seguir. https://flutter&\*¨¨%$&#%&\*&/products.json

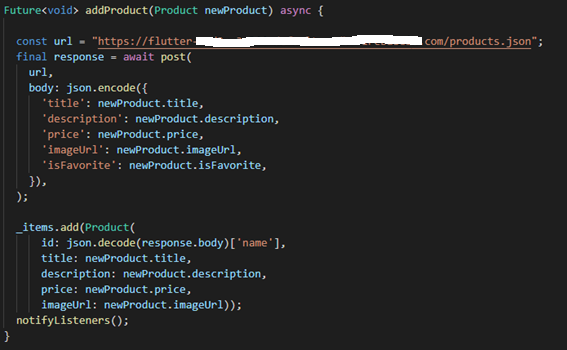
Nessa tela clicando em + você também pode cadastrar novas coleções de dados para usar como teste

Exemplo utilização:

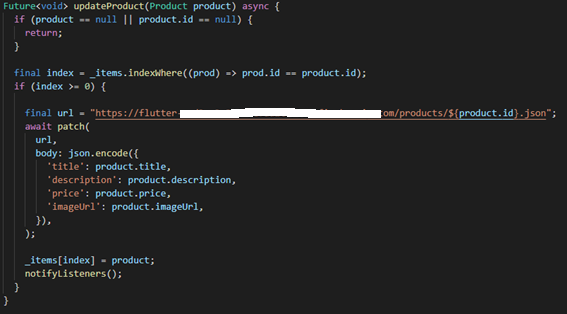
Lendo dados:



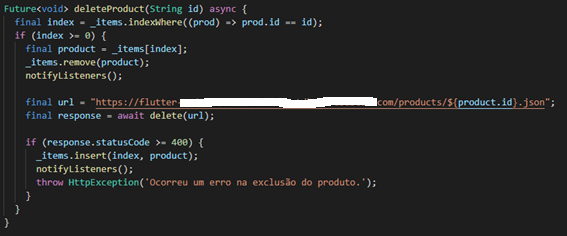
Adicionando dados:



Alterando dados:

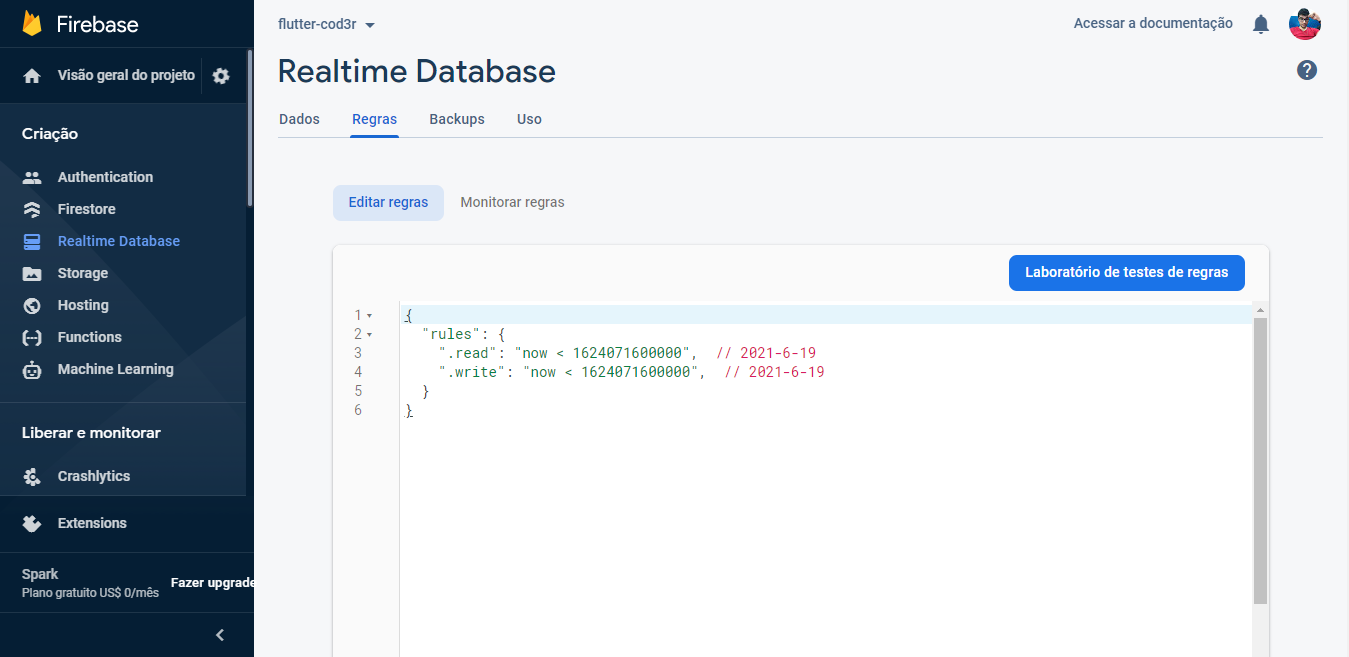


Deletando dados:

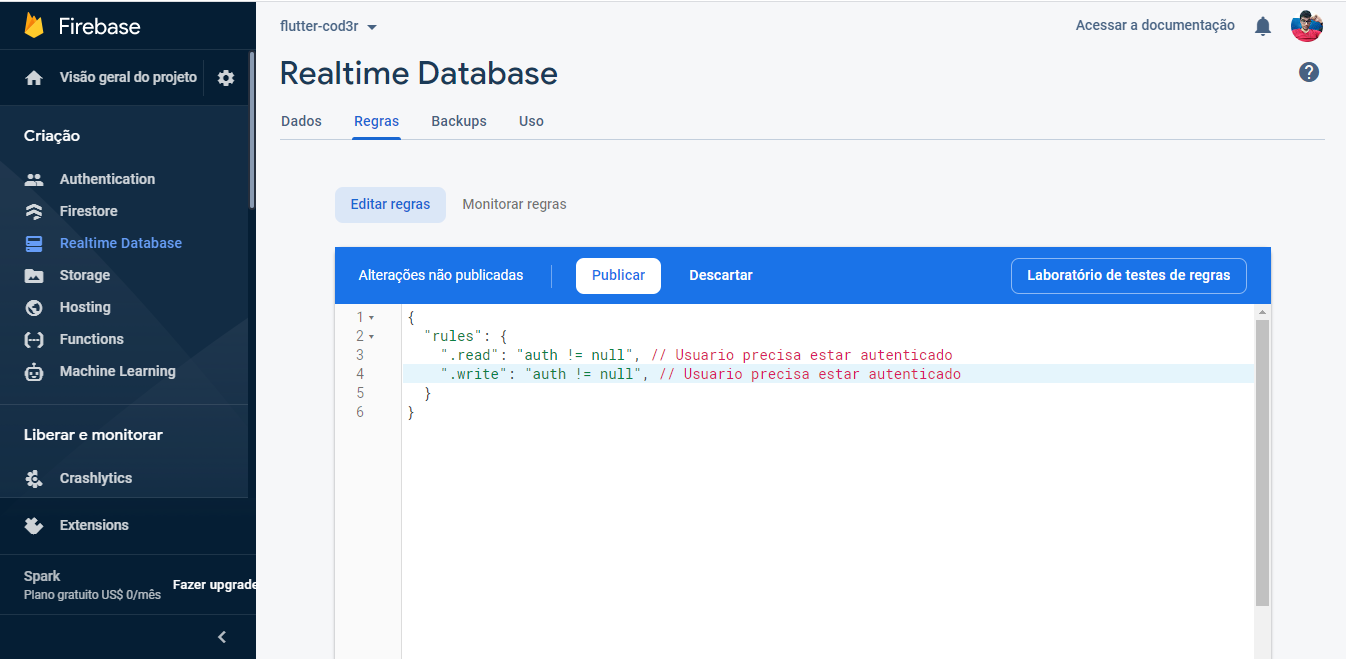


**Autenticação (API)**

Acesse regras

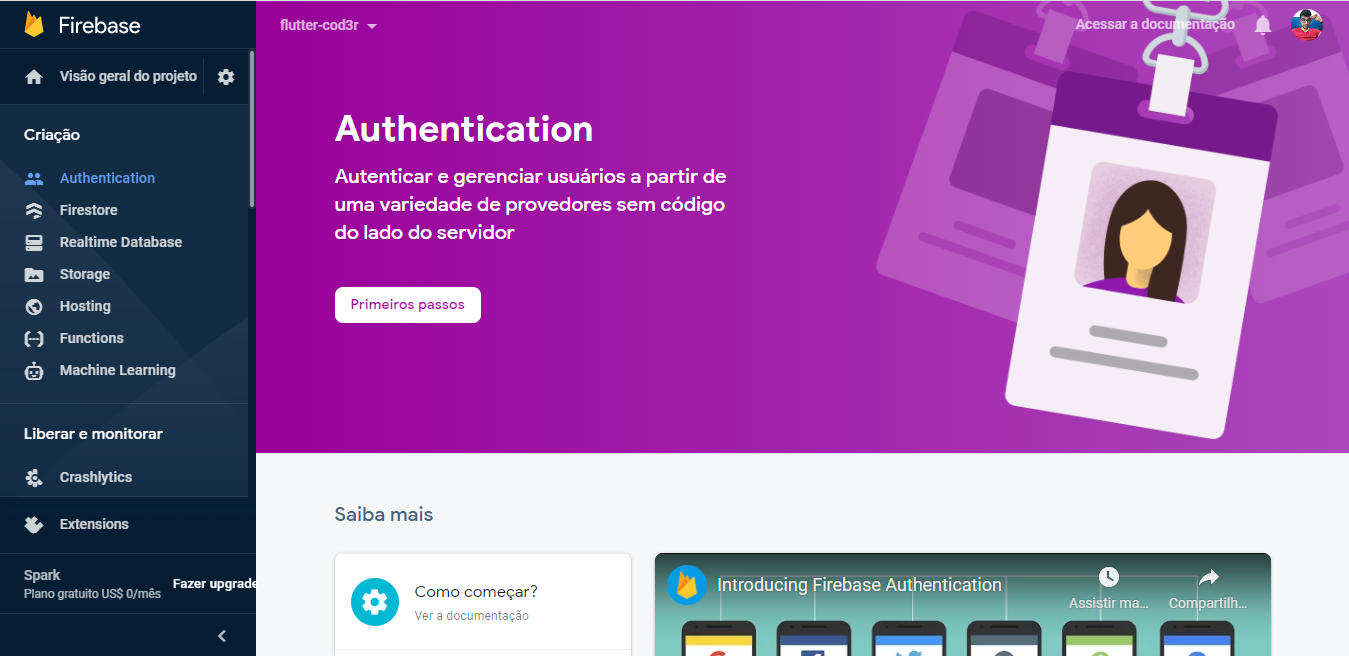


Adicione regras clique em publicar



Obs\* para teste foram aplicadas regras básicas, onde o usuário apenas precisa estar logado para efetuar alterações nas coleções de dados. É possível a configuração de regras mais específicas, como nível de acesso (usuário admin, usuário comum), e definir oque cada tipo de usuário poderá manipular.

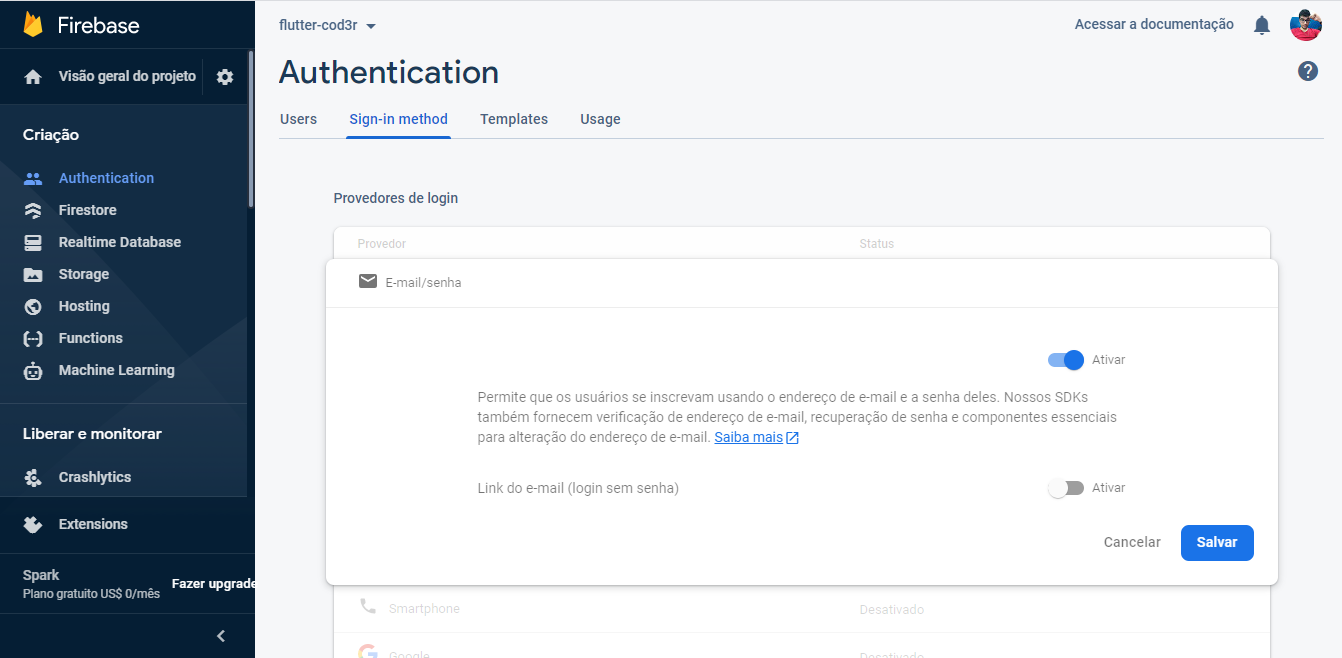
Agora clique em Authentication. E em seguida primeiros passos.



Em Sign-in method pode ser escolhido diversos provedores de login como google, facebook etc..

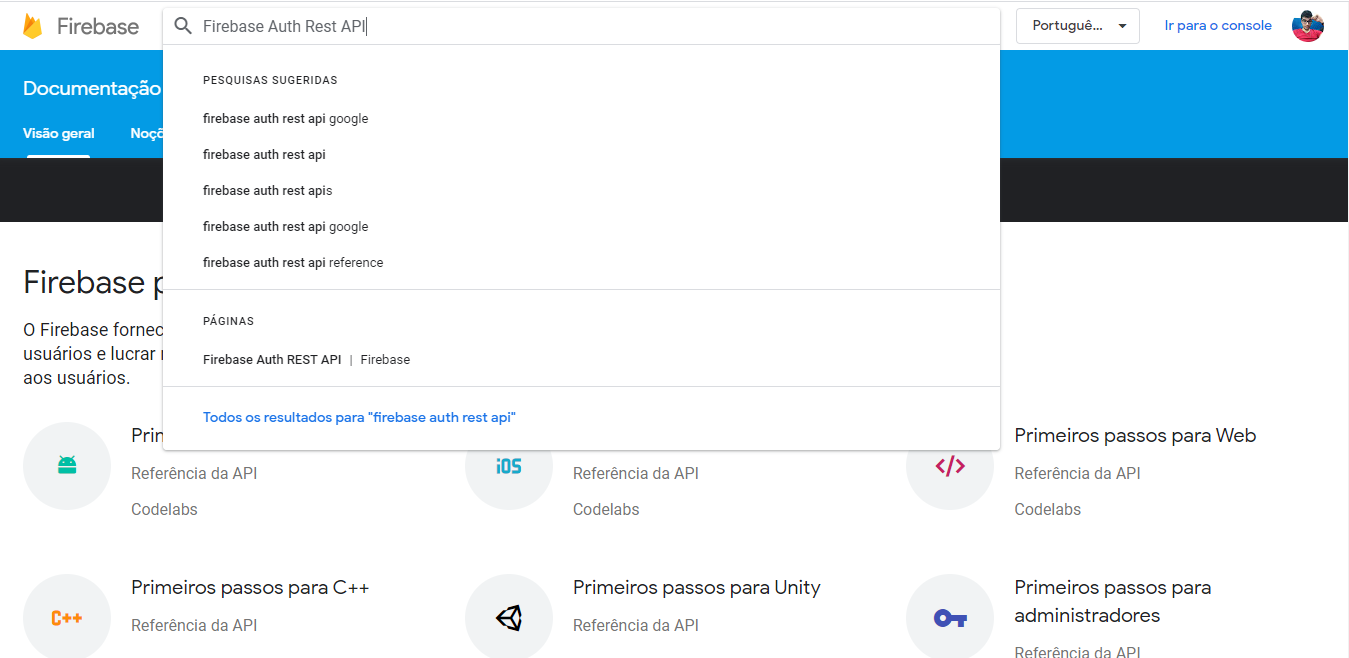
No teste iremos utilizar email/senha.

E clique em ativar.



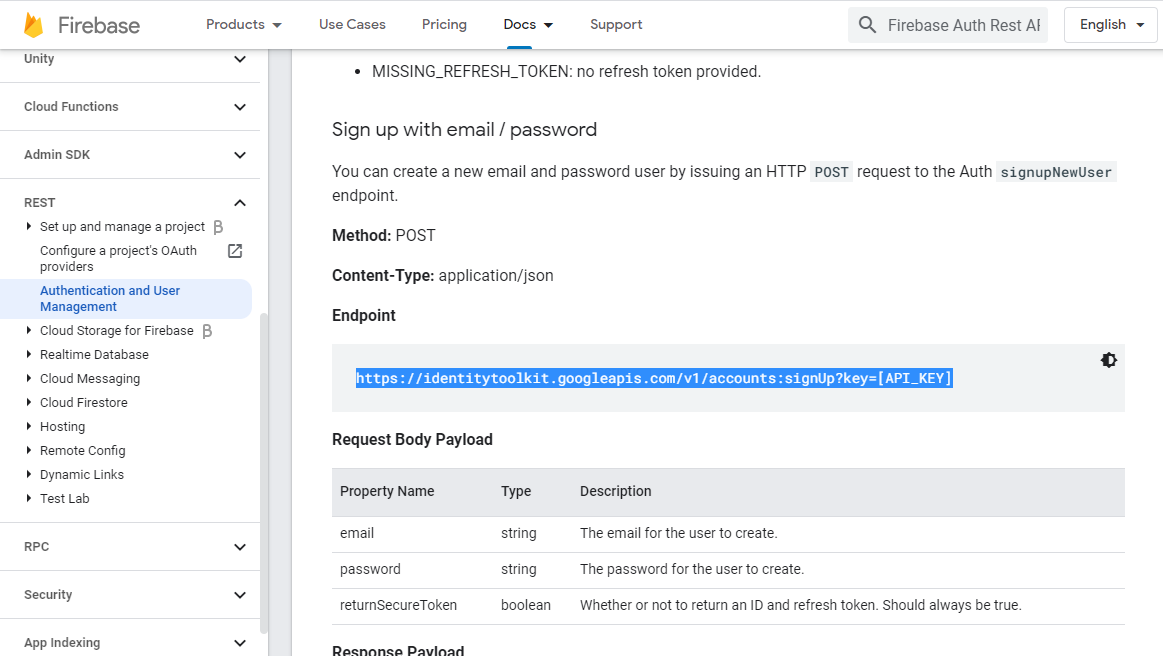
Agora é preciso acessar <https://firebase.google.com/docs> para pegar a url que será utilizada para o serviço de autenticação.

Agora pesquise por Firebase Auth Rest API e acesse a página.

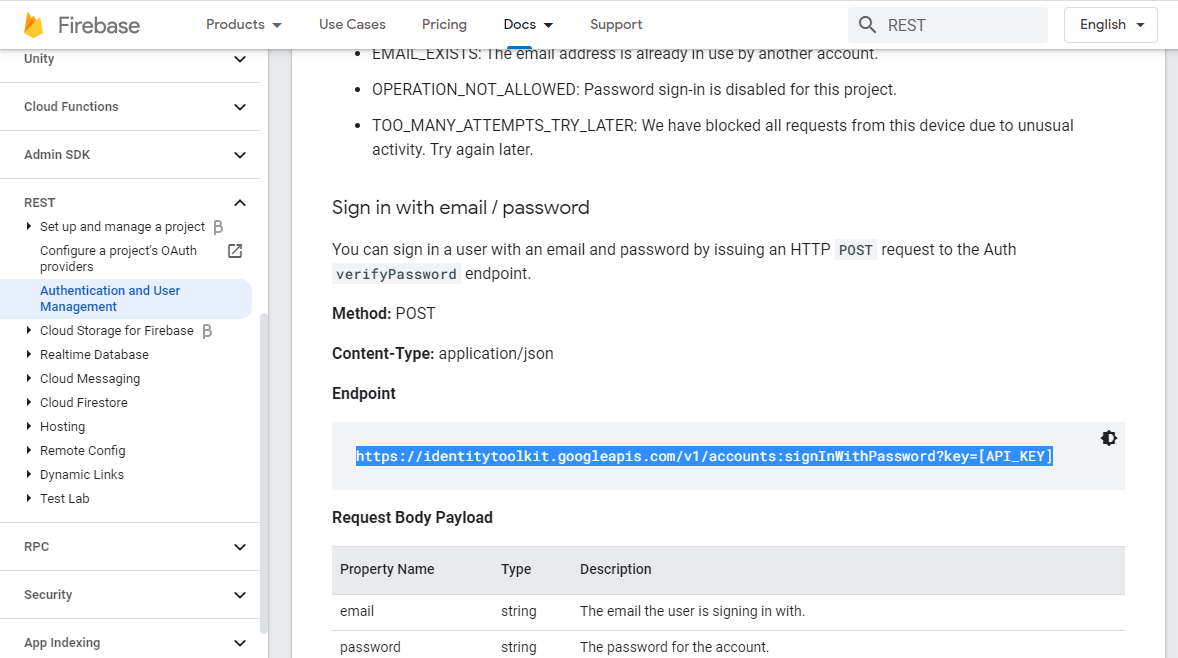


Em seguida na documentação iremos procurar pelo tópico “Sign up with email / password”.

E então podemos identificar a url que sera utilizada pra efetuar o processo de cadastro de usuario.



E procurando pelo tópico “Sign in with email / password” podemos achar a url para efetuar a autenticação.

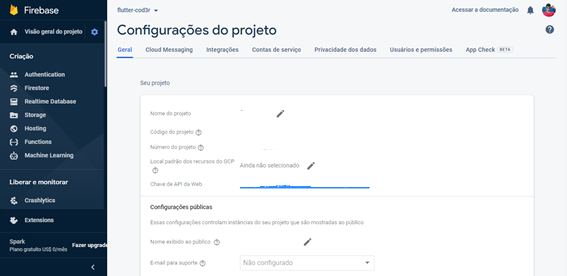


Repare que essas urls tem uma chave dinâmica [API\_KEY]

Para encontrar essa chave acesse Visão geral do projeto/Configuração do projeto.

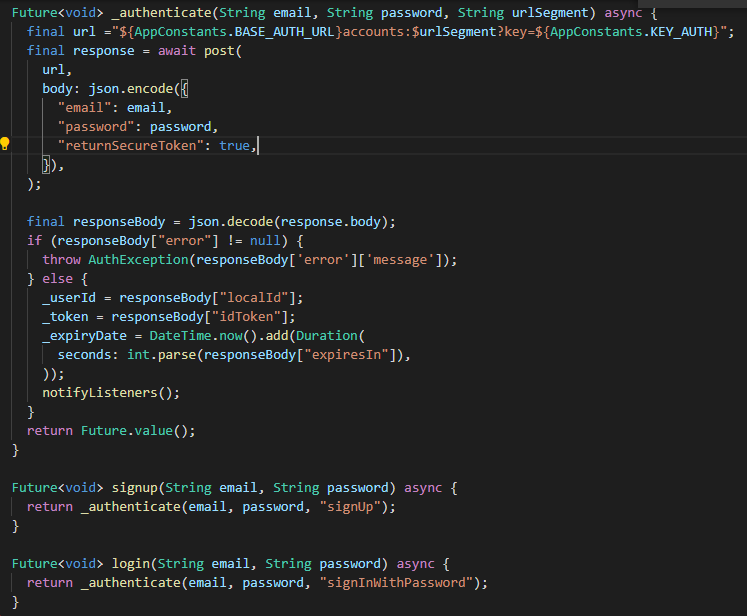


Aqui podemos encontrá-la.

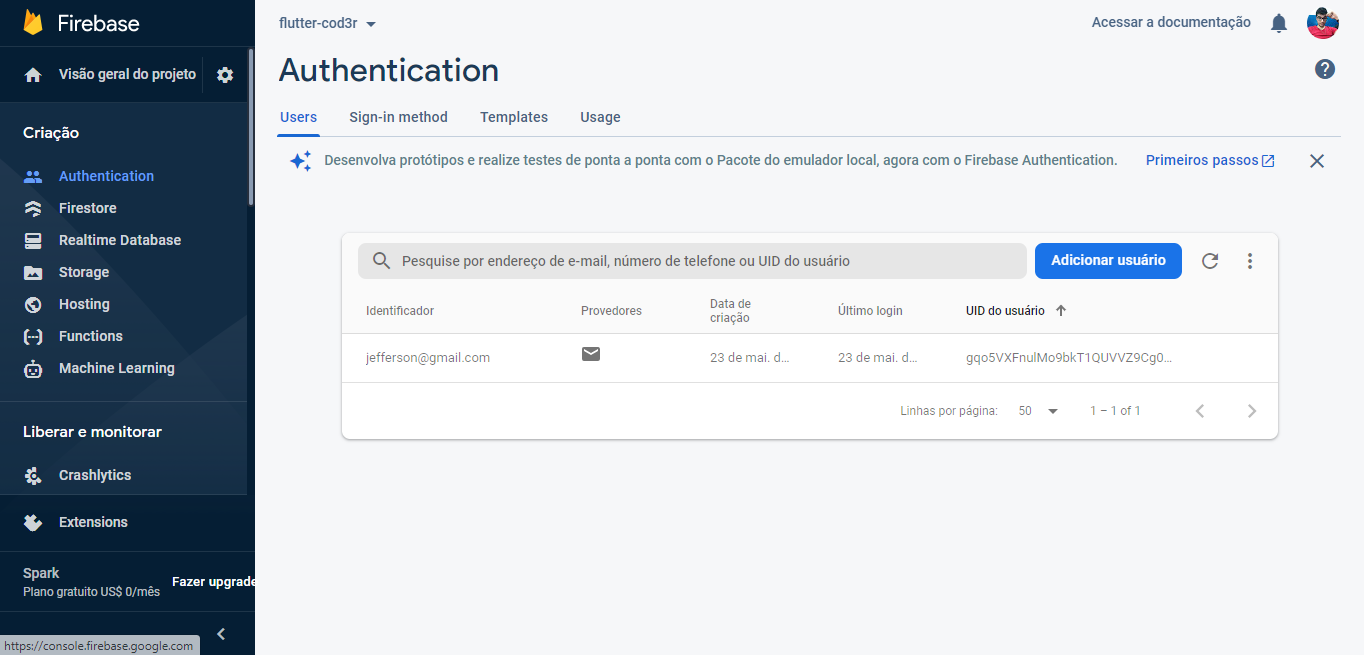


Dessa forma a url ficara como “https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=KEY\_FIREBASE”

Exemplo Utilização:



Aqui podemos ver os usuários adicionados.



Agora com o WebService utilizando autenticação, é preciso enviar um token valido em todas chamavas, como o exemplo abaixo.



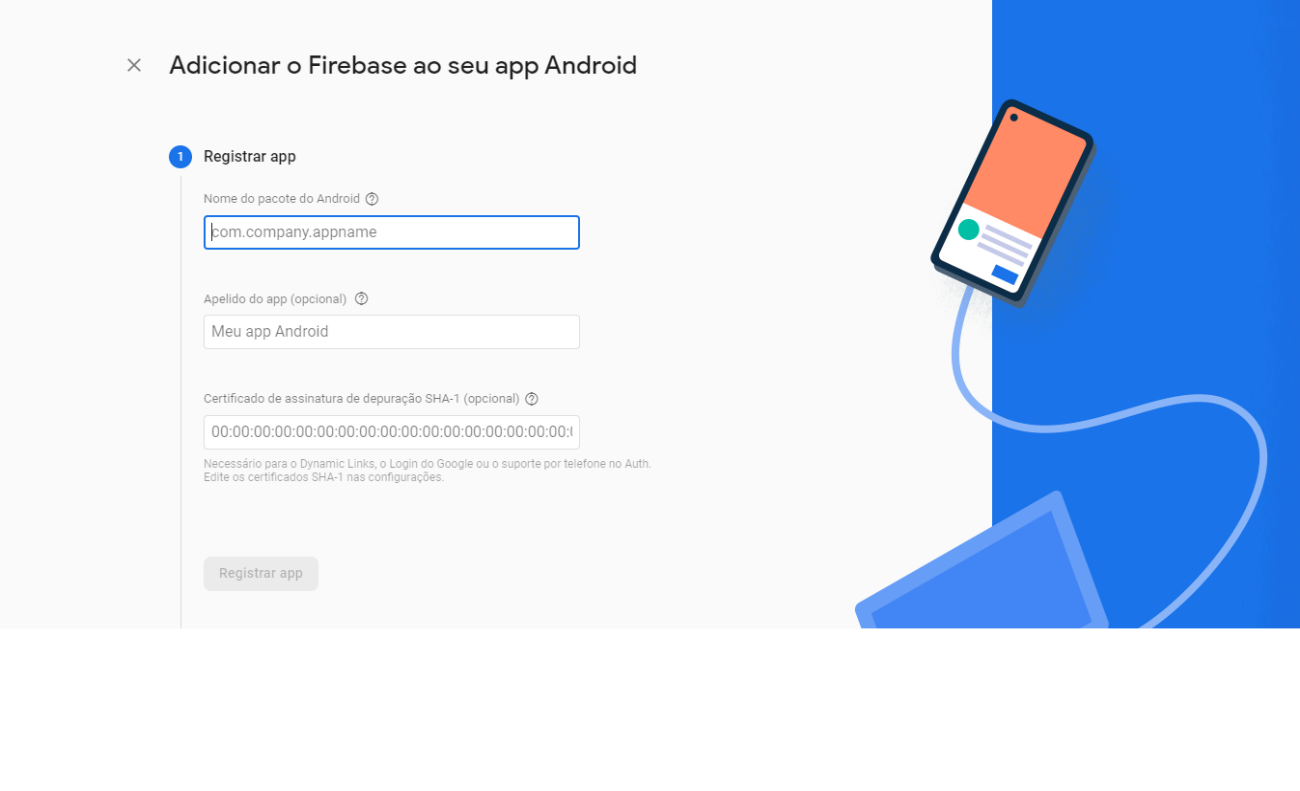
**Adicionando o Firebase ao seu APP**

**Android:**

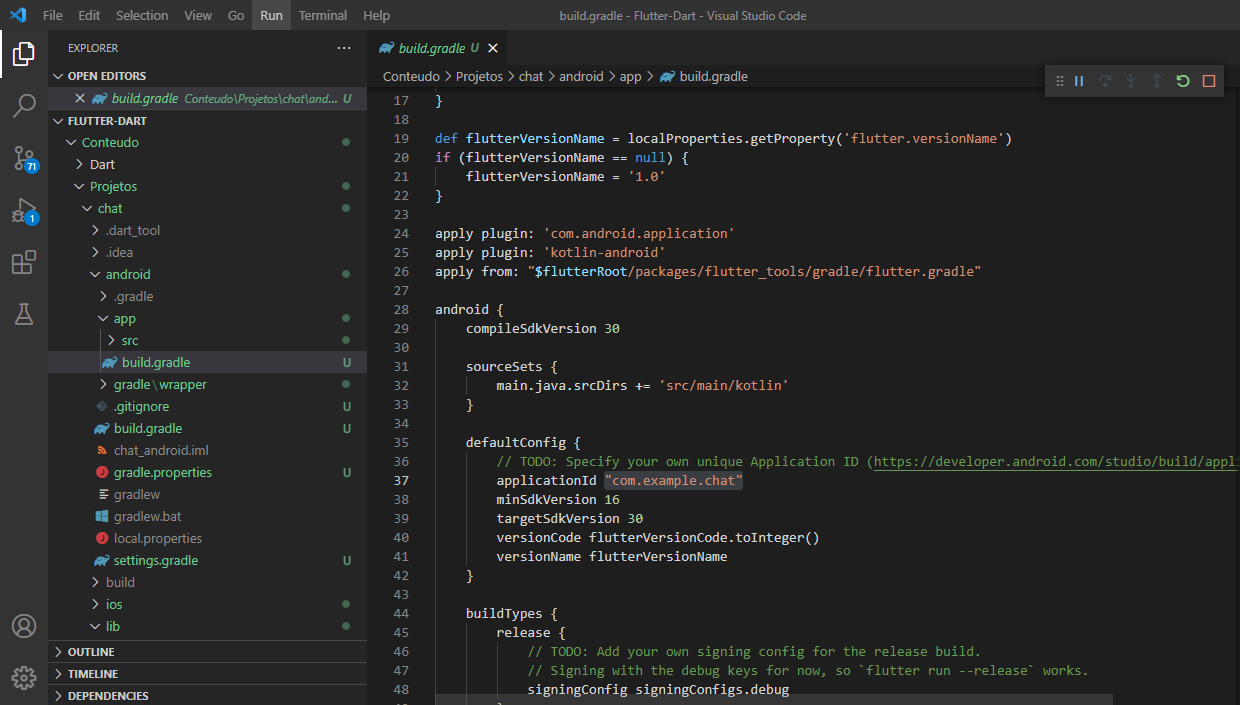
Na página Home clique no ícone do android.



Você deverá preencher algumas informações do seu app.



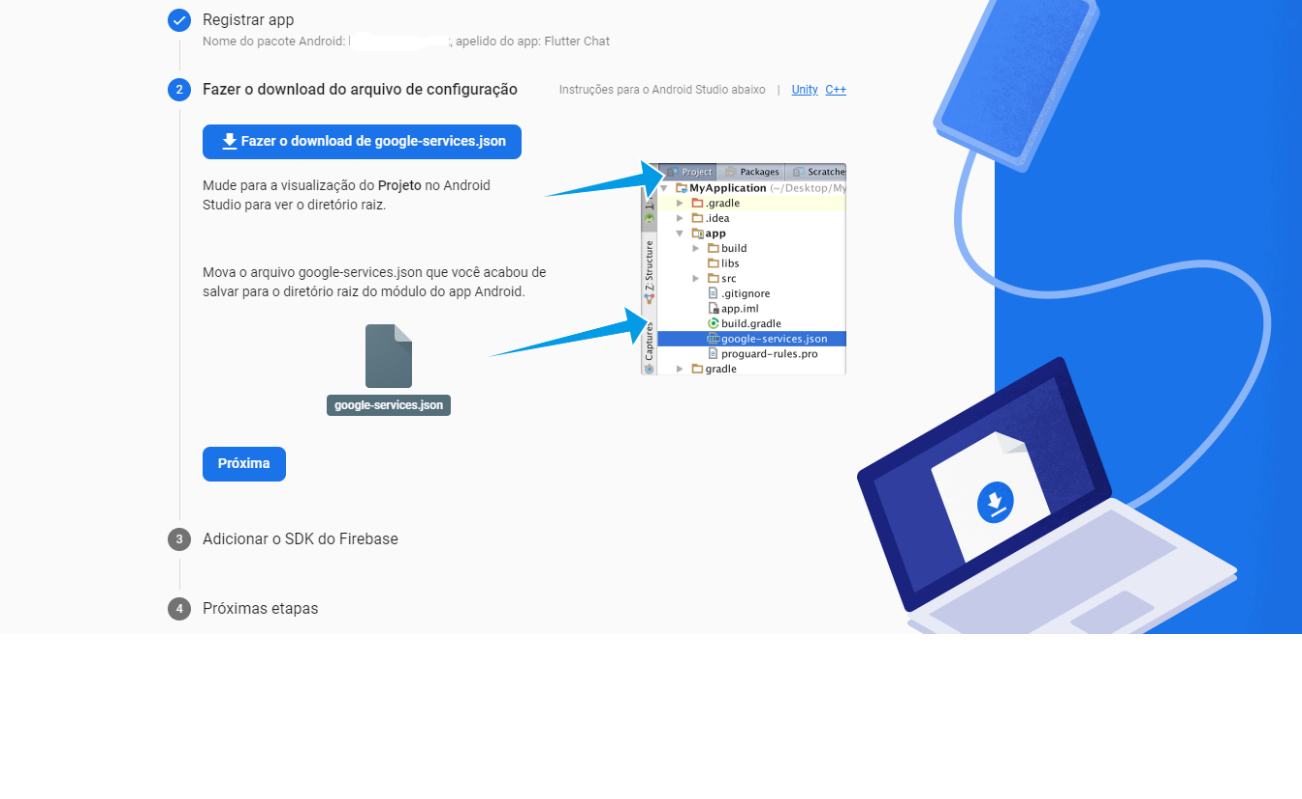
Nome do pacote do Android - Localizado no build.gradle do seu projeto(Antes de você vincular ao firebase. Esse código único pode ser alterado)



Apelido do app (opcional) – Campo opcional que pode ser definido a sua escolha

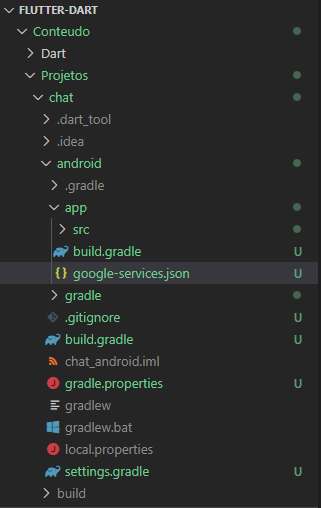
Em seguida clique em registrar APP

Após isso será gerado um arquivo que deverá ser baixado.



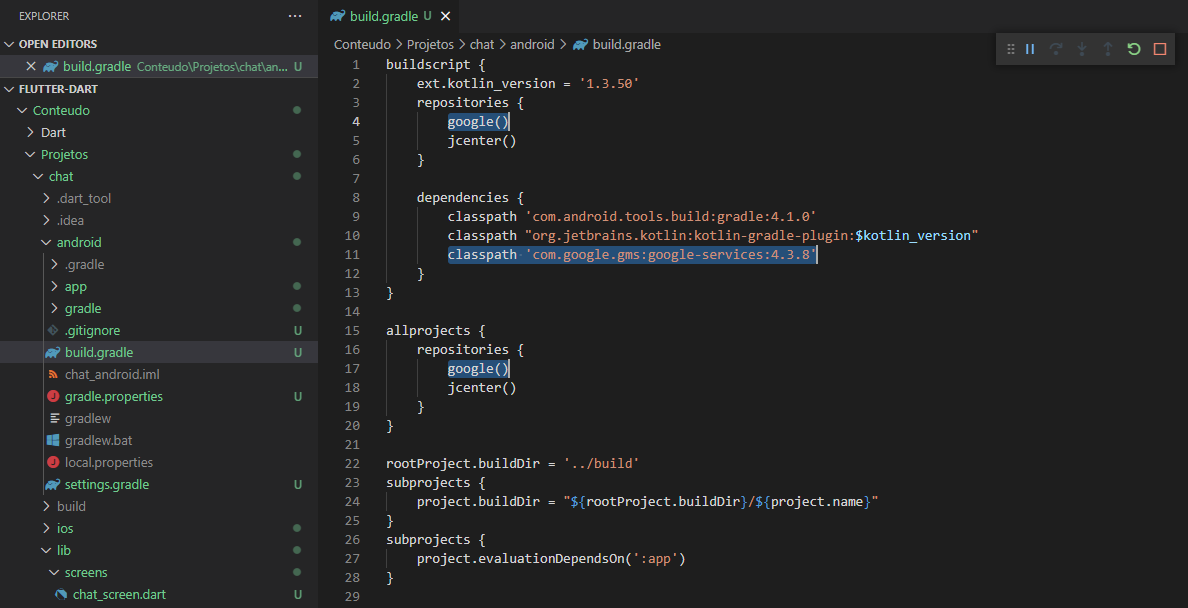
Baixe o e adicione no seu projeto.

Adicione dentro da pasta App.



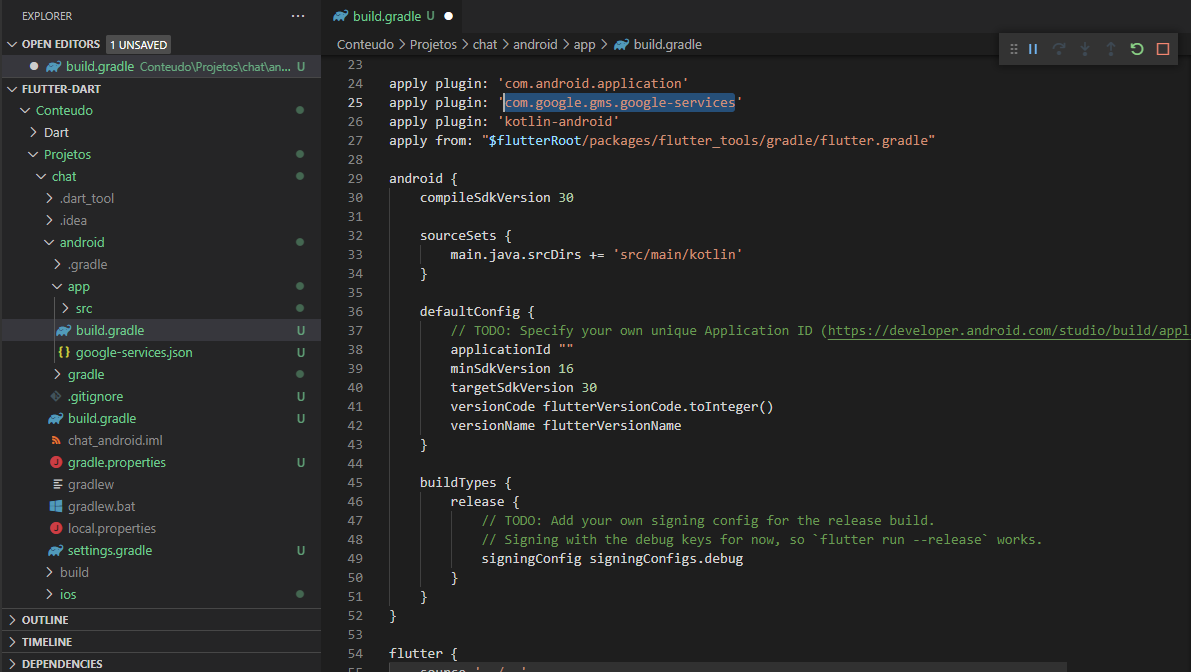
Em seguida clique em próxima.

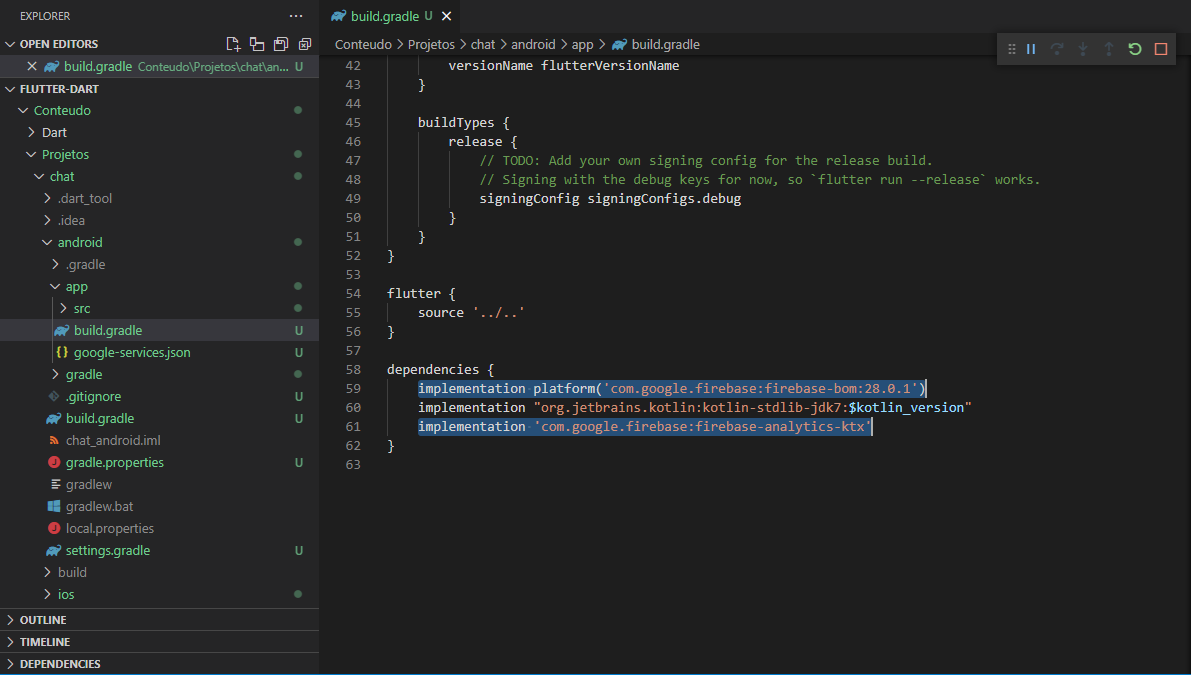
Como mostra no passo a passo, deve ser adicionado as seguintes linhas no arquivo build.gradle da pasta android, cuidado, tem mais de um build.gradle no projeto, se atente a isso.



No próximo passo selecionei a opção Kotlin no firebase

Assim como mostrado no passo a passo adicionei as seguintes referências no build.gradle dentro da pasta app.

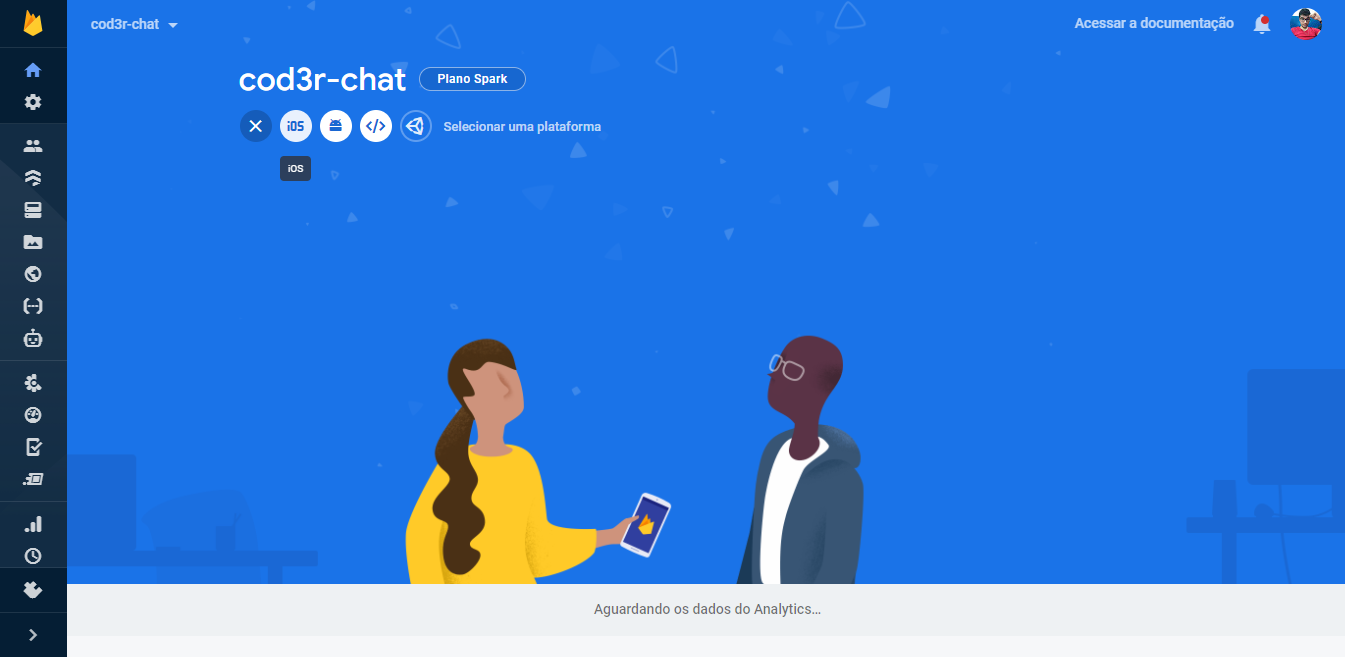




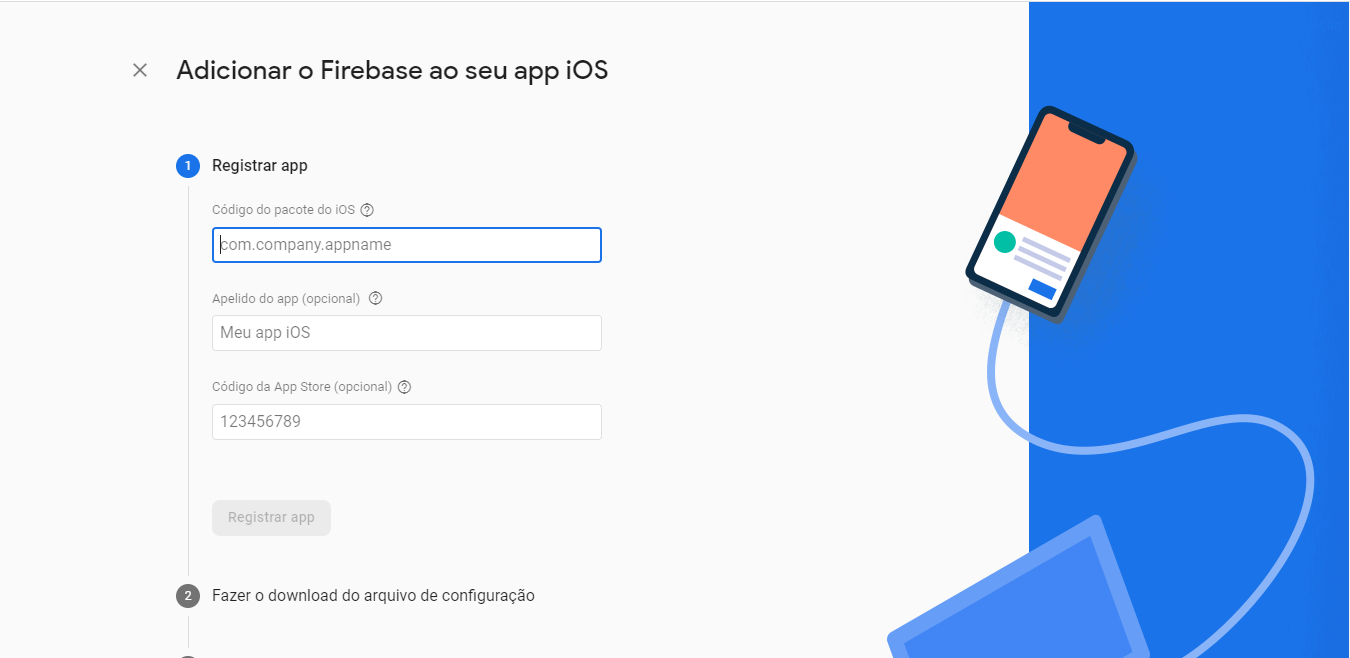
Agora clique em próxima de novo, e em seguida ir para console.

**IOS:**

Agora clique em Adicionar APP, e no símbolo do IOS.



Como pode ser visto, ele pede os mesmos dados que no Android

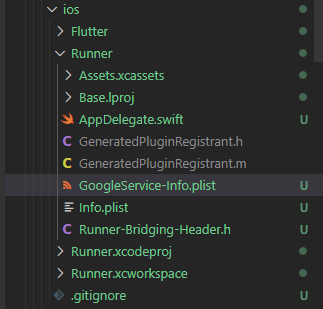


Irei pegar o código do pacote abrindo o projeto IOS com o XCode.

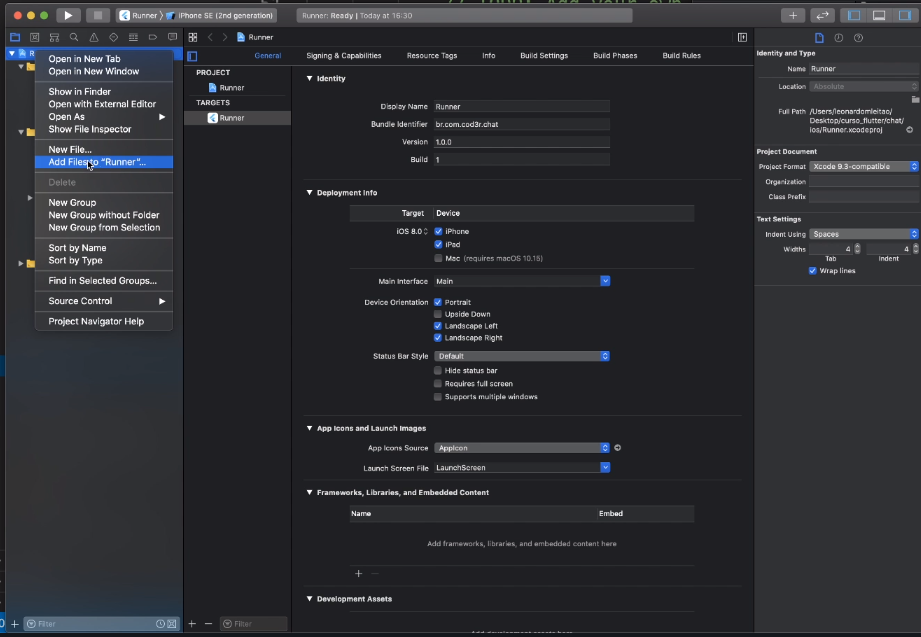




Após preencher os dados, faça donwload do arquivo gerado. E coloque na pasta Runner.



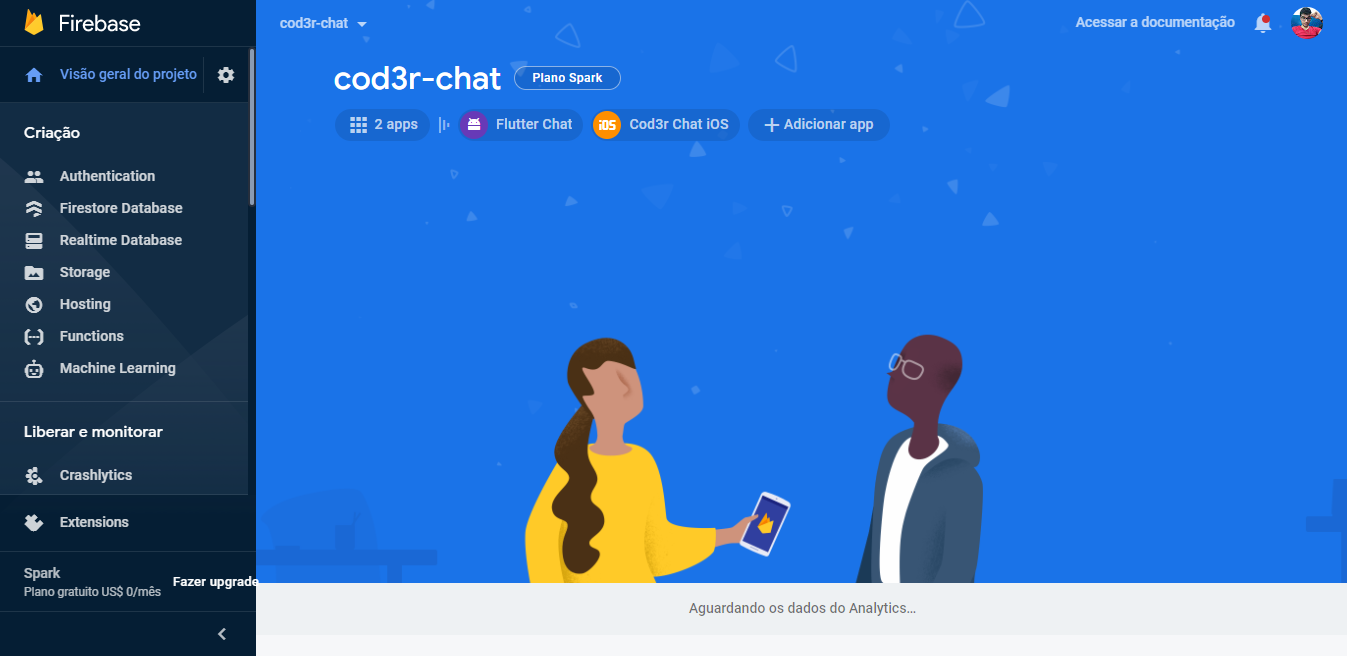
Após jogar o arquivo na pasta, é preciso adicioná-lo ao projeto pelo XCode.



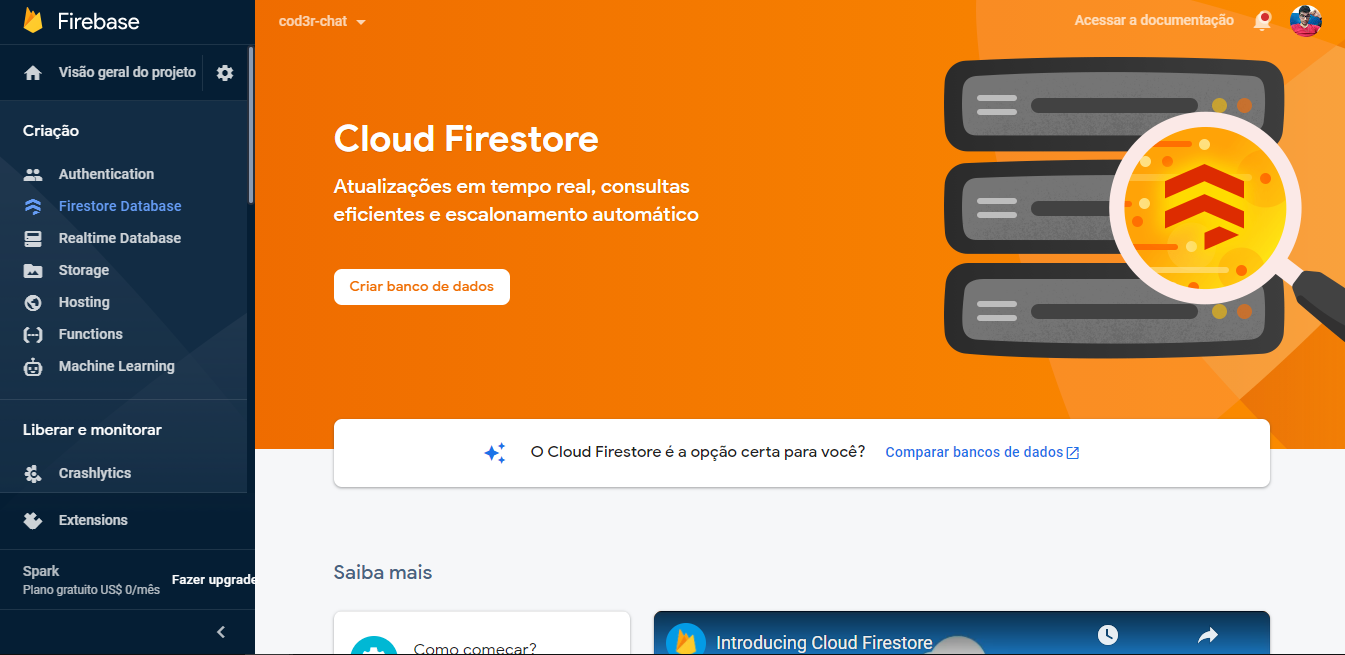
Agora irei pular as próximas etapas e voltar para o console.

**Utilizando Firestore Database**

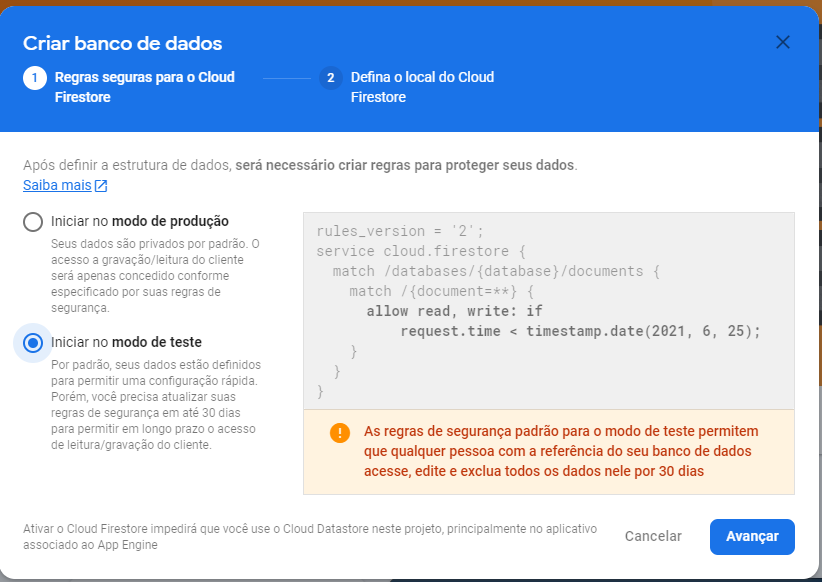
No menu lateral clique em Firestore.



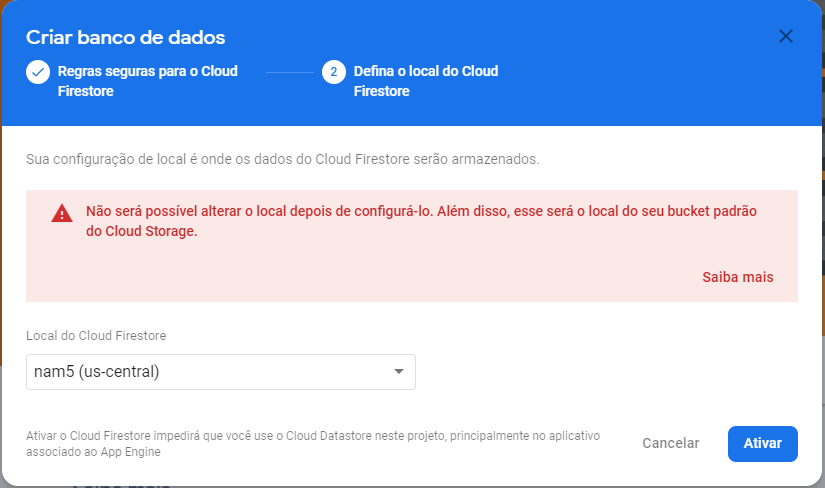
Agora clique em criar um novo banco de dados.



Irei selecionar modo de teste.

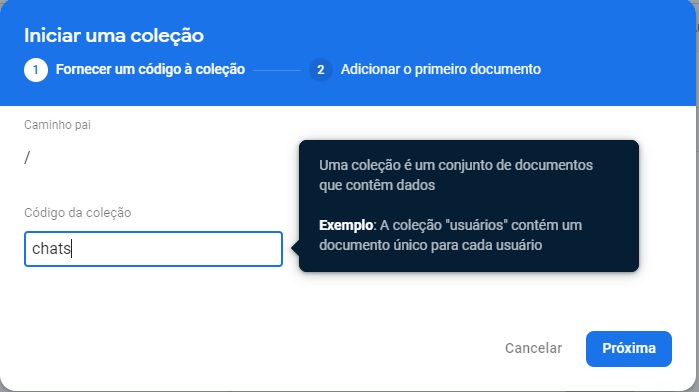


Selecione o local que será armazenado.

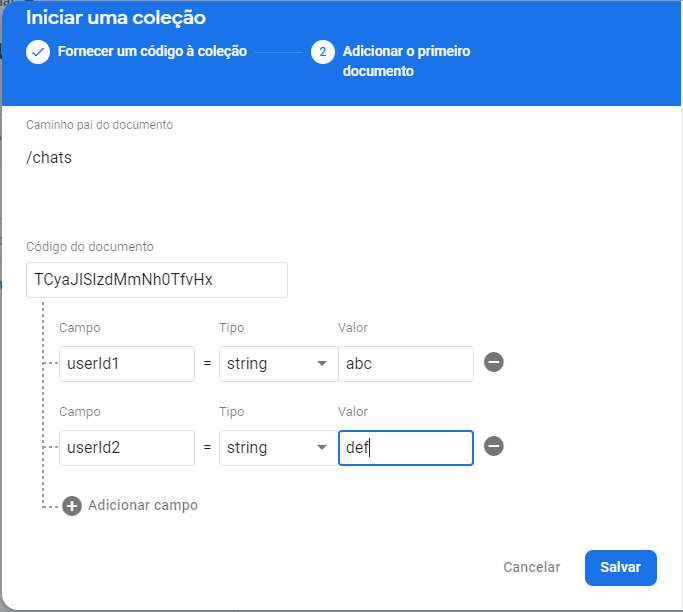


Em seguida clique em ativar.

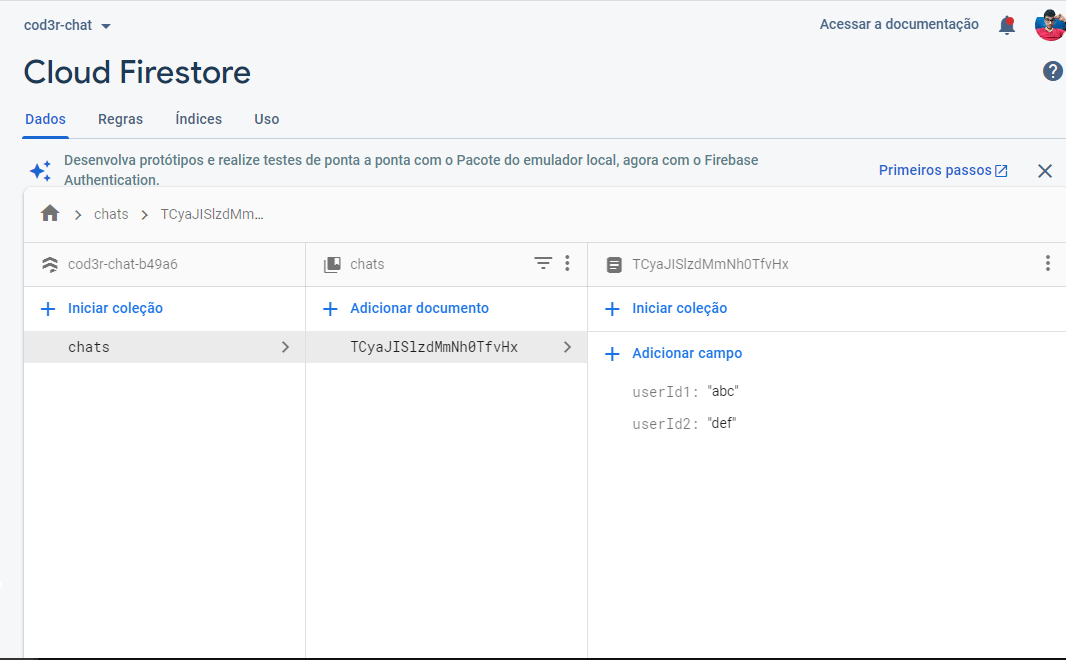
Agora cliquem em iniciar coleção, e coloque o nome desejado.



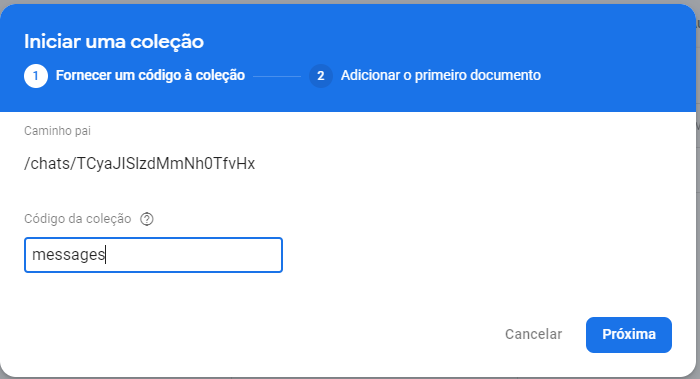
Na próxima etapa, caso deseje, é possível adicionar o primeiro documento

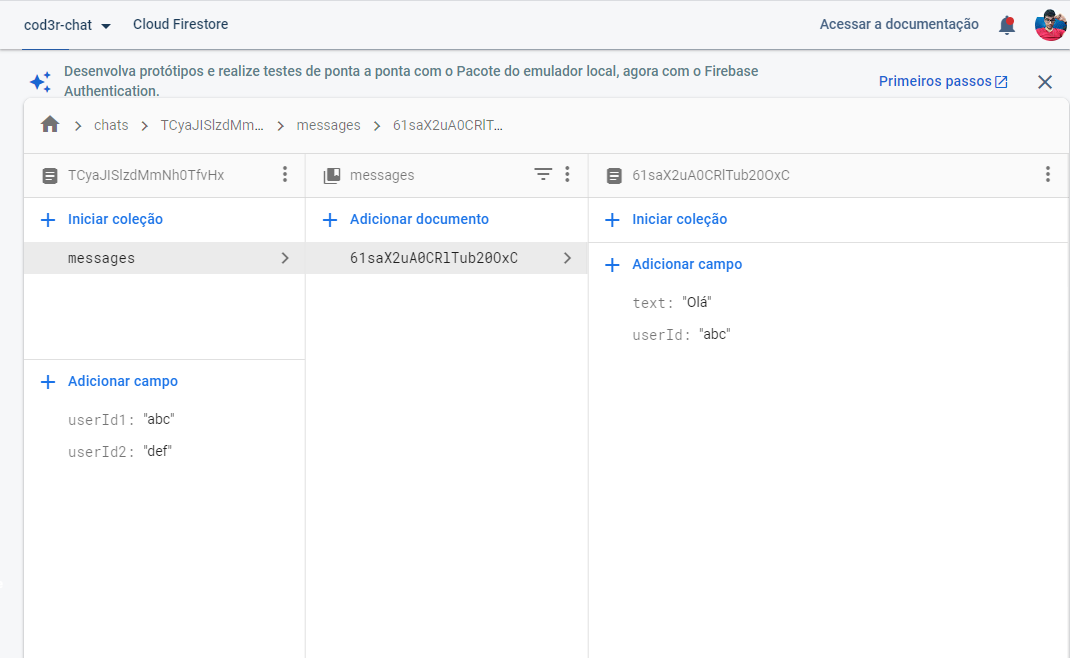


Dessa forma, foi criado uma coleção contendo um documento com seus respectivos dados.

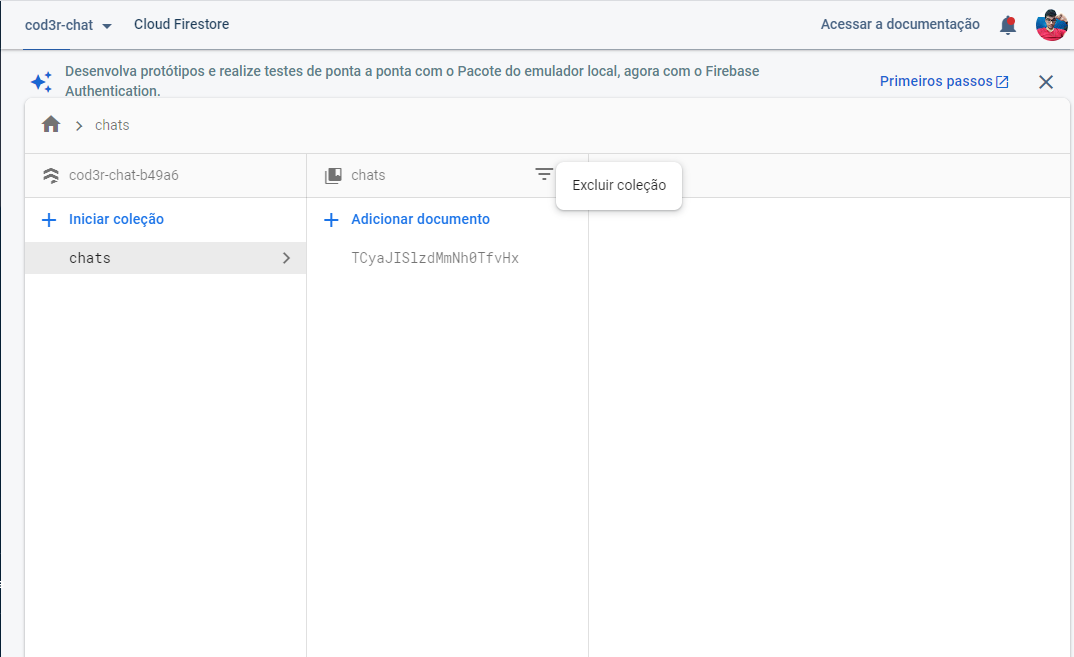


E dentro desse documento, é possível adicionar uma nova coleção de documentos.

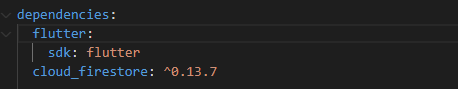




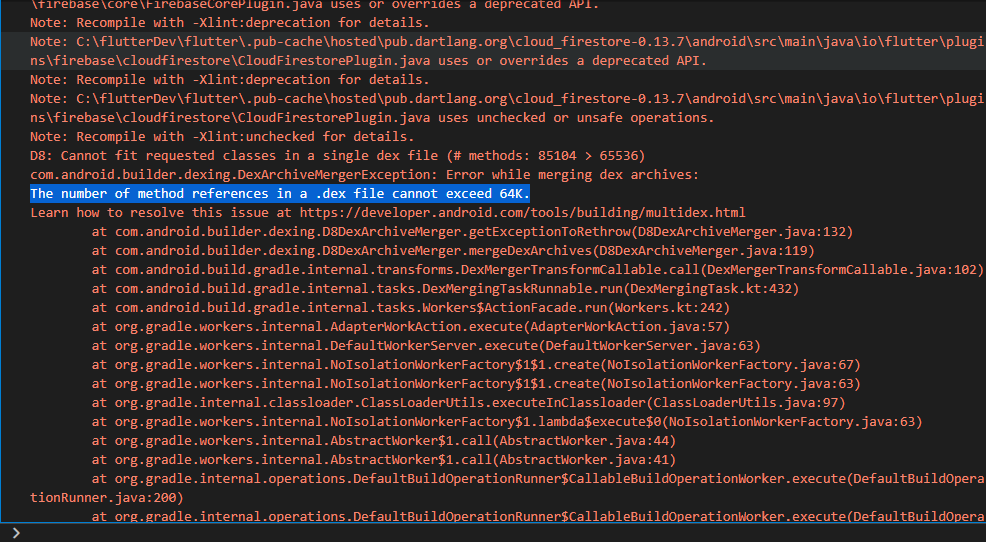
Agora voltarei para chats, e excluirei as coleções.



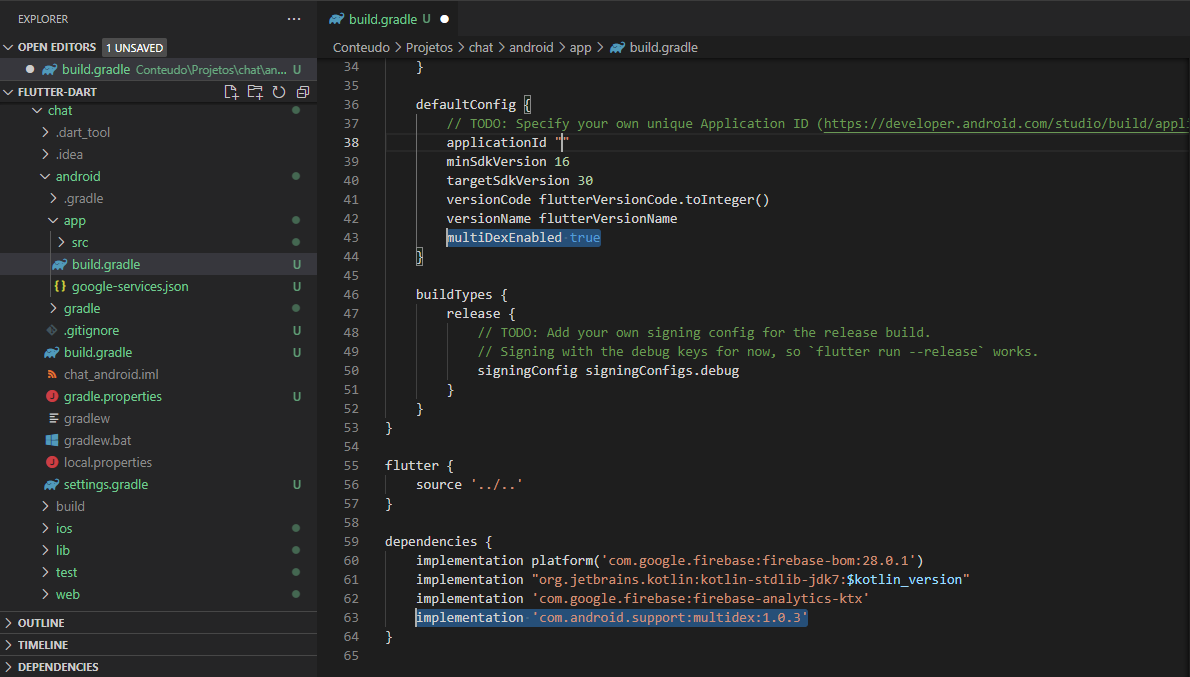
Em seguida instalei as dependências no meu projeto para utilizar o firestore.



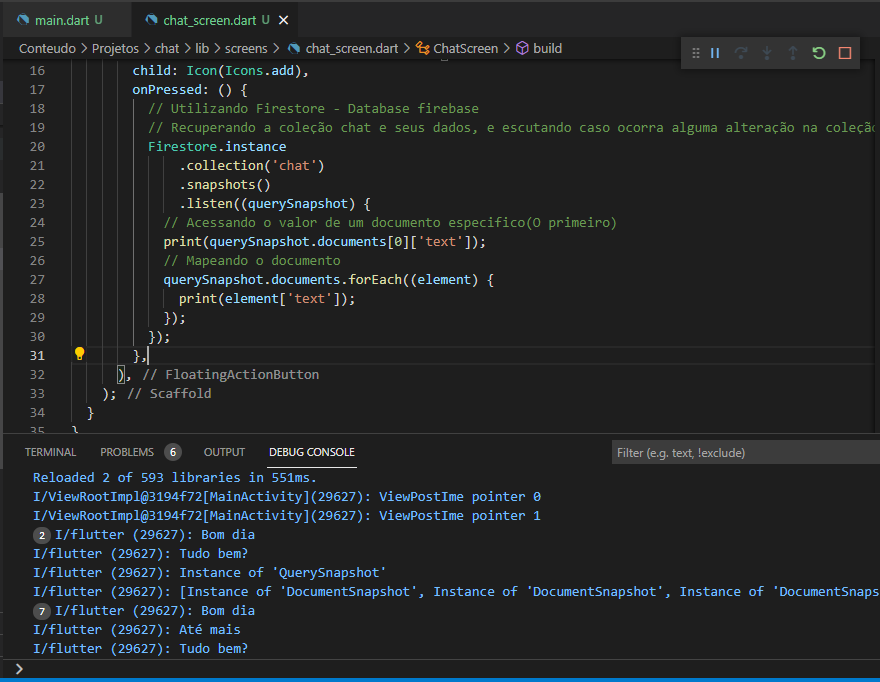
Ao buildar em Flutter, caso ocorra o seguinte erro.



Adicione no arquivo build.gradle da pasta app as seguintes referências.

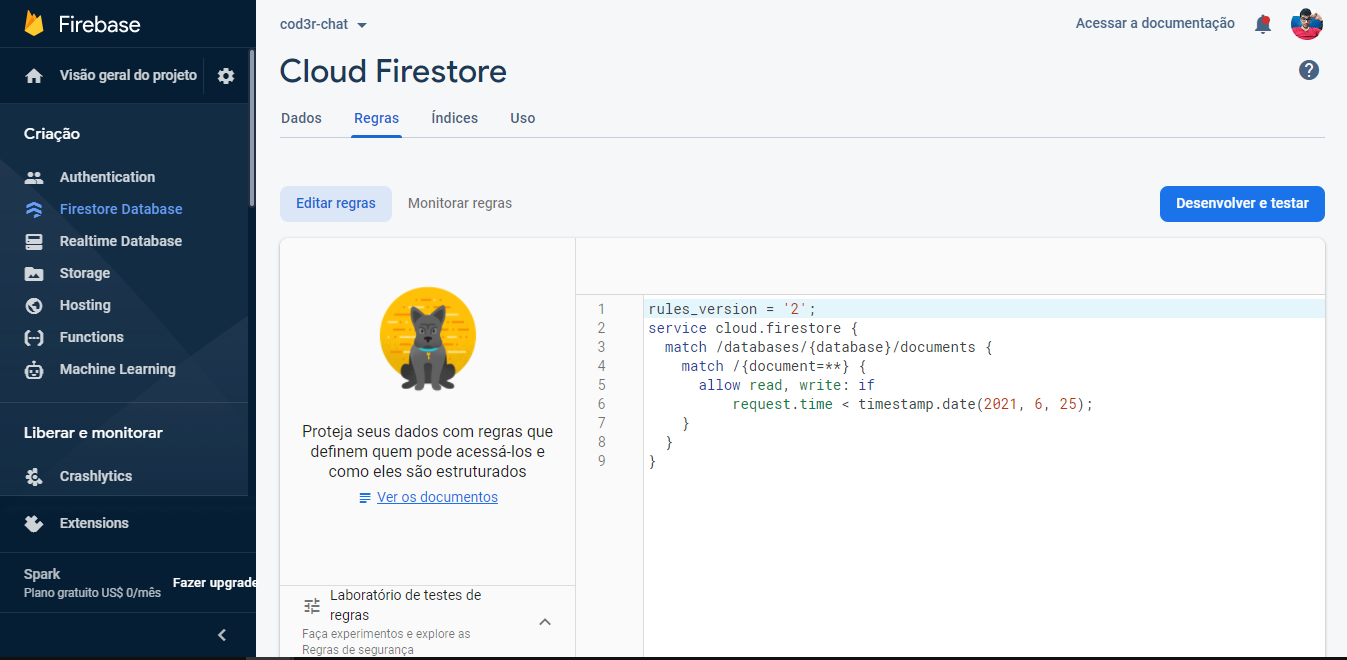


Exemplo utilização

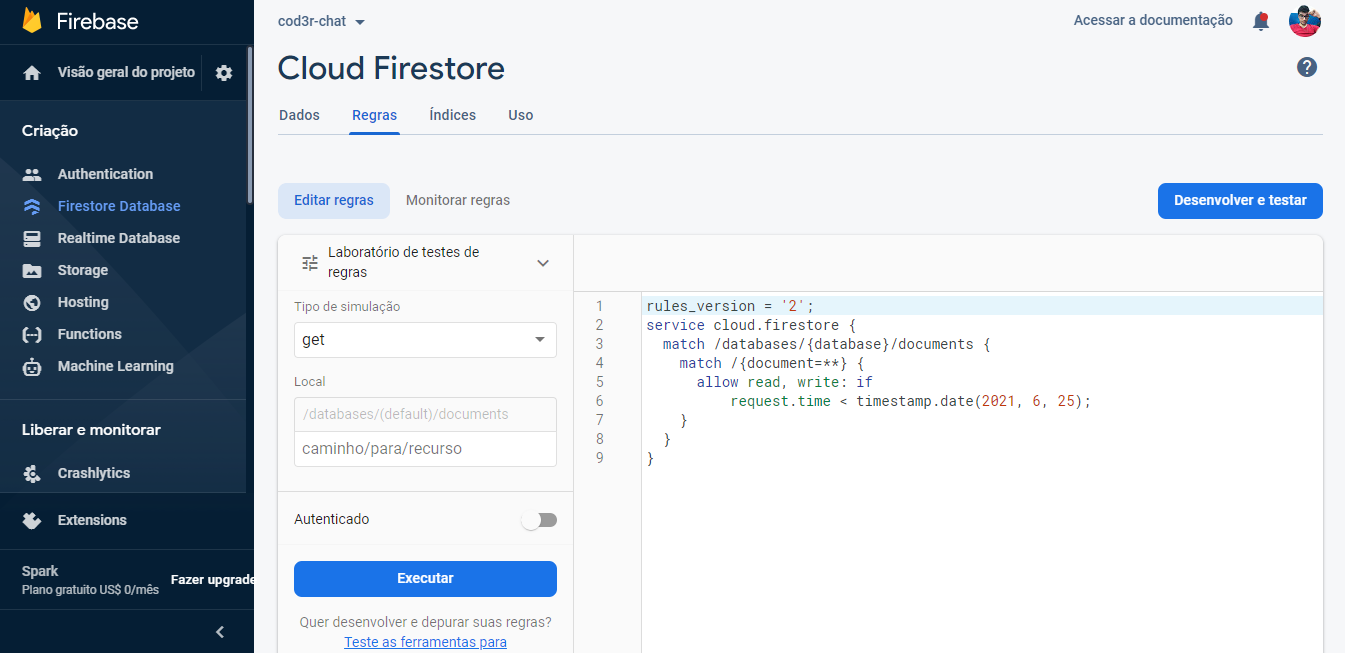


-- Adicionando regras de restrição de acesso ao firestore

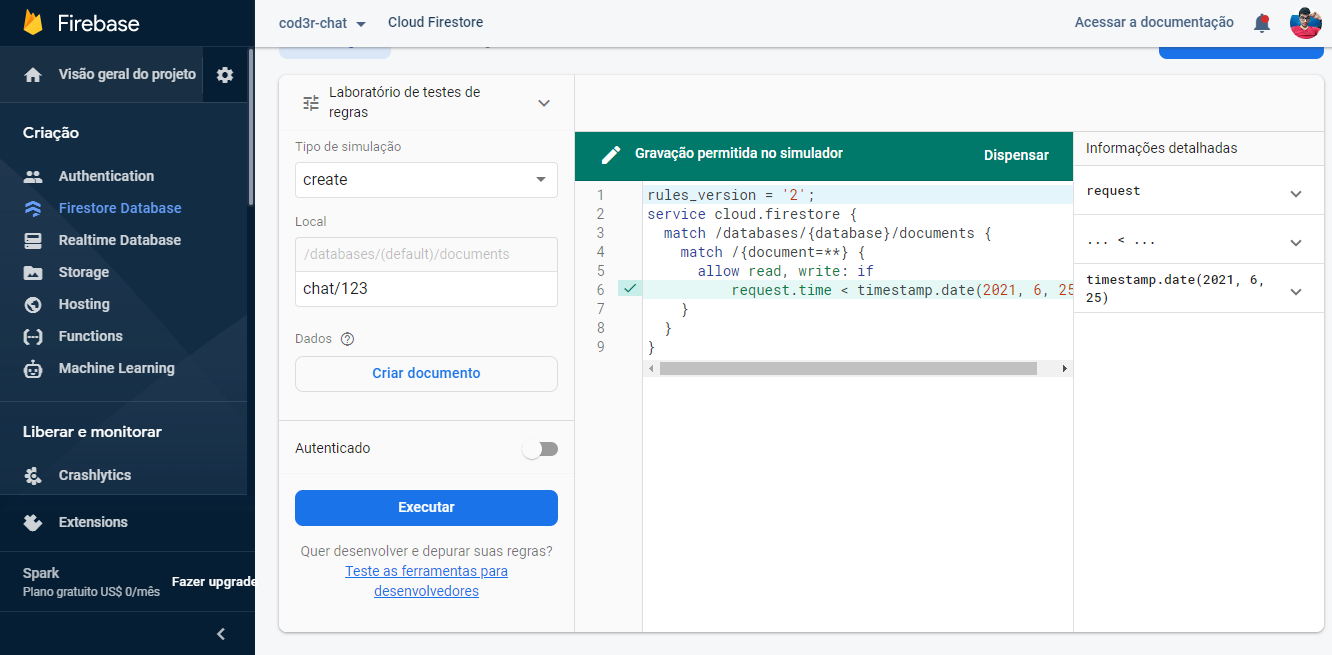
Acesse a aba regras.



E clique em laboratório de teste para efetuar simulações (Get, Create etc...) e testar as regras criadas

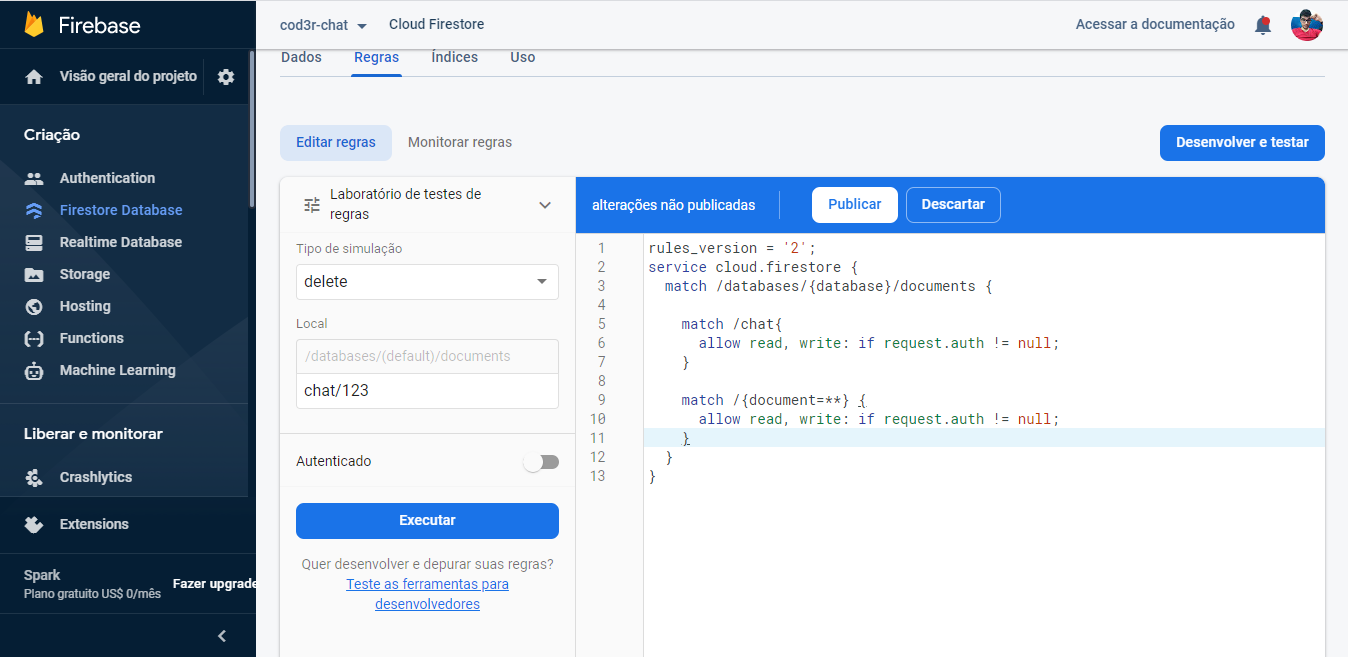


Como pode ser visto abaixo, no momento não existe nenhuma regra, então consegui adicionar dados facilmente, mesmo sem estar utilizando autenticação.



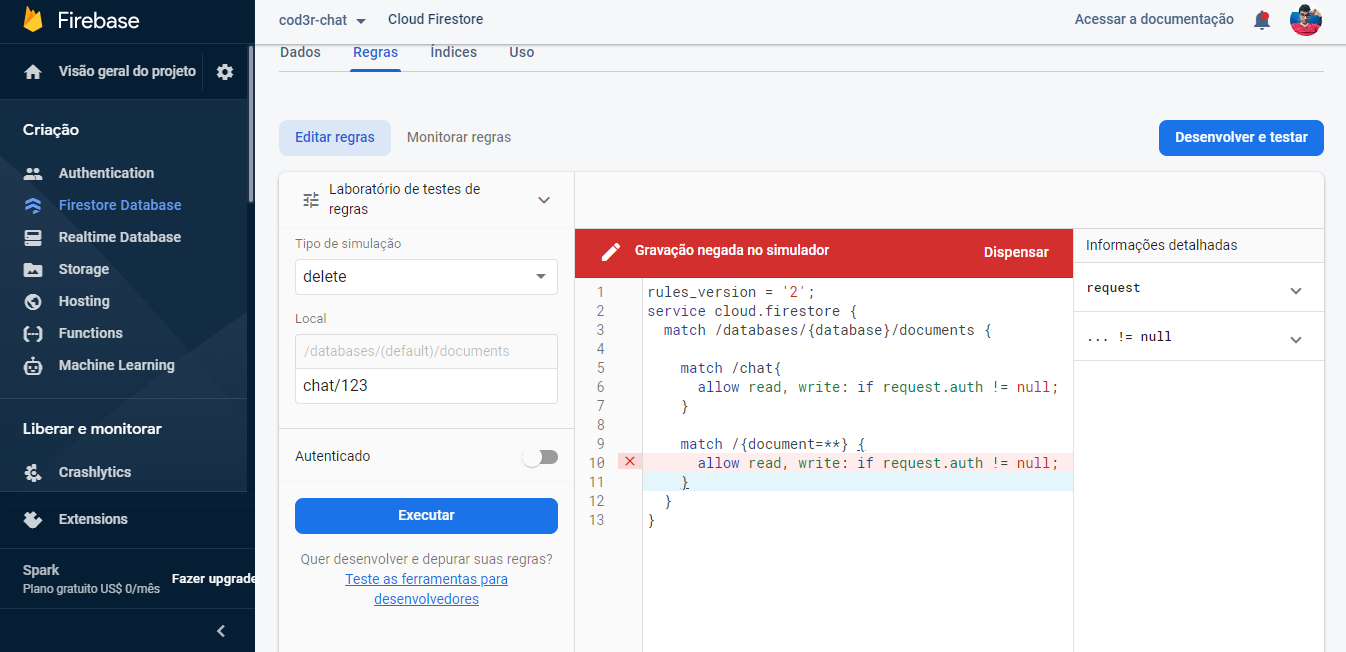
Agora adicionarei uma regra, que para efetuar alteração e leitura nas coleções é preciso estar autenticado.

A primeira restrição esta apenas para coleção chat, e a segunda englobando todas.

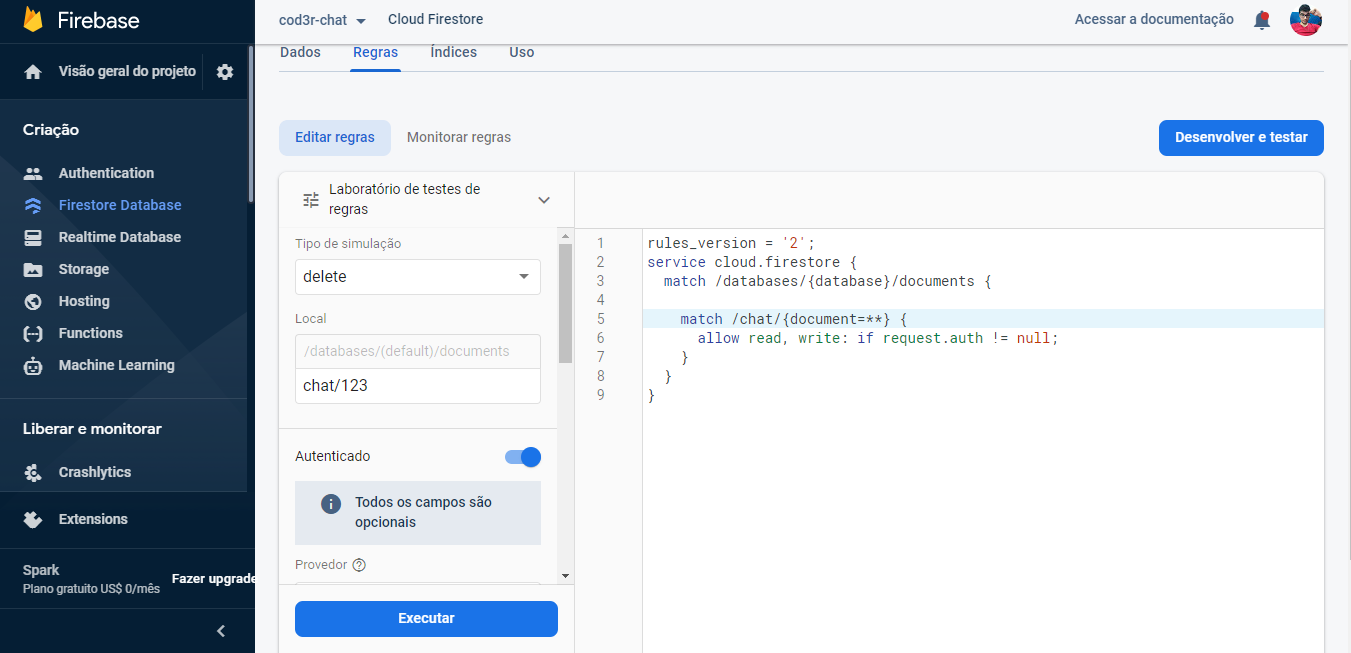


É preciso clicar em publicar para surtir efeito.

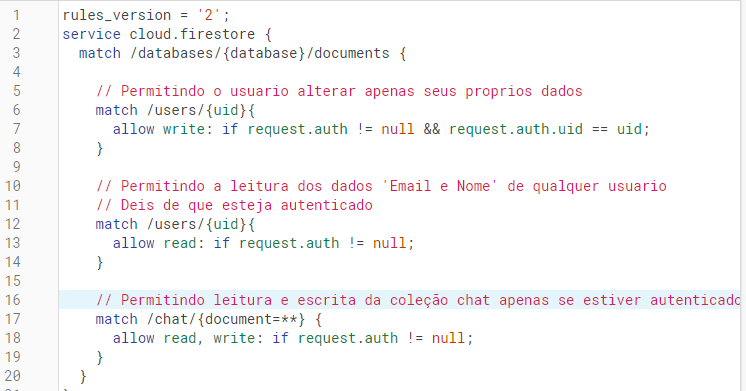
Agora caso efetue alguma leitura e escrita sem estar autenticado é exibido a seguinte mensagem como esperado.



Agora aplicando apenas para tudo que estiver dentro da coleção chat

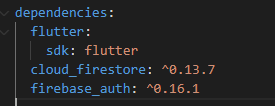


Dessa forma deixarei apenas essas regras como teste



**Autenticação (Integrado por Biblioteca)**

Primeiramente instalei a biblioteca firebase\_auth, agora em vez de efetuarmos a autenticação via APi como alguns passos anteriores, utilizaremos de uma forma mais integrada a flutter.



\* A configuração para utilização é a mesma da forma via API, podendo escolher a forma de autenticação desejada.

Exemplo utilização:

