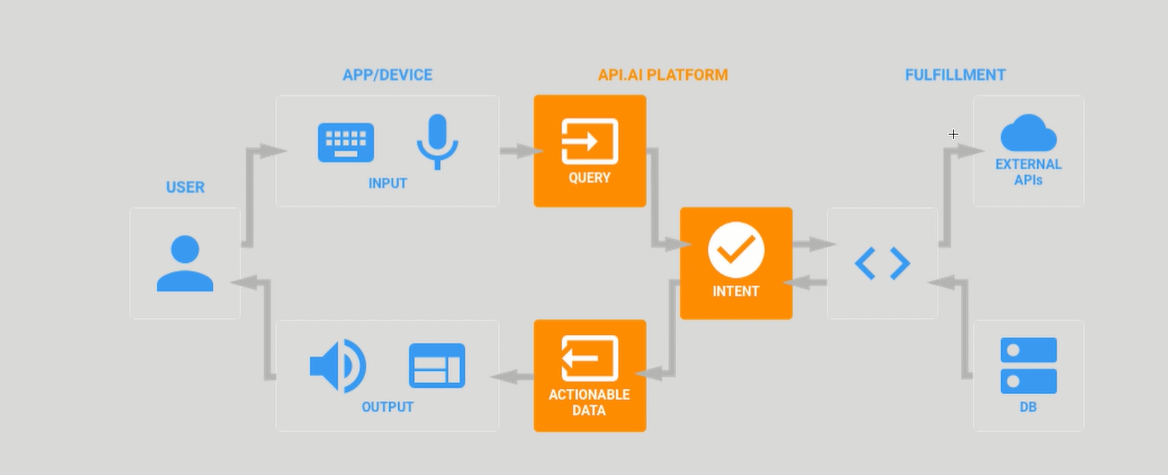
**Plano de aula 06 – WebHook**

Nós trabalhamos até então, com um usuário fazendo uma consulta ao agente. O agente analisava uma entrada de texto ou voz e respondia de volta ao usuário via texto ou via voz. Mais potencialmente nós podemos conectar essa intente a um WebHook onde o código será executado no nosso servidor para retornar a resposta usuário.



Vamos criar uma intente para consultar preço de curso, ainda usando uma resposta padrão, sem usar Webhook.

1. Crie uma entidade chamada curso
2. Entre com os seguintes cursos:
   1. Python
   2. R
   3. Machine Learning 🡪 ML
   4. Hadoop
3. Crie uma intente chamada preco.curso
4. Entre com as seguintes frases para treinamento
   1. Qual o preço do curso de Machine Learning?
   2. Qual o valor do curso de R?
   3. Quanto custa o curso @curso?
5. Resposta: 100 reais.
6. Salve e teste

A partir desse ponto vamos trabalhar com WebHook para retornar uma resposta do servidor.

O código que será executado em um backend pode estar em um servidor da empresa, pode estar em uma plataforma de cloud computing, ele pode estar em uma máquina com o sistema operacional windows e linguagem C#, ele pode estar em uma máquina que usa linux com php e assim por diante.

1. Vá até a opção fulfiment e habilite a opção webhook.
2. Observe a Figura 1.

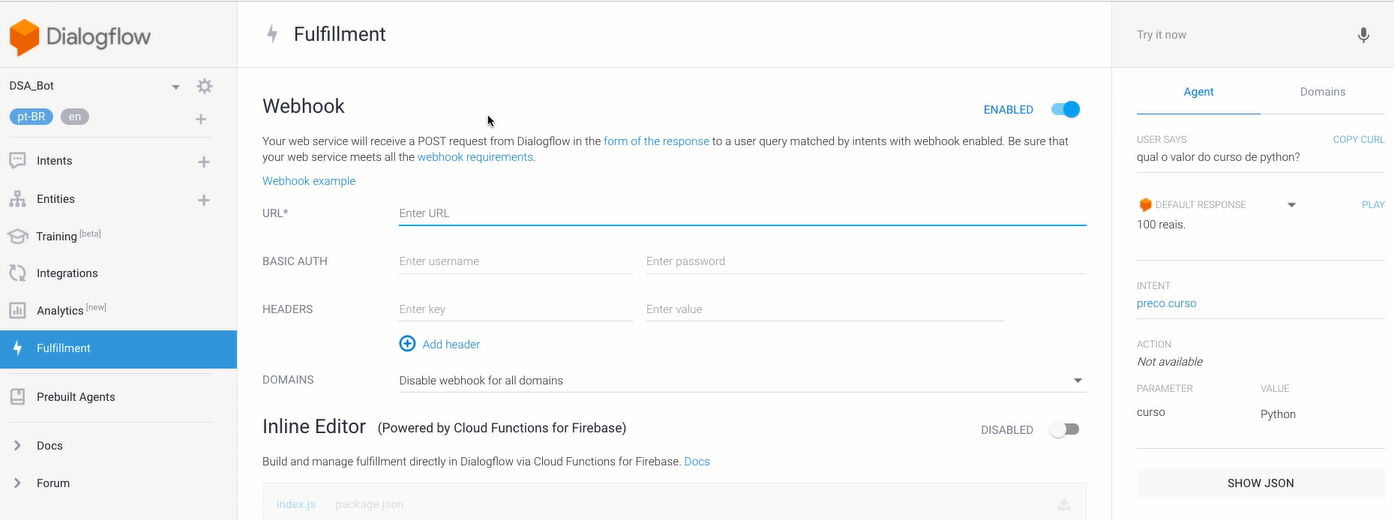


Figura 1

Não é obrigado você trabalhar com WebHook em todas as intentes. Você pode configurar individualmente cada intente que usará WebHook.

Observe que você pode informar uma URL, um login e uma senha para seu agente.

1. Habilite Inline Editor e observe que ele fornece dois arquivos index.js e package.json. veja a Figura 2.



Figura 2

A qualquer momento você pode fazer download do arquivo, clicando na seta do lado direto superior.

1. Faça o deploy do index.js clicando no botão DEPLY, como mostra a Figura 3.

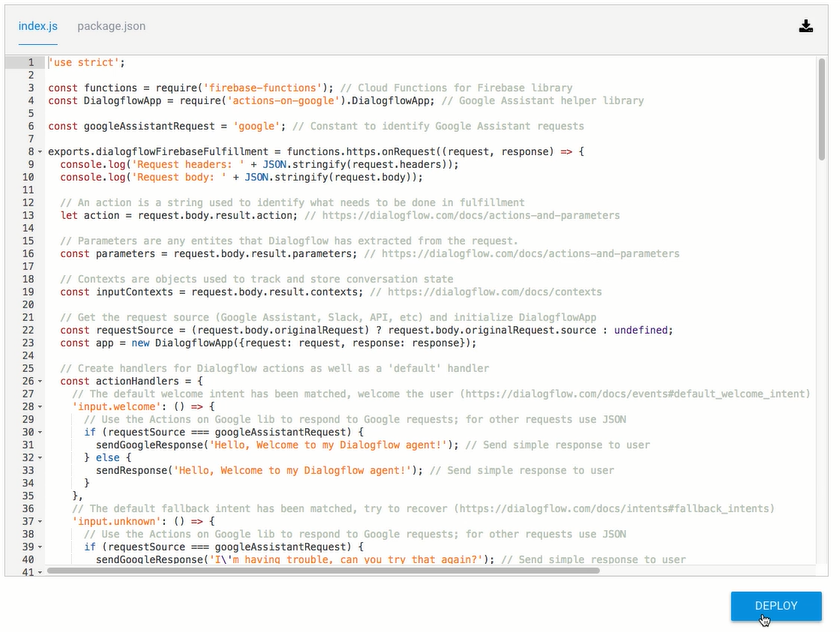


Figura 3.

Isso envia o código para o serviço gerenciado Google Cloud Function que irá executar esse código quando Dialogflow enviar uma requisição a ele.

Agora vamos configurar uma intente para utilizar WebHook.

1. Quando fizemos o deploy na seção anterior, nós conectamos nosso código index.js ao Google Cloud Functions.
2. Abra a intente que você quer usar WebHook e habilite o seu uso, como mostra a Figura 4.

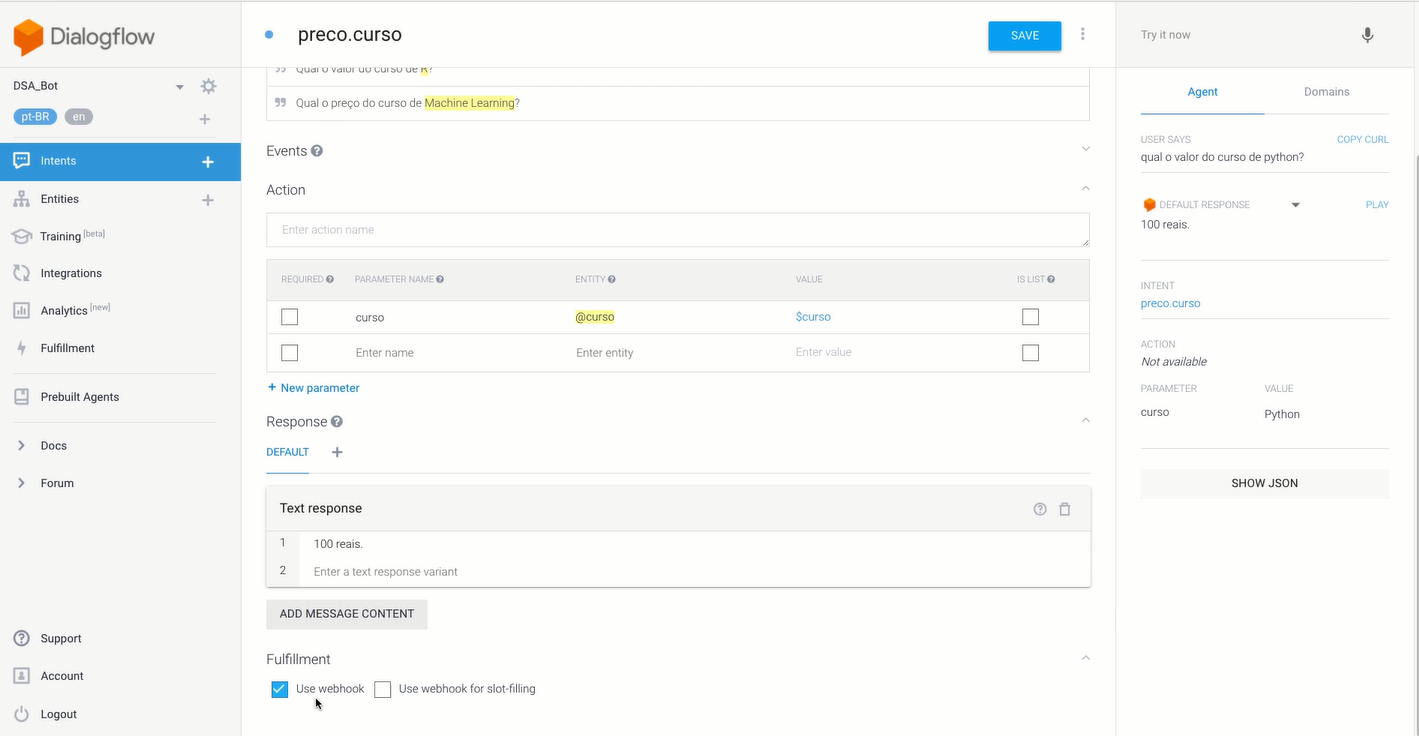
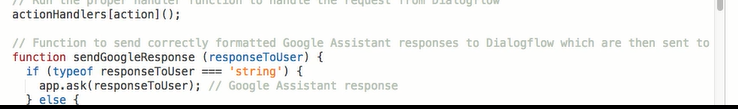


Figura 4. Habilitando a intenção para usar WebHook.

1. Sete uma action na intente preco.curso com a string “**preco.curso**”. Não é obrigado ser o mesmo nome da intente pode ser outro. Uma action é uma string que será passada para o servidor.
2. Acessando a action no código. Para isso altere o código fornecido no WebHook para o mostrado na Figura 6.



Figura 6





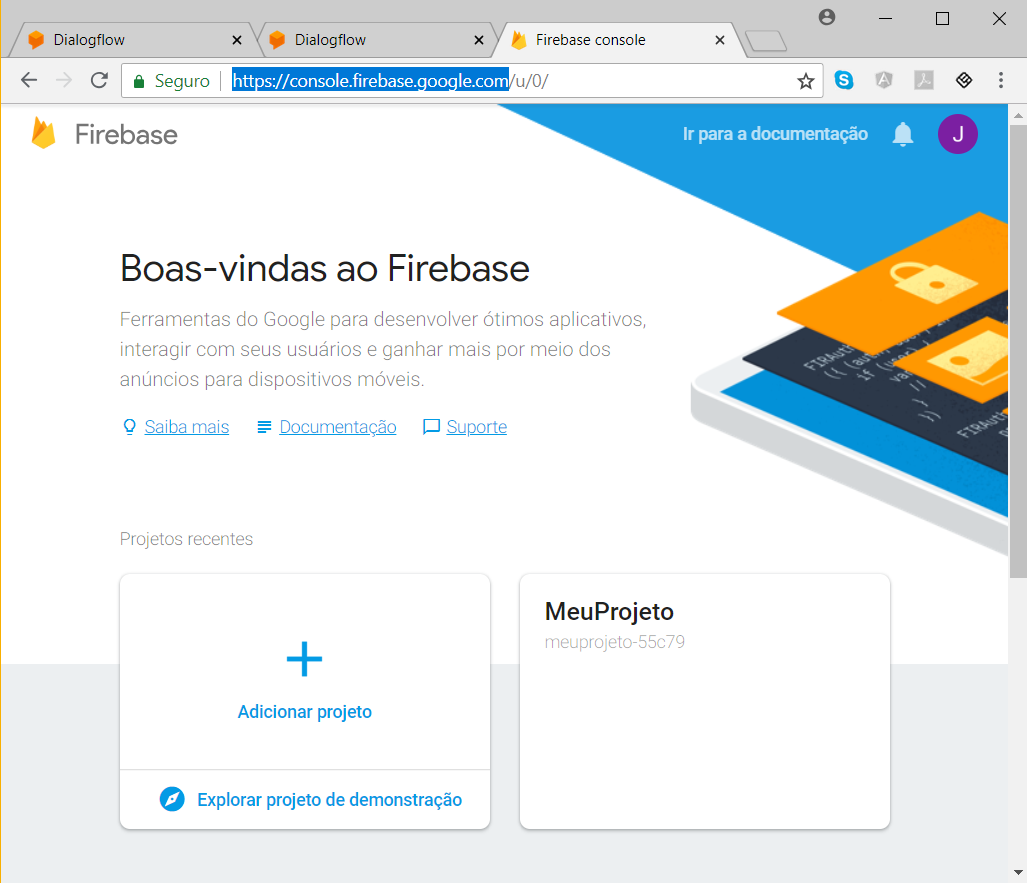


1. Faça o deploy para confirmar essas alterações. Para isso, clique novamente no botão DEPLOY.
2. Salve e teste. Observe que a resposta agora é “O valor do curso é de 200 reais” que vem do backend.
3. E a resposta padrão que era “100 reais”, não mais vai poder ser assim, já que a resposta vem do backend. Então, podemos alterá-la para uma resposta que indique se houve algum problema com o backend. Por exemplo, “Estamos com problema, tente novamente mais tarde.”

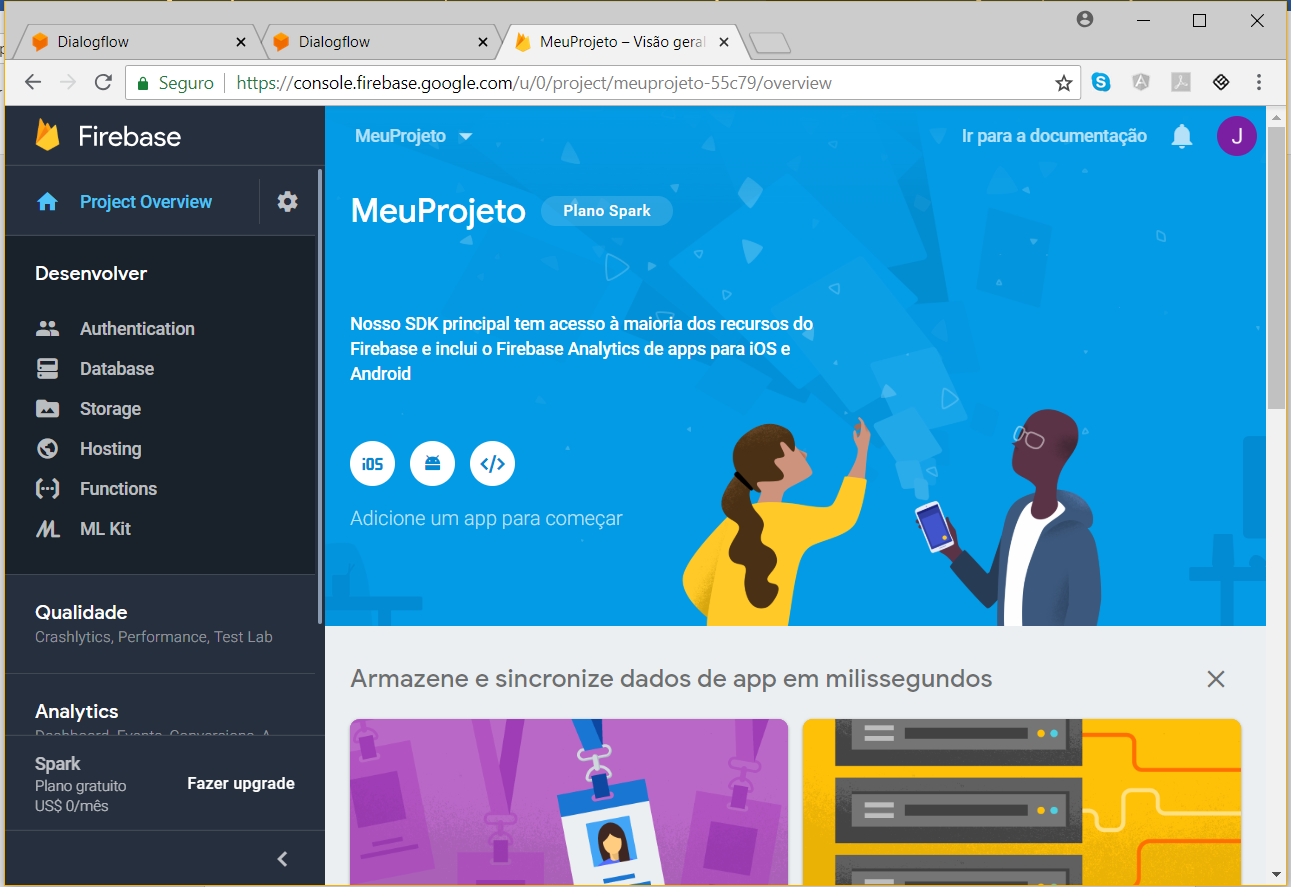
**A plataforma Firebase**

Esse a plataforma Firebase, usando a seguinte URL: <https://console.firebase.google.com>

Você será direcionado para a seguinte tela

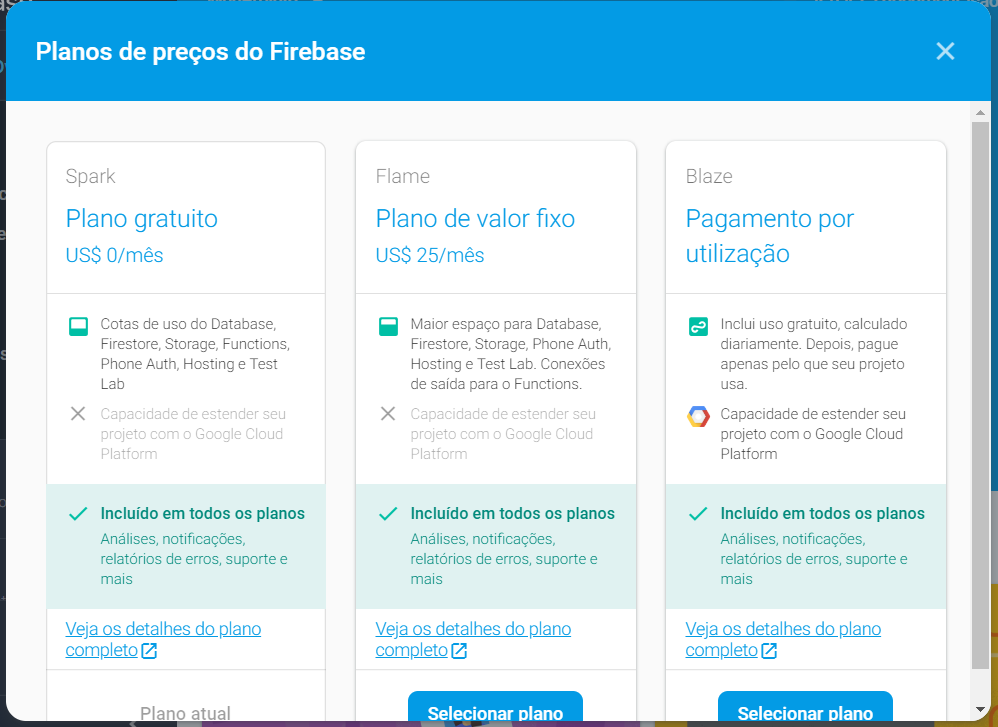


Se você já estiver conectado ao Google, com o login que você está desenvolvendo seu bot, o mesmo irá aparecer na tela. Então, dê dois cliques no mesmo para acessá-lo. Você será direcionado para a tela a seguir:



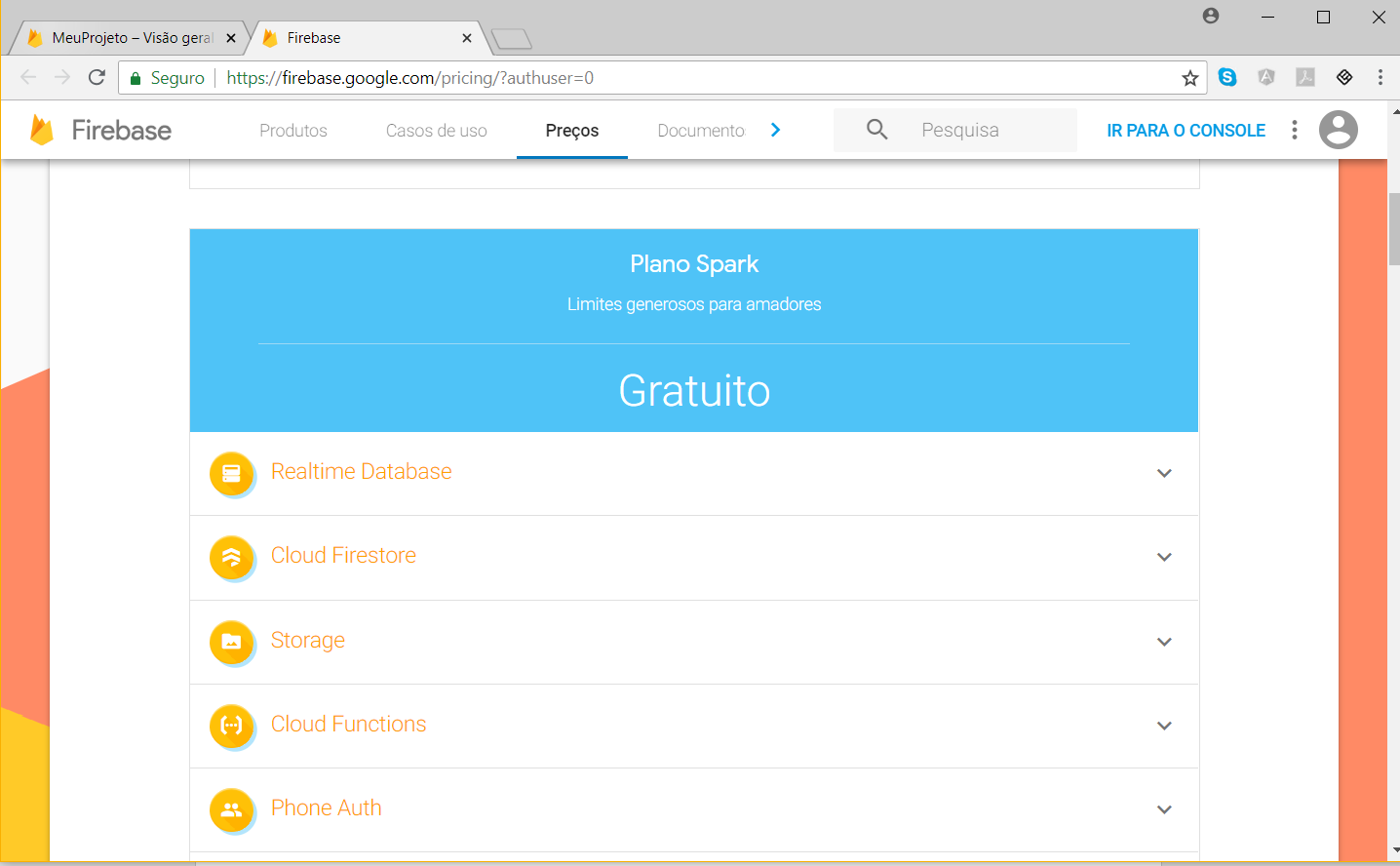
Dê uma olhada nas opções ou recursos do menu da direita.

Na opção Plano gratuito 🡪 clique em Fazer upgrade, você será direcionado para a seguinte tela



Essa tela tem informações sobre os planos de uso do FireBase.

Clique em Veja os detalhes e, você irá para a tela

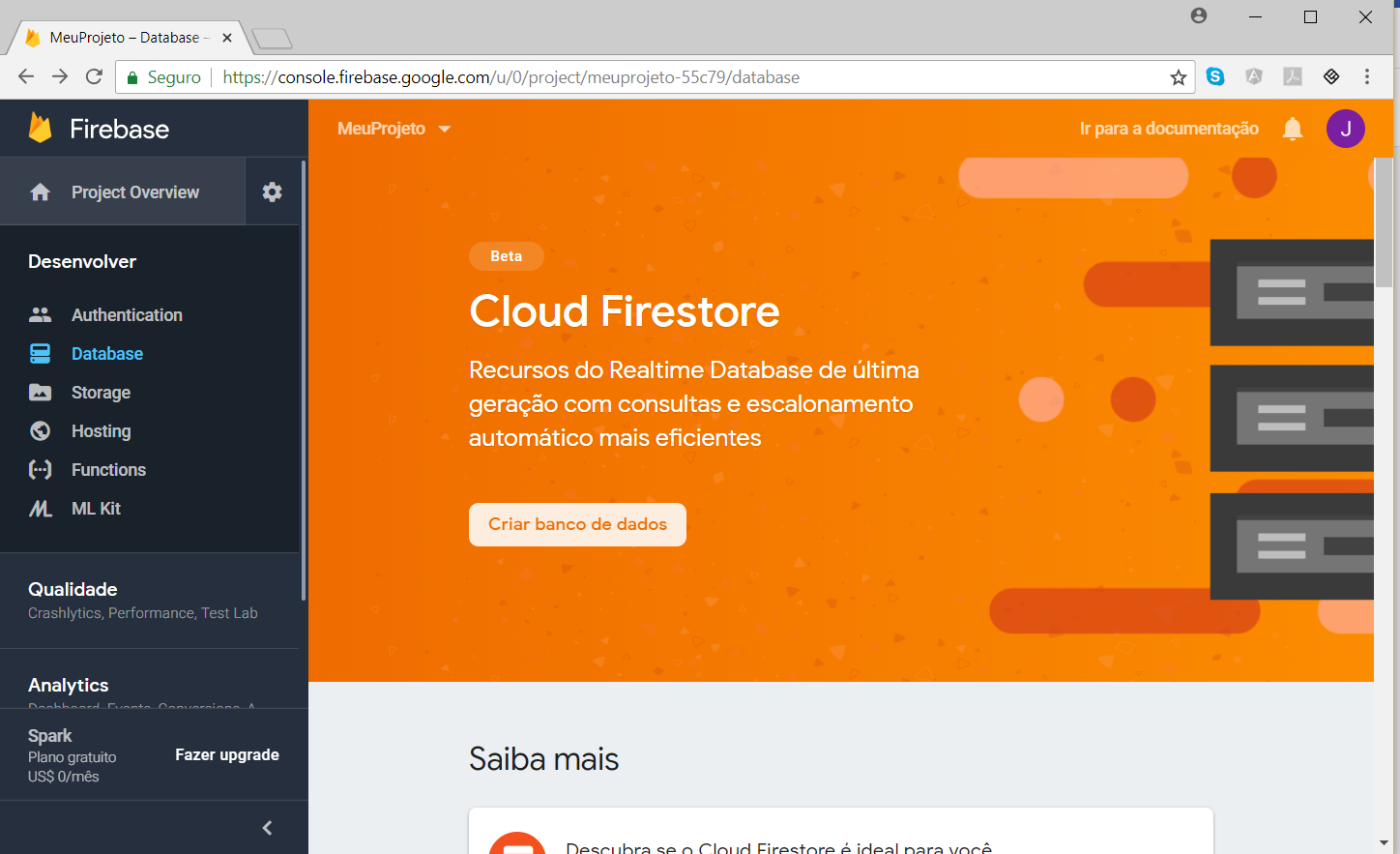


Dê uma olhada detalhada nos recursos disponibilizados em cada plano.

**Google Cloud Firestore**

Volte para o console de seu projeto.

Clique na opção **Database** e você será direcionado para a tela



Nessa tela temos duas opções de banco de dados: **Cloud Firestore** **e Realtime Database**

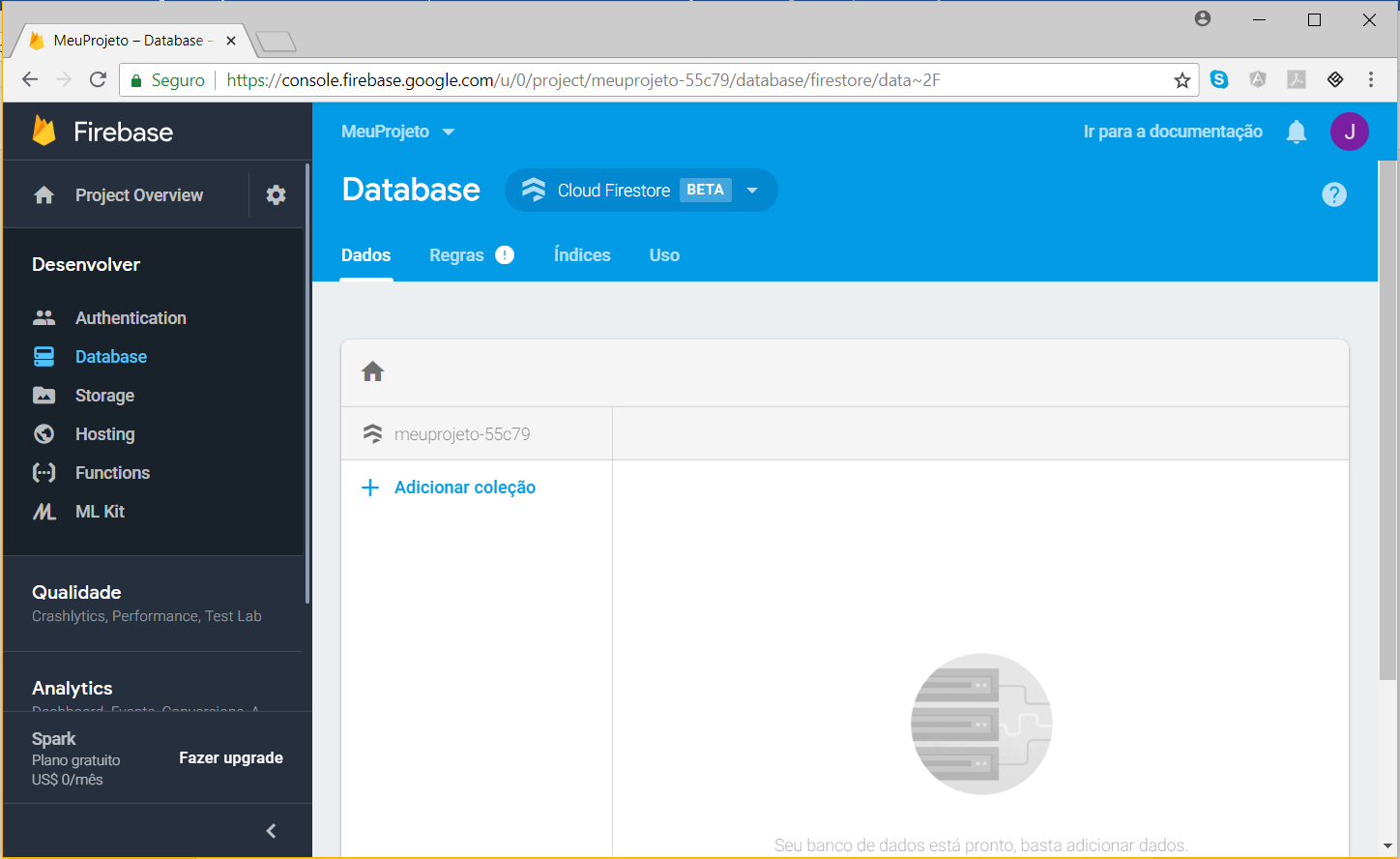
Para esse projeto de teste, vamos trabalhar com a opção Cloud Firestore. Portanto, clique em Criar banco de dados e, você irá para a seguinte tela



Nessa tela temos duas opções modo bloqueado e modo de teste. Vamos selecionar a opção modo de teste, por enquanto. Veja o aviso que o sistema fornece



Clique em ativar e você será levado para a seguinte tela



Nessa janela você crie as suas coleções, define as regras de acesso e assim por diante.

**Conceitos de Cloud Firestore**

O **Cloud Fierstore** é um banco de dados escalável que roda em uma plataforma de nuvem e ele não é um banco de dados relacionais. Ele tem alguns conceitos importantes como: documentos que serão criados dentro de coleções, como por exemplo nós podemos ter uma coleção de cursos e os documentos nesta coleção, sendo que, cada documento corresponde a cursos diferentes.

**Criando coleções e documentos no Cloud Firestore**

Crie a coleção cursos com os documentos

**R**

Campos: nome=R fundamentos, valor=250 e instrutor= José Cunha

**Python**

Campos: nome=Introdução ao Python, valor=300 e instrutor= Pedro

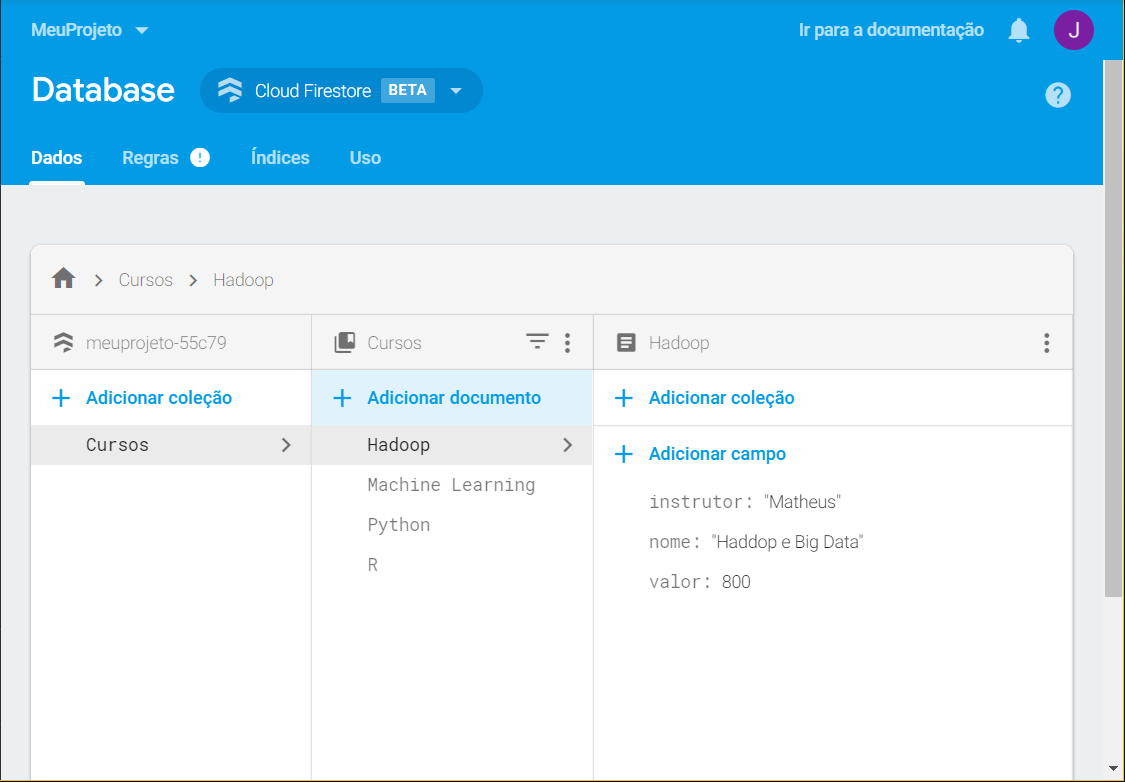
**Machine Learning**

Campos: nome=Machine learning para análise de dados, valor=500 e instrutor= Leo

**Hadoop**

Campos: nome=Hadoop e Big Data, valor=800 e instrutor= Matheus

Veja a tela a seguir



**Instalando o NodeJS e o Firebase -Tools**

Antes de instalar o CLI – Command Line Interface na sua máquina, instale o Node.js. Depois disso use o npm para instalar a CLI exectando este comando

**npm install -g firebase-tools**

agora você tem um comando “firebase” global que pode ser executado em qualquer janela de terminal da sua máquina. Depois de instalar a CLI do Firebase, faça login usando sua conta no Google:

**Logando no Firebase via linha de comando**

**firebase login**

com esse comando, você conecta a máquina local à sua conta do Firebase e tem acesso a seus projetos.

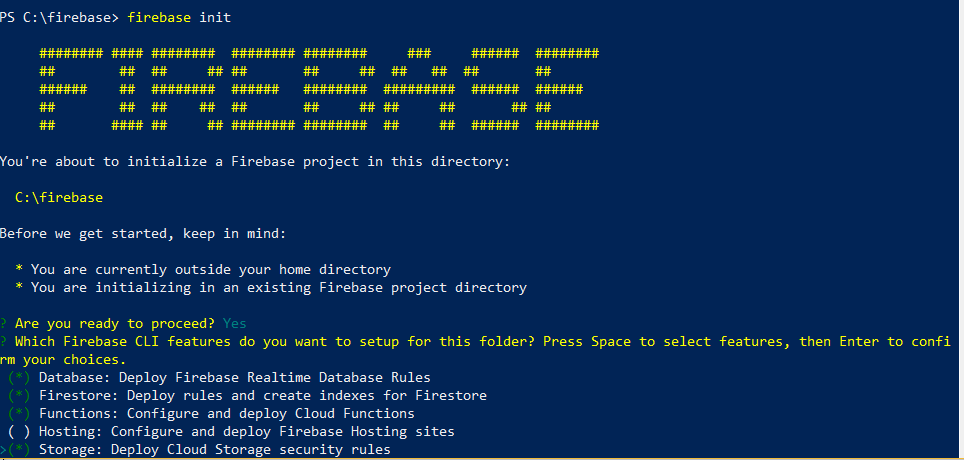
**Associando um diretório de projeto ao Firebase**

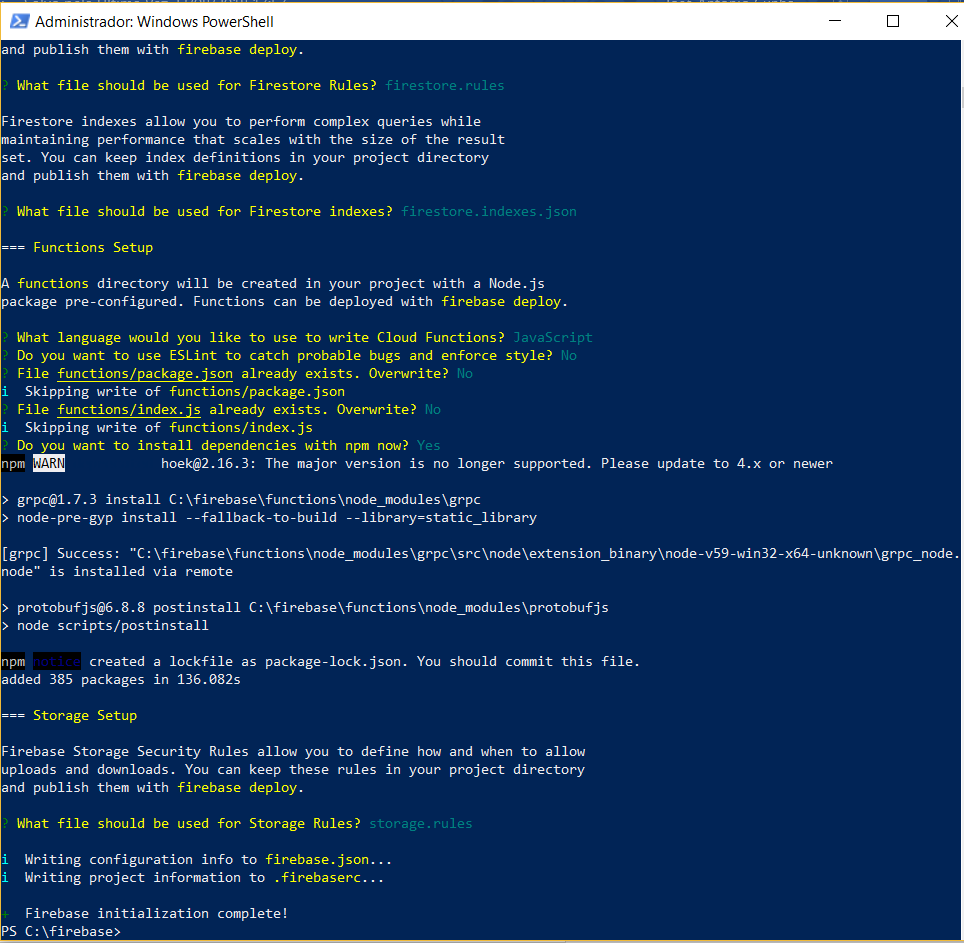
No terminal, mude o diretório atual para o novo diretório de projeto que quer inicializar e execute:

**firebase init**

ao executá-lo, algumas orientações são exibidas durante a configuração do diretório de projeto (no meu caso a pasta é firebase e, dentro dela deve ter o arquivo firebase.json), inclusive perguntando quais recursos do Firebase você quer usar.

Adicione os seguintes recursos, como mostra a janela a seguir:



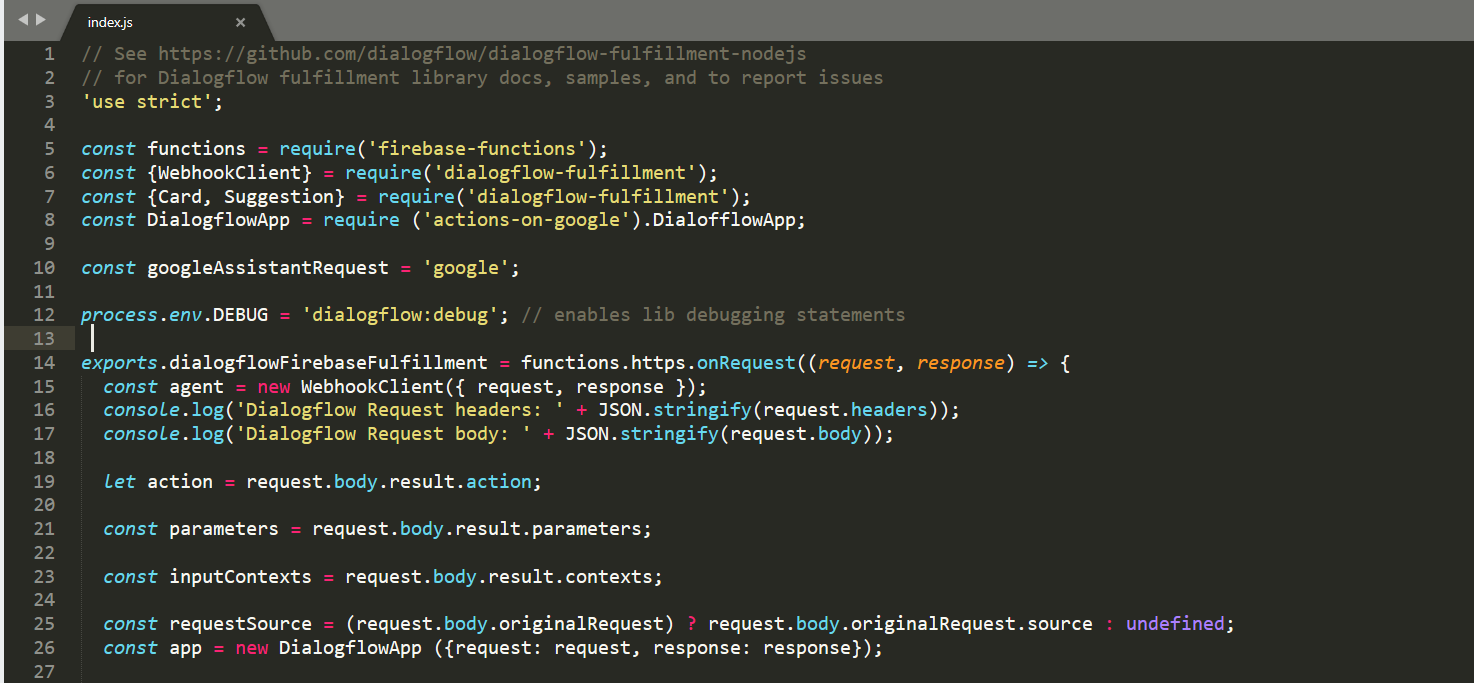


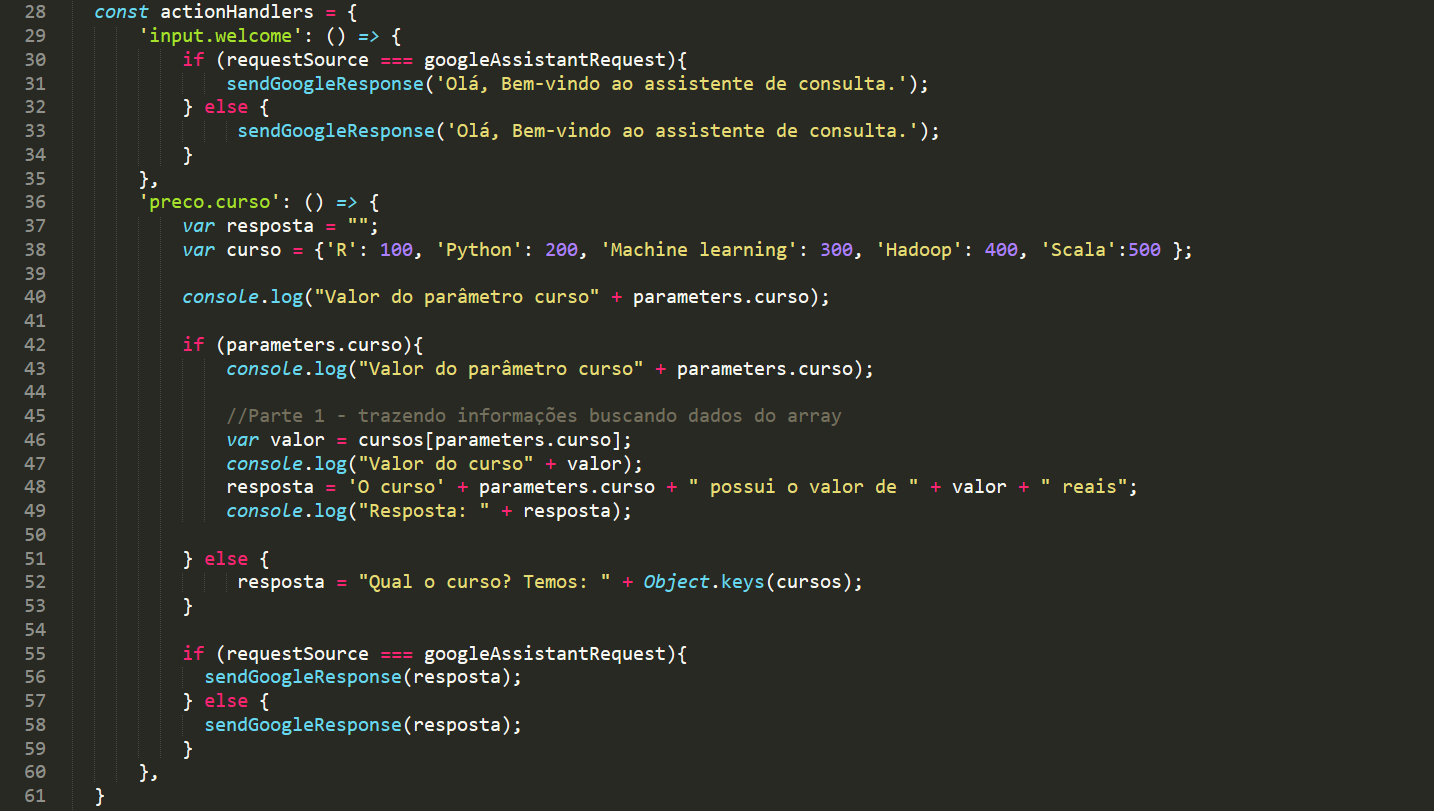
Para selecionar um recurso basta usar a barra de espaço e em seguida dá um Enter.

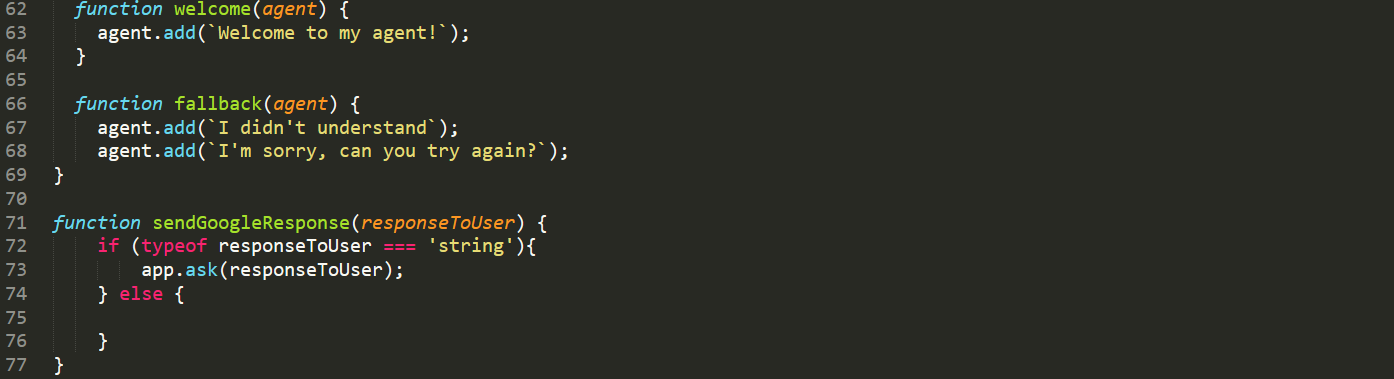
Agora ele está perguntando a qual projeto do Firebase você quer vincular esse diretório. Em seguida ele irá fazer várias perguntas, inclusive se você quer substituir os arquivos index.js e packger.js, responda não e, nas demais perguntas dê um Enter para aceitar o padrão.

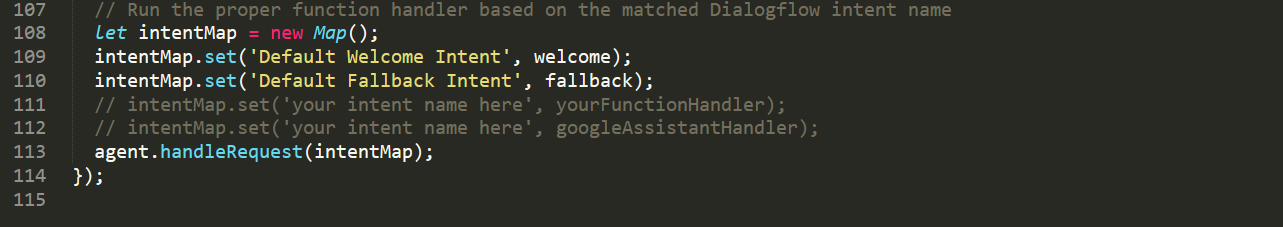
**Deploy para o Google Cloud Functions via Firebase CLI.**

Então, abra o arquivo index.js (Na pasta functions) em seu editor de código favorito e faças as mudanças necessárias.









Depois de ter feito as mudanças, é hora de fazer o deploy. Então, no terminal, execute

**firebase deploy --only funtions**

**Integração do Google Cloud Functions para o Clould Firestore**

Abra o arquivo package.json e altere a chave

**“firebase-admin”: “^5.4.2”**

Depois vá para o terminal e execute o seguinte comando

**npm install –save firebase-admin**

