

Desenvolvimento de Apps para Mobile

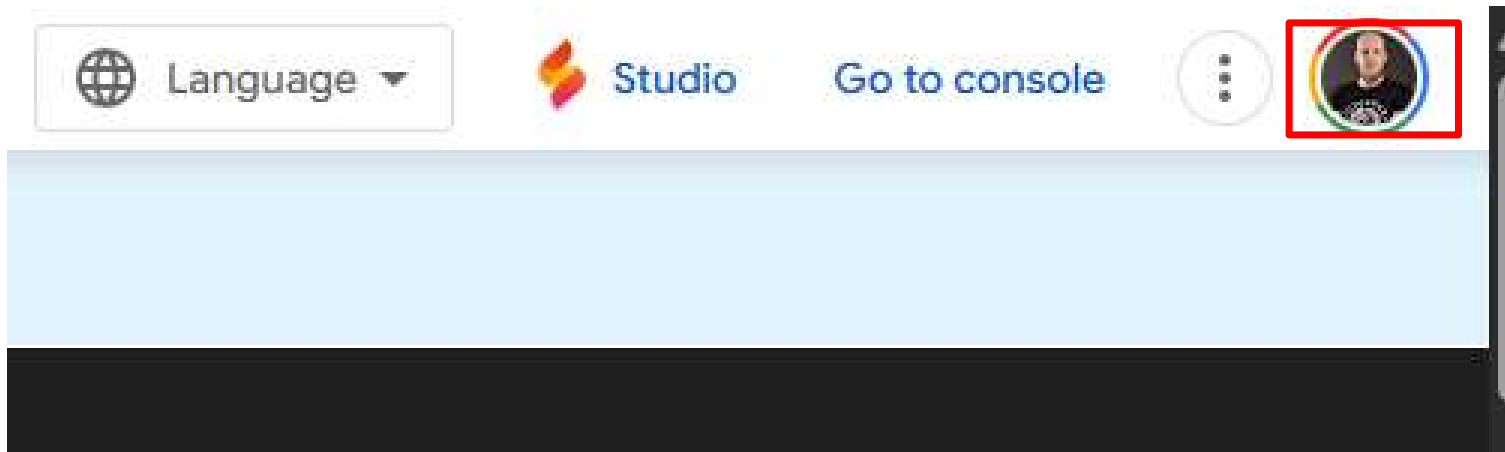
Fabrício Tonetto Londero

Aula 13 - Firebase



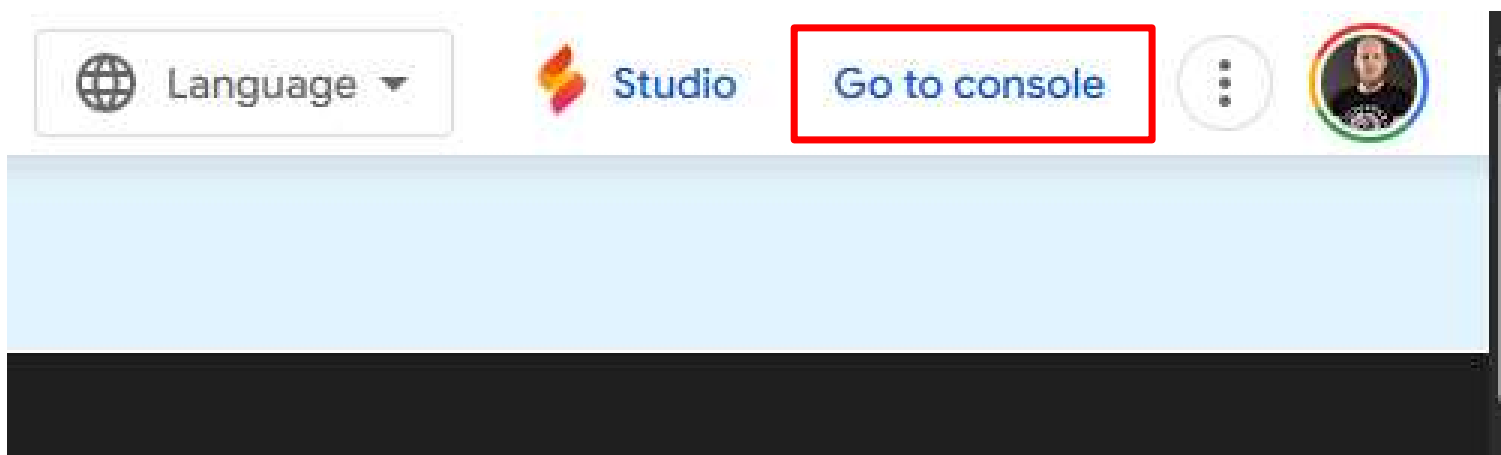
Configuração de um projeto

1. Acessar o site <https://firebase.google.com/> e realizar o login no canto superior direito:



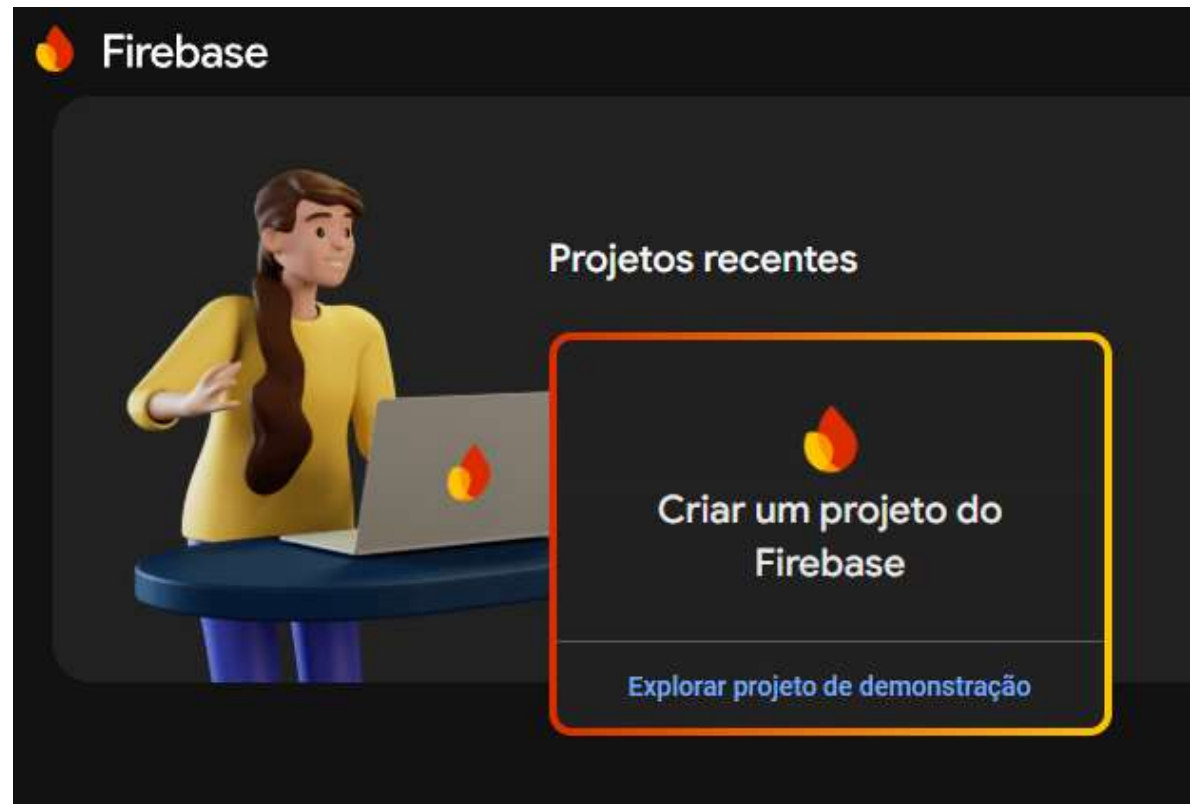
Configuração de um projeto

2. Após realizar o login, clique em **Ir para o console**:



Configuração de um projeto

- 3. Você irá para a tela inicial do Firebase, com todos os seus projetos. Para criarmos um projeto, clique em **Criar um projeto**:



Configuração de um projeto

- 4. Dê um nome para o seu projeto:

× Criar um projeto

Vamos começar nomeando o projeto[?]

Nome do projeto
Projeto AulaConfiguração de um projeto

✎ projeto-aula-db41f

☒ Participe do [Programa para desenvolvedores do Google](#) para aperfeiçoar sua jornada com assistência de IA, recursos de aprendizado, selos de perfil e muito mais.

Já tem um projeto do Google Cloud?
[Adicionar o Firebase ao projeto do Google Cloud](#)

Continuar

× Criar um projeto

Assistência de IA para seu projeto do Firebase

O Gemini está integrado ao console do Firebase para ajudar a simplificar o processo de desenvolvimento.

- ✦ Converse com o Gemini para planejar e criar seu aplicativo, solucionar problemas e receber recomendações personalizadas para seu projeto
- ⚙️ Receba assistência de IA no Firebase Crashlytics para depuração e solução de problemas nos seus apps Apple e Android
- 📩 Insights e recomendações de campanha para ajudar a manter o engajamento dos usuários com o Firebase Cloud Messaging
- 🔗 Geração de esquemas e análise detalhada de dados em linguagem natural com o Firebase Data Connect

☒ **Ativar o Gemini no Firebase**
Recomendado

Exonerações de responsabilidade:
O Gemini no Firebase ainda está em desenvolvimento e pode fornecer informações imprecisas. Teste os códigos gerados antes de usá-los nos seus apps. O Gemini pode interagir com seus dados do Firebase e personalizar as respostas do app, mas não usa seus comandos ou dados para treinamento. [Saiba como treinamos nossos modelos](#) e [O uso do Gemini no Firebase está sujeito aos Termos de Serviço do Google](#) e à [Política de uso proibido da IA generativa](#)

[Anterior](#)

Continuar

× 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
- 👤 Gatilhos do Cloud Functions com base em eventos
- 👤 Segmentação de usuários em produtos do Firebase
- 📊 Geração de relatórios ilimitada gratuita
- ⚙️ Registros de navegação estrutural no Crashlytics

☒ **Ativar o Google Analytics neste projeto**
Recomendado

[Anterior](#)

Continuar

× Criar um projeto

Configurar o Google Analytics

Localização do Analytics ⓘ

Brasil

O Google Analytics é uma ferramenta de negócios. Use-a exclusivamente para fins relacionados ao seu comércio, negócio, ofício ou profissão.

Configurações de compartilhamento de dados e termos do Google Analytics

☒ Usar as configurações padrão para o compartilhamento de dados do Google Analytics. [Saiba mais](#) ⓘ

- × Compartilhe seus dados do Analytics com o Google para melhorar os produtos e serviços da empresa
- ✓ Compartilhe seus dados do Analytics com o Google para ativar o Comparativo de mercado
- ✓ Compartilhe seus dados do Analytics com o Google para ativar o suporte técnico
- ✓ Compartilhe seus dados do Analytics com os especialistas em contas do Google

☒ Eu aceito os [Termos do Google Analytics](#) ⓘ

Após a criação do projeto, uma nova propriedade do Google Analytics será criada e vinculada ao seu projeto do Firebase. Esse processo vai permitir o fluxo de dados entre os produtos. Os dados da propriedade do Google Analytics exportados para o Firebase ficam sujeitos aos Termos de Serviço do Firebase, e os dados do Firebase importados para o Google Analytics ficam sujeitos aos Termos de Serviço do Google Analytics. [Saiba mais](#) ⓘ.

[Anterior](#)

Criar projeto



Preparando seu projeto, aguarde.

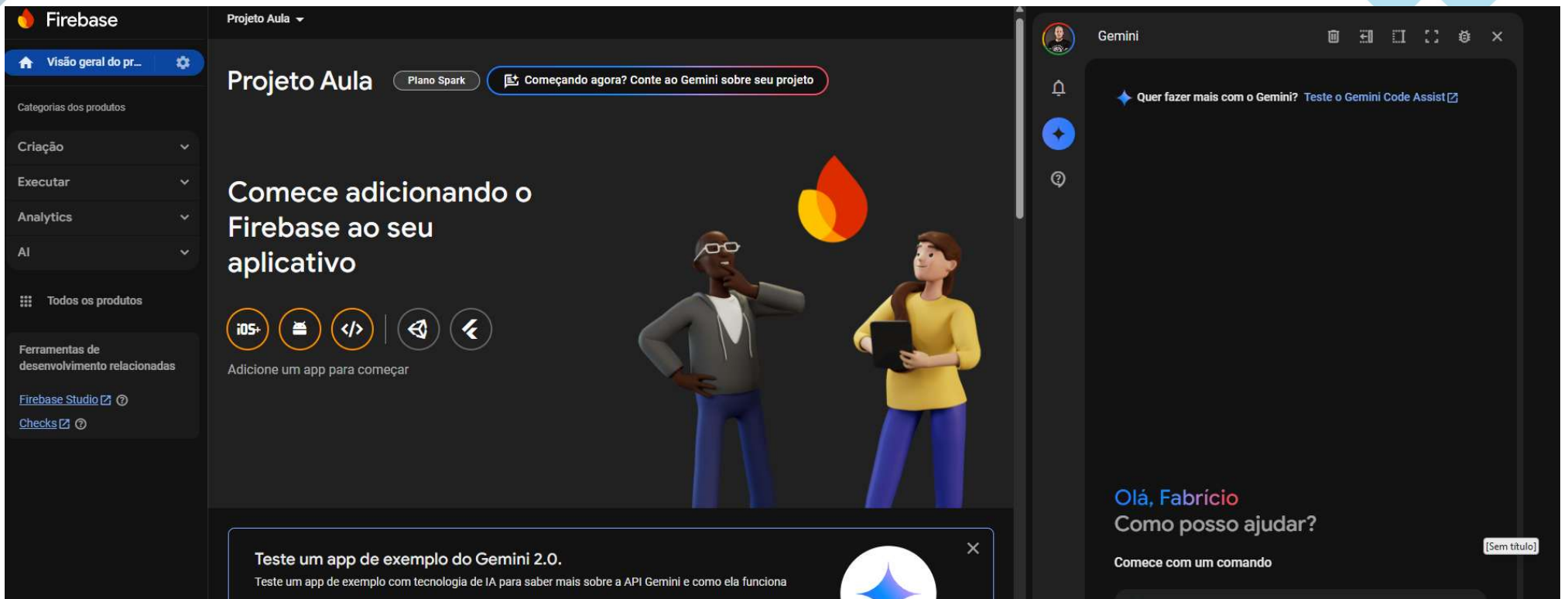
Projeto Aula



Projeto Aula

✓ Seu projeto do Firebase está pronto

Continuar



Vinculação com o Android Studio

[illegible]

× Adicionar o Firebase ao seu app Android



Registrar app

Nome do pacote Android: com.ernakh.googlefirebaseandroidstudiojava, apelido do app: Google Firebase Android S



Faça o download e adicione o arquivo de configuração para o Android Studio abaixo | [Unity](#) [C++](#)

📄 Fazer o download de google-services.json

Mude para a visualização Projeto no Android Studio para acessar o diretório raiz do seu projeto.

Mova o arquivo `google-services.json` salvo para o diretório raiz do módulo (nível do app).



google-services.json

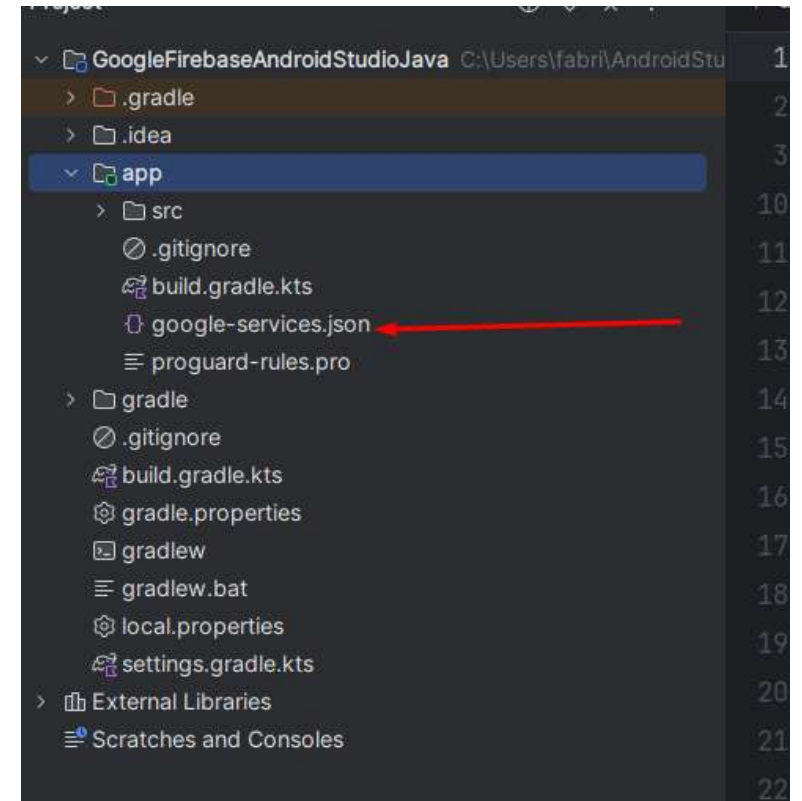
Próxima



Adicionar o SDK do Firebase



Próximas etapas



3 Adicionar o SDK do Firebase

Instruções para Gradle | [Unity](#) [C++](#)

★ Você ainda usa a sintaxe `buildscript` para gerenciar plug-ins? Saiba como [adicionar plug-ins do Firebase](#) usando essa sintaxe.

1. Para tornar os valores de configuração do arquivo `google-services.json` acessíveis aos SDKs do Firebase, você precisa do plug-in do Gradle para os Serviços do Google.

☒ DSL do Kotlin (`build.gradle.kts`) ☐ Groovy (`build.gradle`)

Adicione o plug-in como uma dependência do arquivo `build.gradle.kts` no nível do projeto:

Arquivo do Gradle no nível raiz (nível do projeto) (`<project>/build.gradle.kts`):

```
plugins {  
    // ...  
  
    // Add the dependency for the Google services Gradle plugin  
    id("com.google.gms.google-services") version "4.4.2" apply false  
}
```

2. Em seguida, no arquivo `build.gradle.kts` do módulo (nível do app), adicione o plug-in `google-services` e todos os SDKs do Firebase que você quer usar no app:

Arquivo do Gradle do módulo (nível do app) (`<project>/<app-module>/build.gradle.kts`):

2. Em seguida, no arquivo `build.gradle.kts` do módulo (nível do app), adicione o plug-in `google-services` e todos os SDKs do Firebase que você quer usar no app:

Arquivo do Gradle do módulo (nível do app) (`<project>/<app-module>/build.gradle.kts`):

```
plugins {  
    id("com.android.application")  
    // Add the Google services Gradle plugin  
    id("com.google.gms.google-services")  
    ...  
}  
  
dependencies {  
    // Import the Firebase BoM  
    implementation(platform("com.google.firebase:firebase-bom:33.13.0"))  
  
    // TODO: Add the dependencies for Firebase products you want to use  
    // When using the BoM, don't specify versions in Firebase dependencies  
    implementation("com.google.firebase:firebase-analytics")  
  
    // Add the dependencies for any other desired Firebase products  
    // https://firebase.google.com/docs/android/setup#available-libraries  
}
```

Ao usar a BoM do Firebase para Android, o app sempre vai utilizar versões da biblioteca do Firebase compatíveis.

[Saiba mais](#)

3. Depois de adicionar o plug-in e os SDKs desejados, sincronize seu projeto do Android com os arquivos do Gradle.

[Anterior](#)

[Próxima](#)

× Adicionar o Firebase ao seu app Android



Registrar app

Nome do pacote Android: com.ernakh.googlefirebaseandroidstudiojava, apelido do app: Google Firebase Android SI



Faça o download e adicione o arquivo de configuração



Adicionar o SDK do Firebase



Próximas etapas

Está tudo pronto!

Certifique-se de verificar a [documentação](#) para saber como começar a usar os produtos Firebase que quiser usar em seu aplicativo.

Também é possível explorar [apps de amostra do Firebase](#).

Como opção, continue no console para explorar o Firebase.

[Anterior](#)

[Continuar no console](#)

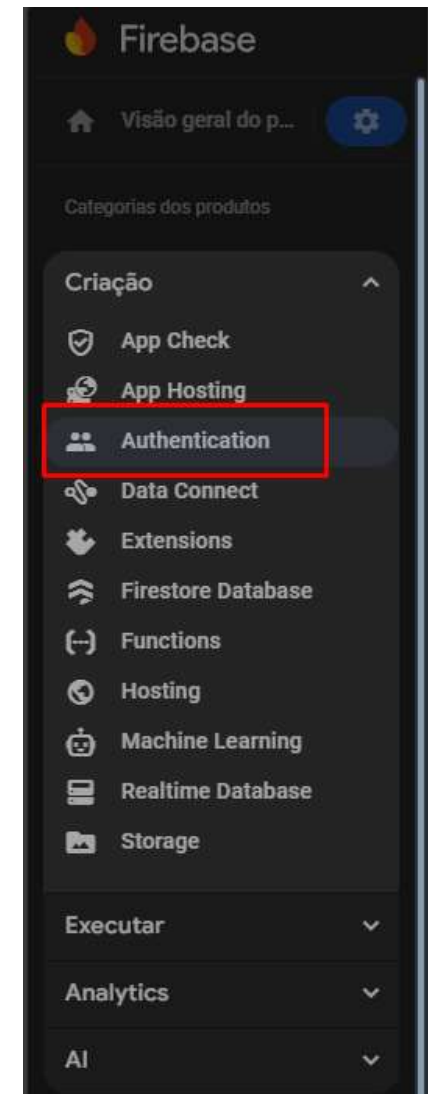


Android e Firebase

Autenticação

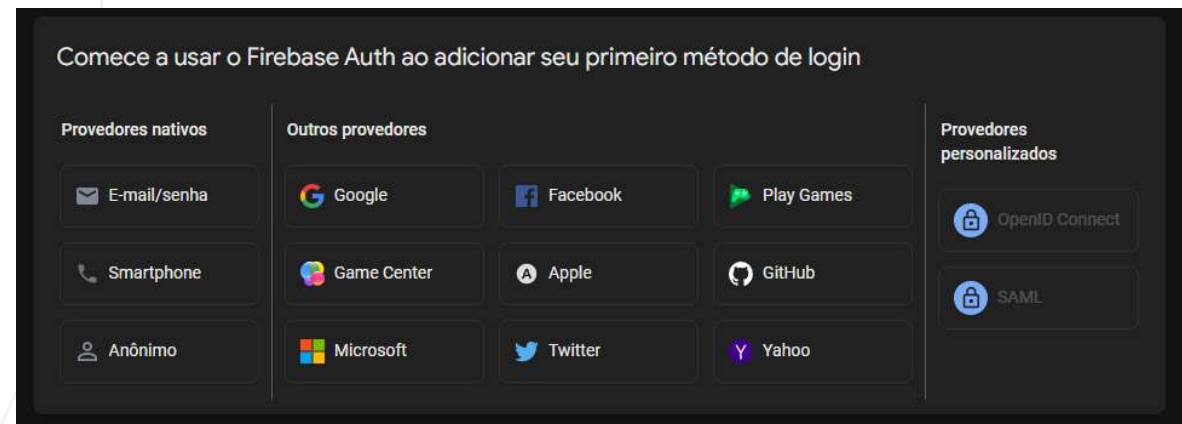
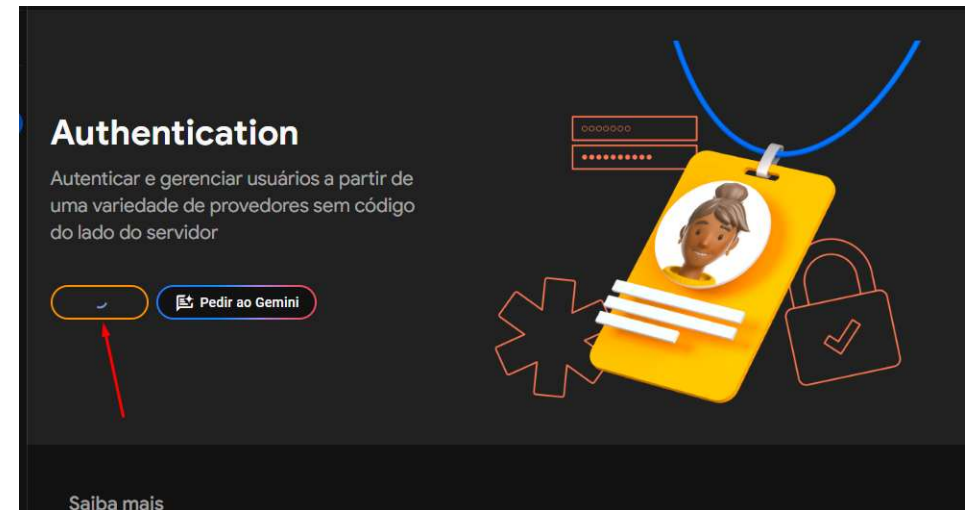
Autenticação


1. Primeiramente, precisamos adicionar a aplicação de autenticação ao nosso projeto do Firebase. Para isso, acesse o console do seu projeto, clique em **“Todos os produtos”** na esquerda, vá em **Authentication**.




Autenticação

2. Na janela que abrir, clique em “Vamos começar” e escolha os métodos de autenticação e ative os que você deseja adicionar a sua aplicação. Nesse exemplo, adicionaremos a autenticação por email.



 E-mail/senha

☒ Ativar

Permite que os usuários se inscrevam usando o endereço de e-mail e a senha deles. Nossos SDKs também fornecem verificação de endereço de e-mail, recuperação de senha e componentes essenciais para alteração do endereço de e-mail. [Saiba mais](#) 

Link do e-mail (login sem senha)

☐ Ativar

Cancelar

Salvar

Provedores de login

Adicionar novo fornecedor

Provedor

Status



E-mail/senha



ativado

MainActivity



A user interface for a login and registration screen. It features two text input fields at the top, one for an email address and one for a password. Below these fields are two buttons: 'Logar' (Login) and 'Cadastrar' (Register). The entire form is set against a light pink background.

fabriciotonettolondero@g

Logar

Cadastrar

MainActivity

```
FirebaseAuth mAuth = FirebaseAuth.getInstance();
```

```
public void cadastrarUsuario(View v)
{
    EditText edtEmail = findViewById(R.id.edtEmail);
    EditText edtSenha = findViewById(R.id.edtSenha);

    mAuth.createUserWithEmailAndPassword(edtEmail.getText().toString(), edtSenha.getText().toString())
        .addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                Toast.makeText(this, "Usuário criado com sucesso", Toast.LENGTH_LONG).show();
                Log.d("FIREBASE", "Usuário criado com sucesso");
            } else {
                Toast.makeText(this, "Erro ao criar usuário: " + task.getException(), Toast.LENGTH_LONG).show();
                Log.e("FIREBASE", "Erro ao criar usuário", task.getException());
            }
        });
}
```

MainActivity

```
public void logarUsuario(View v)
{
    EditText edtEmail = findViewById(R.id.edtEmail);
    EditText edtSenha = findViewById(R.id.edtSenha);

    mAuth.signInWithEmailAndPassword(edtEmail.getText().toString(), edtSenha.getText().toString())
        .addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                Toast.makeText(this, "Login bem-sucedido", Toast.LENGTH_LONG).show();
                Log.d("FIREBASE", "Login bem-sucedido");
            } else {
                Toast.makeText(this, "Erro no login: " + task.getException(), Toast.LENGTH_LONG).show();
                Log.e("FIREBASE", "Erro no login", task.getException());
            }
        });
}
```

Curiosidade: Deletar usuário?

- O próprio usuário pode se excluir:

```
FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();

if (user != null) {
    user.delete()
        .addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                Log.d("FIREBASE", "Usuário deletado com sucesso.");
                // Você pode redirecionar o usuário para a tela de login, por exemplo
            } else {
                Log.e("FIREBASE", "Erro ao deletar usuário", task.getException());
            }
        });
} else {
    Log.e("FIREBASE", "Nenhum usuário logado para deletar.");
}
```

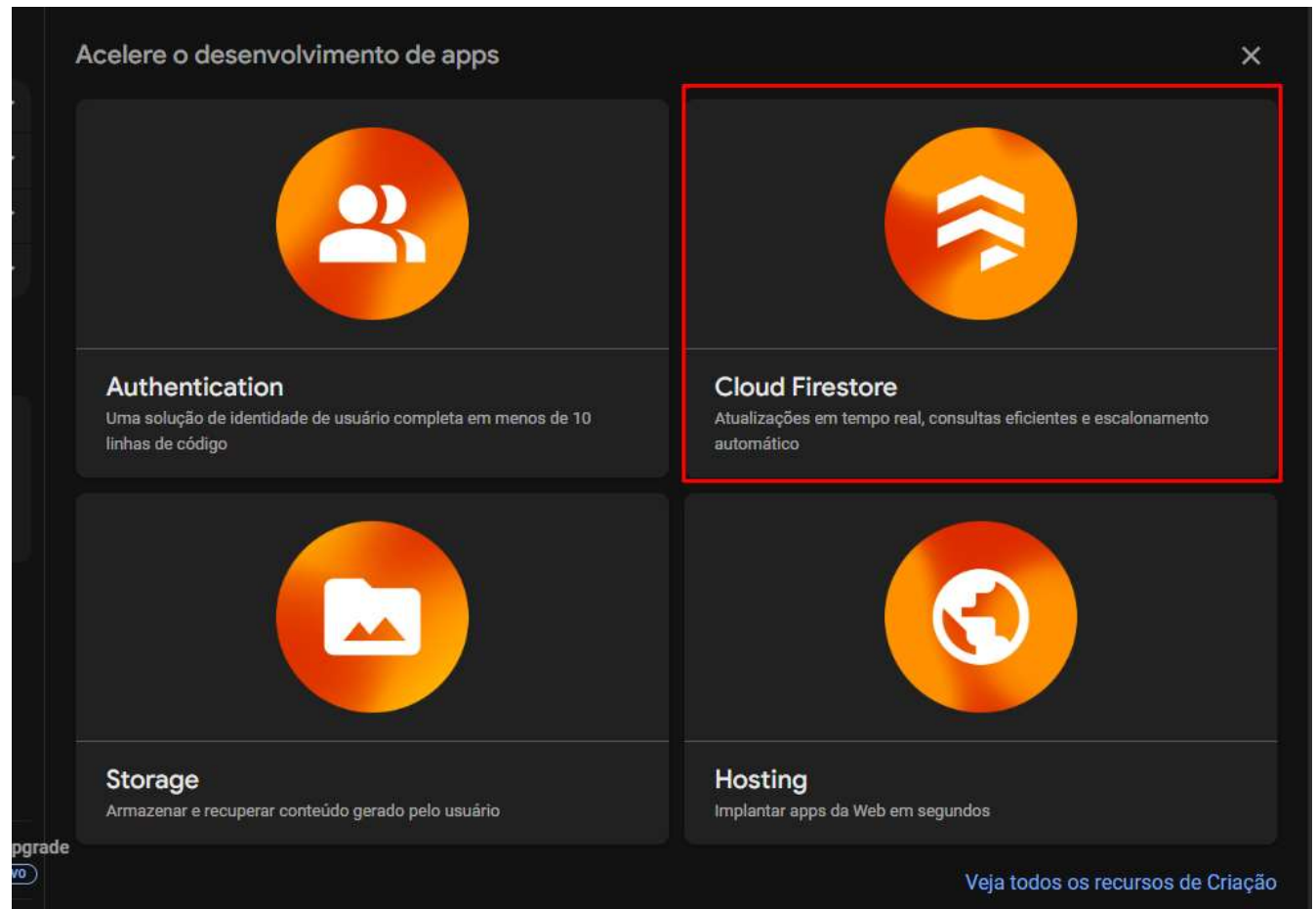
Curiosidade: Deletar usuário?

- Ou, como o Firebase Authentication não permite que você delete um usuário diretamente a partir de outro usuário autenticado via SDK cliente (como com mAuth), devemos usar o Firebase Admin SDK (no backend ou servidor)

```
admin.auth().deleteUser(uid)
  .then(() => {
    console.log('Usuário deletado com sucesso');
  })
  .catch((error) => {
    console.log('Erro ao deletar usuário:', error);
  });
```

Devemos criar uma base
de dados no Firestore
Database


CRUD



Cloud Firestore

Atualizações em tempo real, consultas eficientes e escalonamento automático

Criar banco de dados

 Pedir ao Gemini

Criar banco de dados



1 Definir nome e local — 2 Regras de segurança



Quer usar o Firestore, que tem compatibilidade com o MongoDB? Crie um banco de dados do Firestore Enterprise no console do Google Cloud.

Saiba mais [↗](#)

ID do banco de dados

(default)

Local

nam5 (United States) ▼

ⓘ Sua configuração de localização é onde os dados do Cloud Firestore serão armazenados



Não será possível alterar o local depois de terminar a configuração dele.

Saiba mais [↗](#)

Cancelar

Próxima

Criar banco de dados



Definir nome e local



Regras de segurança

Após definir a estrutura de dados, será necessário criar regras para proteger seus dados.

[Saiba mais](#)



Iniciar no modo de produção

Seus dados são particulares por padrão. O acesso de leitura/gravação do cliente será concedido apenas se especificado por suas regras de segurança.



Iniciar no modo de teste

Por padrão, seus dados estão definidos para permitir uma configuração rápida. Porém, você precisa atualizar suas regras de segurança em até 30 dias para permitir o acesso de leitura/gravação do cliente em longo prazo.

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if false;
    }
  }
}
```




Todas as leituras e gravações de terceiros serão negadas

Cancelar


Criar


Cloud Firestore

Adicionar banco de dados

 Pergunte ao Gemini como começar a usar o Firestore

- Dados
- Regras
- Índices
- Recuperação de desastres NOVO
- Uso
- Extensions

 Proteja os recursos do Cloud Firestore de abusos, como fraude de faturamento ou phishing [Configurar o App Check](#) ✕

 Conheça a edição Enterprise do Firestore, que tem compatibilidade com o MongoDB. Crie um banco de dados do Firestore Enterprise no console do Google Cloud. [Saiba mais](#) Dispensar

[Visualização do painel](#) [Criador de consultas](#) ⋮



 Mais no Google Cloud ▼

 (default)

+ Iniciar coleção

Iniciar uma coleção

1

 Fornecer um código à coleção

2

 Adicionar o primeiro documento

Caminho pai

/

Código da coleção ?

produtos

Cancelar

Próxima

Iniciar uma coleção

✓ Fornecer um código à coleção

2 Adicionar o primeiro documento

Caminho pai do documento

/produtos

ID do documento ?

o9TEXdUai5tsmYW9sDVU

Campo	Tipo	
nome	string	⊖
String		
<div></div>		

Campo	Tipo	Valor
estoque	number	0

⊕ Adicionar campo

Cancelar

Salvar

Cloud Firestore

[Adicionar banco de dados](#)[Pergunte ao Gemini como começar a usar o Firestore](#)[Dados](#)[Regras](#)[Índices](#)[Recuperação de desastres](#) NOVO[Uso](#)[Extensions](#)

Proteja os recursos do Cloud Firestore de abusos, como fraude de faturamento ou phishing

[Configurar o App Check](#)

Conheça a edição Enterprise do Firestore, que tem compatibilidade com o MongoDB. Crie um banco de dados do Firestore Enterprise no console do Google Cloud.

[Saiba mais](#)[Dispensar](#)[Visualização do painel](#)[Criador de consultas](#)

> produtos > o9TEXdUai5tsm...

[Mais no Google Cloud](#)

(default)



produtos



o9TEXdUai5tsmYW9sDVU



+ Iniciar coleção

+ Adicionar documento

+ Iniciar coleção

produtos



o9TEXdUai5tsmYW9sDVU



+ Adicionar campo

estoque: 0

nome: ""

CRUD

- Com base na nossa coleção, vamos criar a classe Produto
- <https://github.com/Ernakh/GoogleFirestoreAndroidStudioJava/blob/main/app/src/main/java/com/ernakh/googlefirebaseandroidstudiojava/Produto.java>

Vamos criar um ProdutoAdapter

- <https://github.com/Ernakh/GoogleFirebaseAndroidStudioJava/blob/main/app/src/main/java/com/ernakh/googlefirebaseandroidstudiojava/ProdutoAdapter.java>
- Utilizamos para deixar a atualização dos objetos com a interface (RecyclerView) de forma automática quando for identificada alterações na lista de produtos e nas ações dos usuários no componente

Activity

Nome

Estoque

Salvar Produto

Item 0

Item 1

Item 2

Item 3

Item 4

Item 5

Item 6


Item 7

Item 8

Item 9

Cloud Firestore

Adicionar banco de dados

 Pergunte ao Gemini como começar a usar o Firestore

Dados


Regras

Índices

Recuperação de desastres

NOVO

Uso

 Extensions

Desenvolver e testar



Hoje • 3:05 PM



Hoje • 2:34 PM

Comparar com as regras mais recentes



Suas regras de segurança estão definidas como públicas. Com essa configuração, qualquer pessoa pode roubar, modificar ou excluir informações do seu banco de dados

Saiba mais [↗](#)

1
2
3
4
5
6
7

```
rules_version = '2';  
  
service cloud.firestore {  
  match /databases/{database}/documents {  
    match /{document=**} {  
      allow read, write: if false;  
    }  
  }  
}
```

- rules_version = '2';
- service cloud.firestore {
- match /databases/{database}/documents {
- match /{document=**}{
- allow read, write: if true;
- }
- }
- }

Activity

```
private FirebaseFirestore db;  
private EditText edtNome, edtEstoque;  
private RecyclerView recyclerProdutos;  
private List<Produto> listaProdutos = new ArrayList<>();  
private ProdutoAdapter adapter;  
private Produto produtoEditando = null;
```


Activity - onCreate

```
db = FirebaseFirestore.getInstance();

edtNome = findViewById(R.id.edtNome);
edtEstoque = findViewById(R.id.edtEstoque);
recyclerProdutos = findViewById(R.id.recyclerProdutos);
recyclerProdutos.setLayoutManager(new LinearLayoutManager(this));
adapter = new ProdutoAdapter(listaProdutos);
recyclerProdutos.setAdapter(adapter);

findViewById(R.id.btnSalvar).setOnClickListener(v -> salvarProduto());

carregarProdutos();
```

Activity

```
private void carregarProdutos() {
    db.collection("produtos")
        .get()
        .addOnSuccessListener(query -> {
            listaProdutos.clear();
            for (QueryDocumentSnapshot doc : query) {
                Produto p = doc.toObject(Produto.class);
                p.setId(doc.getId());
                listaProdutos.add(p);
            }
            adapter.notifyDataSetChanged();
        });

    adapter.setOnItemClickListener(produto -> {
        edtNome.setText(produto.getNome());
        edtEstoque.setText(String.valueOf(produto.getEstoque()));
        produtoEditando = produto;
        ((Button) findViewById(R.id.btnSalvar)).setText("Atualizar Produto");
    });
}
```

Activity

```
private void limparCampos() {  
    edtNome.setText("");  
    edtEstoque.setText("");  
    produtoEditando = null;  
    ((Button) findViewById(R.id.btnSalvar)).setText("Salvar Produto");  
}
```

```
private void salvarProduto() {
    String nome = edtNome.getText().toString();
    int estoque = Integer.parseInt(edtEstoque.getText().toString());
    if (produtoEditando == null) {
        Produto produto = new Produto(null, nome, estoque);
        db.collection("produtos")
            .add(produto)
            .addOnSuccessListener(doc -> {
                produto.setId(doc.getId());
                Toast.makeText(this, "Produto salvo!", Toast.LENGTH_SHORT).show();
                limparCampos();
                carregarProdutos();
            });
    } else {
        produtoEditando.setNome(nome);
        produtoEditando.setEstoque(estoque);

        db.collection("produtos").document(produtoEditando.getId())
            .set(produtoEditando)
            .addOnSuccessListener(aVoid -> {
                Toast.makeText(this, "Produto atualizado!", Toast.LENGTH_SHORT).show();
                limparCampos();
                carregarProdutos();
            });
    }
}
```

Nome

Estoque

Salvar Produto

caneta

Estoque: 32

borracha

Estoque: 1

Estoque: 0

<div><div>produtos</div><div><div></div><div></div></div></div>	<div><div>6pJiPFTDCUGEK9HxUL24</div><div><div></div><div></div></div></div>
<div><div>+ Adicionar documento</div></div>	<div><div>+ Iniciar coleção</div></div>
<div><div>6pJiPFTDCUGEK9HxUL24</div><div>eM73ze1L2Kwpchf8ejgu</div><div>o9TEXdUai5tsmYW9sDVU</div></div>	<div><div>+ Adicionar campo</div><div>estoque: 32</div><div>id: "6pJiPFTDCUGEK9HxUL24"</div><div>nome: "caneta"</div></div>

Nome

Estoque

Salvar Produto

caneta

Estoque: 15

borracha

Estoque: 1

Estoque: 0

	<div><div>produtos</div><div></div></div>	<div><div>6pJiPFTDCUGEK9HxUL24</div><div></div></div>
	<div>+ Adicionar documento</div>	<div>+ Iniciar coleção</div>
	<div><div>6pJiPFTDCUGEK9HxUL24</div><div>eM73ze1L2Kwpchf8ejgu</div><div>o9TEXdUai5tsmYW9sDVU</div></div>	<div><div>+ Adicionar campo</div><div>estoque: 15</div><div>id: "6pJiPFTDCUGEK9HxUL24"</div><div>nome: "caneta"</div></div>