

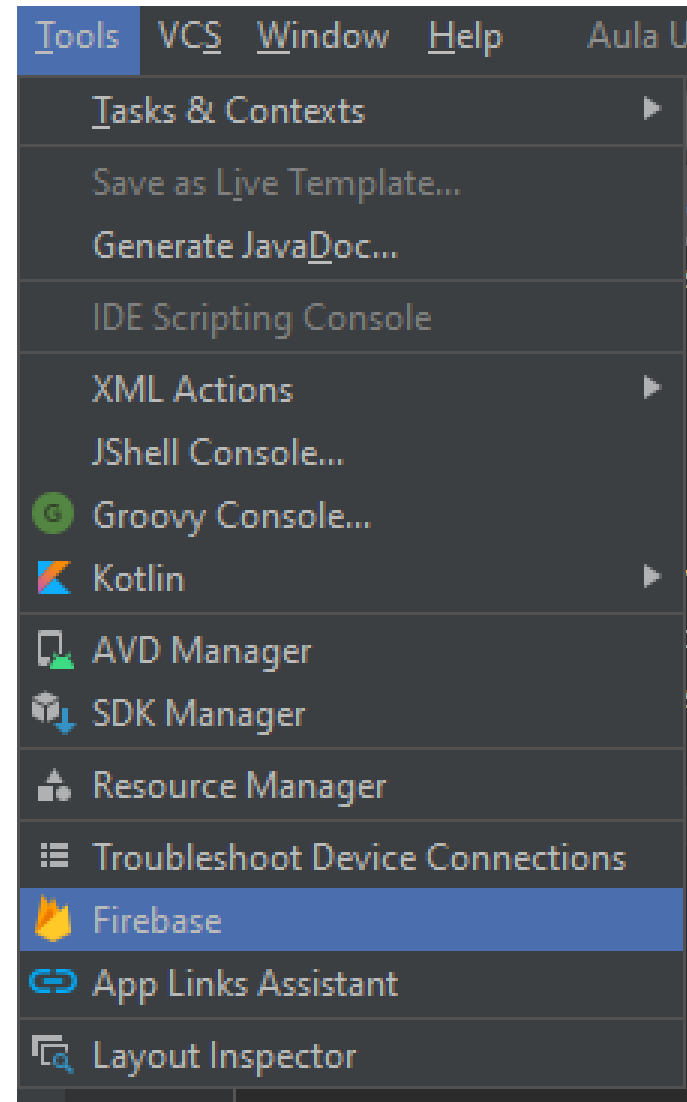
Enviar imagens para o firebase Storage

Prof. Me. Hélio Esperidião



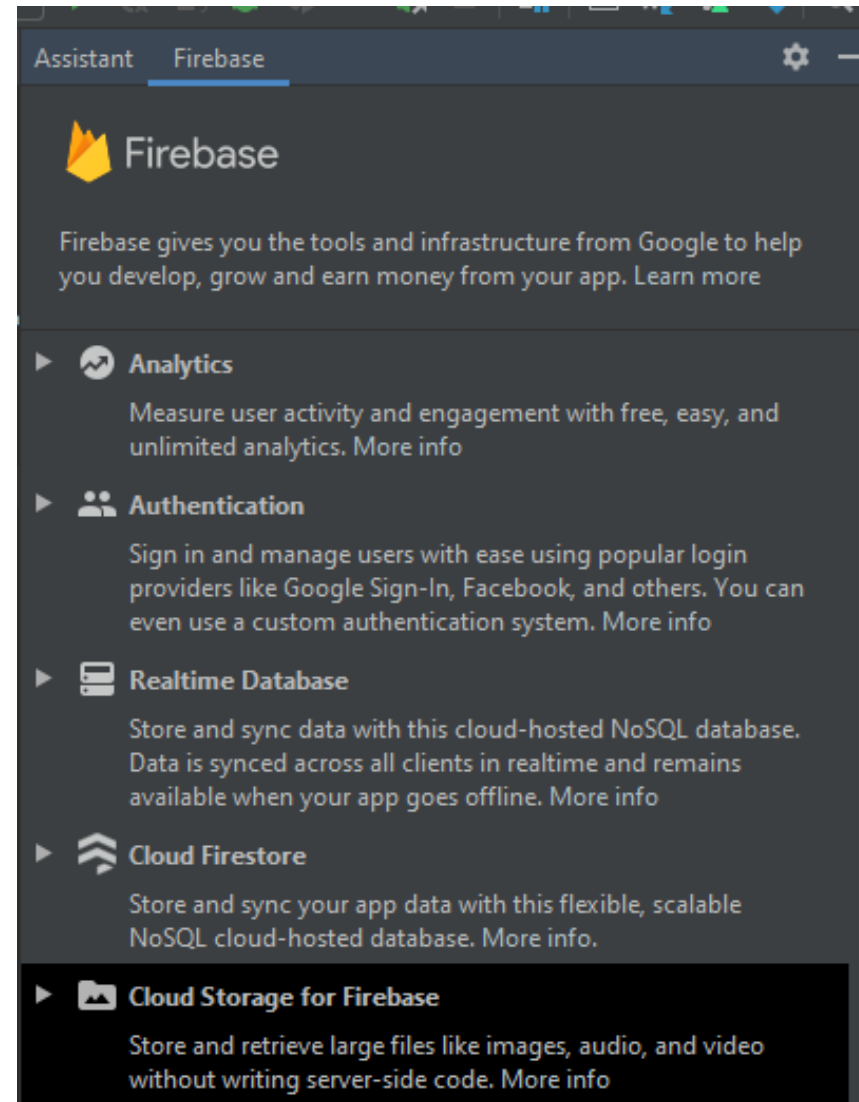
Configuração firebase

- Acesse o assistente do firebase no menu tools



Configuração firebase

- Selecione a opção cloud storage for firebase



Click em get Started with cloud storage

▼ Cloud Storage for Firebase

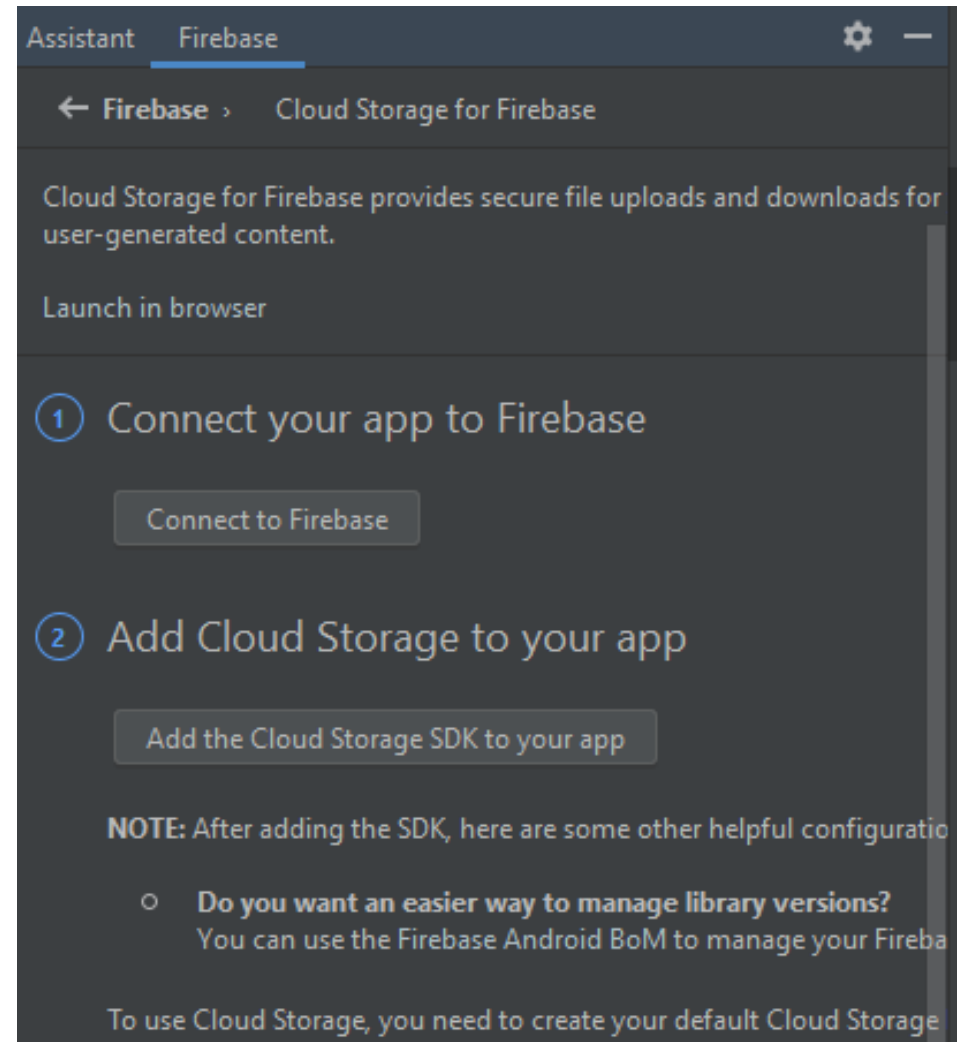
Store and retrieve large files like images, audio, and video without writing server-side code. [More info](#)

 [Get started with Cloud Storage](#)

 [Get started with Cloud Storage \[KOTLIN\]](#)

Conecte

- Conecte seu projeto kotlin a um projeto no fireabase





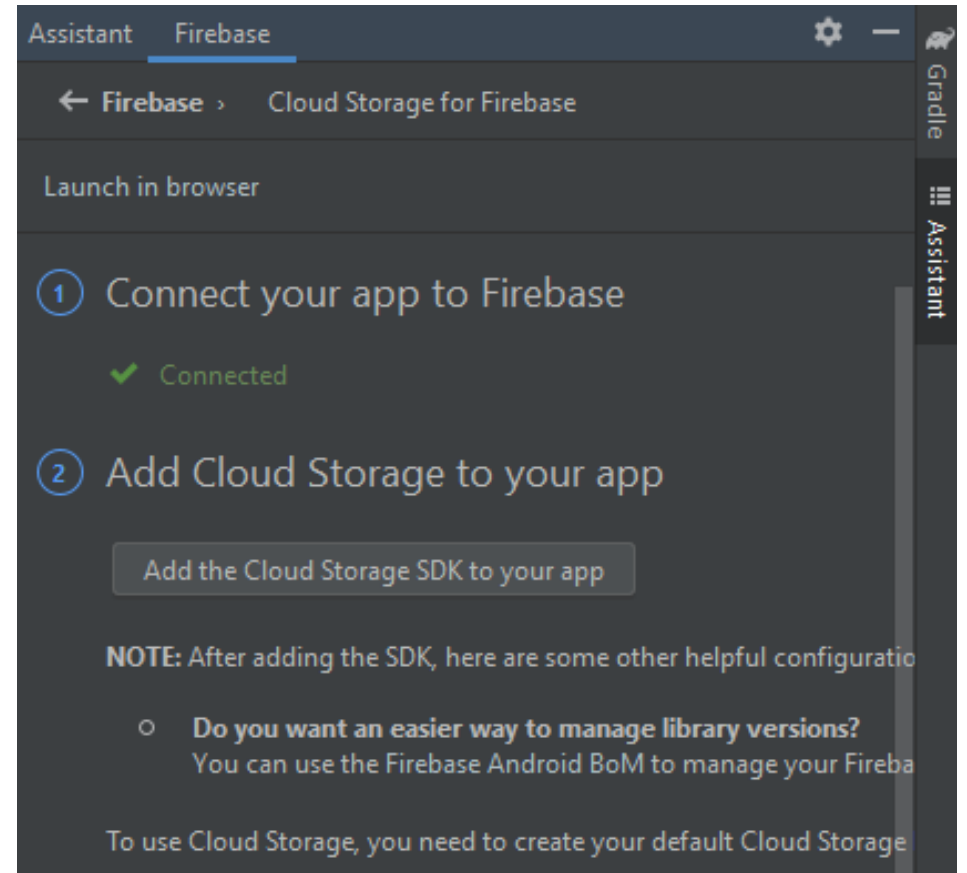
Seu app Android foi criado no Firebase.

Ele está pronto para ser conectado ao projeto do Android Studio.

[Conectar](#)

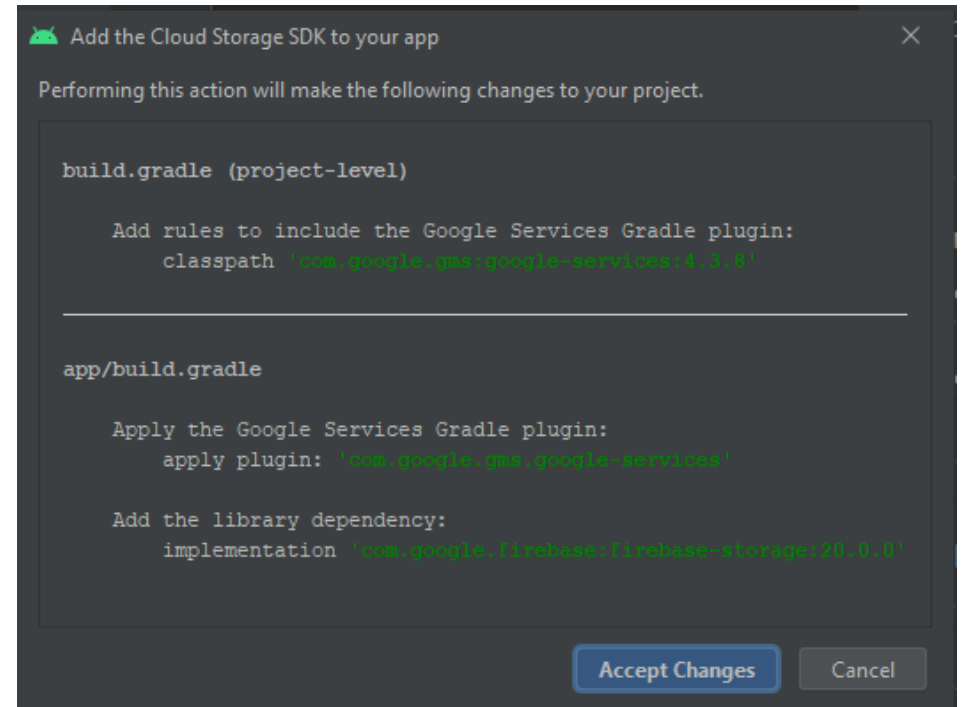
Configure o sdk

- Selecione a opção “add the cloud storage sdk to your app”

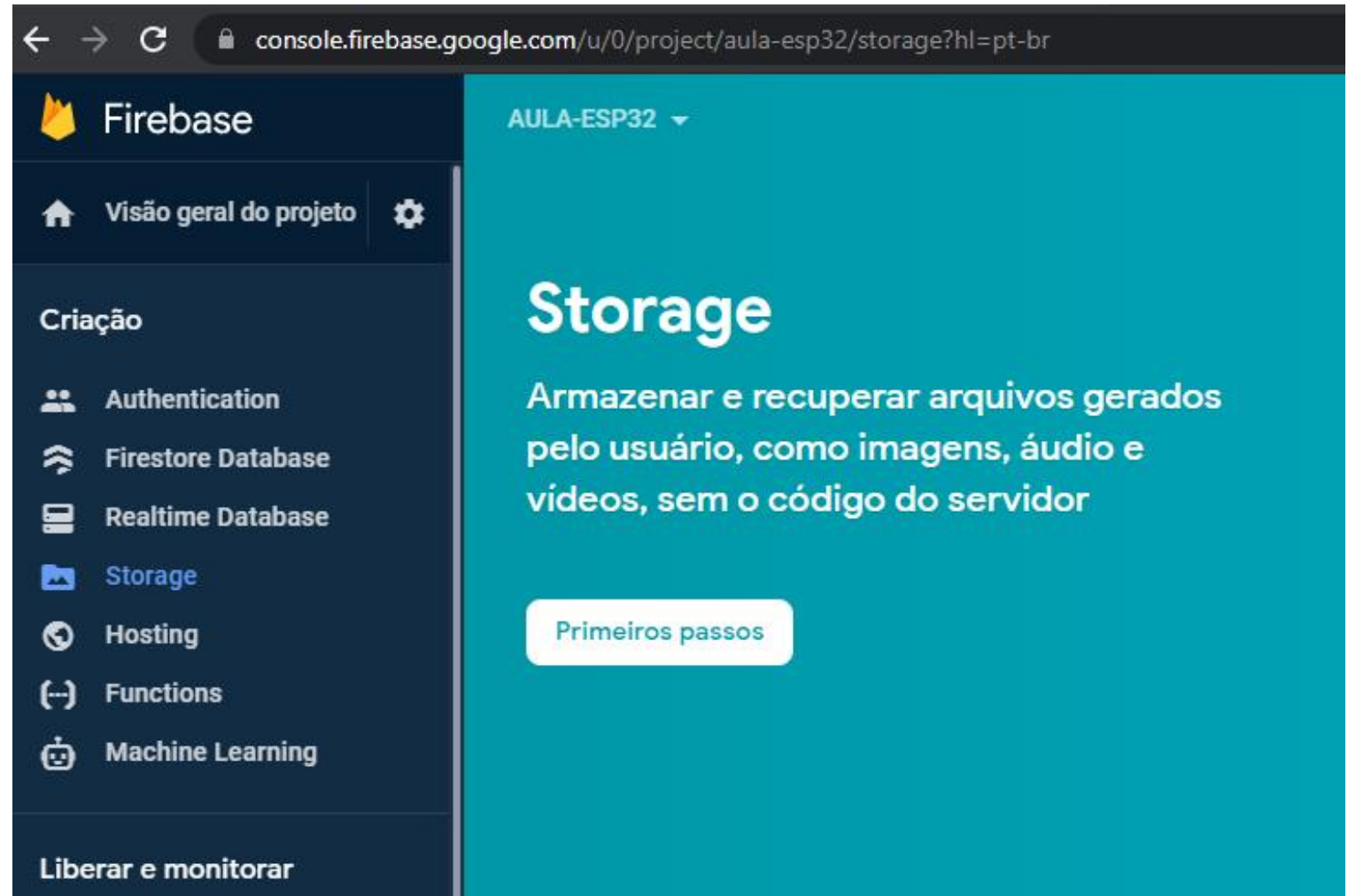


Aceite as mudanças

- Espere que o projeto seja configurado



Ative storage
no firebase



The screenshot shows the Firebase console interface for a project named "AULA-ESP32". The browser address bar displays the URL: `console.firebase.google.com/u/0/project/aula-esp32/storage?hl=pt-br`. The left sidebar contains the "Storage" option, which is highlighted in blue. The main content area features the heading "Storage" and a descriptive text: "Armazenar e recuperar arquivos gerados pelo usuário, como imagens, áudio e vídeos, sem o código do servidor". Below this text is a button labeled "Primeiros passos".

Configurações de acesso

Configurar o Cloud Storage

1 Regras seguras para o Cloud Storage

2 Definir local do Cloud Storage

Por padrão, suas regras permitem todas as leituras e gravações de usuários autenticados.

Após definir a estrutura de dados, será necessário criar regras para proteger seus dados. [Saiba mais](#)

```
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Cancelar

Próxima

Local do
servidor

Configurar o Cloud Storage



Regras seguras para o Cloud Storage



Definir local do Cloud Storage

A configuração do local é o lugar em que o bucket e os dados do Cloud Storage padrão serão armazenados.



Não será possível alterar o local depois de configurá-lo. Esta configuração de local também será o local padrão do Cloud Firestore.

[Saiba mais](#)

Local no Cloud Storage

southamerica-east1

Os clientes do plano Blaze podem selecionar outros locais para intervalos adicionais

Cancelar

Concluir

Arquivos
online

Storage

Files Rules Usage

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

[gs://aula-esp32.appspot.com](#) [Fazer upload do arquivo](#) + ⋮

<input type="checkbox"/>	Name	Tamanho	Tipo	Última modificação
Ainda não há arquivos aqui				

Regras de acesso

Editar regras Monitorar regras

alterações não publicadas **Publicar** Descartar



Proteja seus dados com regras que definem quem pode acessá-los e como eles são estruturados

[Ver os documentos](#)

```
1 rules_version = '2';
2 service firebase.storage {
3   match /b/{bucket}/o {
4     match /{allPaths=**} {
5       allow read, write: if request.auth != null;
6     }
7   }
8 }
9
```

Modifique
para testar

Storage

Files **Rules** Usage

Editar regras

Monitorar regras



Proteja seus dados com regras que definem quem pode acessá-los e como eles são estruturados

```
1 rules_version = '2';  
2 service firebase.storage {  
3   match /b/{bucket}/o {  
4     match /{allPaths=**} {  
5       allow read, write;  
6     }  
7   }  
8 }  
9
```

Atributos

```
class MainActivity : AppCompatActivity() {  
    private val PICK_IMAGE_REQUEST = 22  
    private var filePath: Uri? = null  
}
```

OnCreate

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    var btnSelect: Button = this.findViewById(R.id.btnEscolher)
    var btnUpload: Button = findViewById(R.id.BtnEnviar)
    var img: ImageView = findViewById(R.id.imgView)
    var storageReference = FirebaseStorage.getInstance().getReference()
    btnSelect.setOnClickListener(){
        selecionar()
    }
    btnUpload.setOnClickListener(){
        enviarImagem()
    }
}
```


seleccionar

```
private fun seleccionar() {  
    val intent = Intent()  
    intent.type = "image/*"  
    intent.action = Intent.ACTION_GET_CONTENT  
    startActivityForResult(Intent.createChooser(intent, "Selecione"), PICK_IMAGE_REQUEST)  
}
```

onActivityResult

```
override fun onActivityResult(requestCode: Int,resultCode: Int,data: Intent?) {
    super.onActivityResult(
        requestCode,
        resultCode,
        data
    )
    if (requestCode == PICK_IMAGE_REQUEST && resultCode == Activity.RESULT_OK && data != null && data.data != null) {
        filePath = data.data
        try {
            val bitmap = MediaStore.Images.Media.getBitmap(contentResolver,filePath)
            var img: ImageView = findViewById(R.id.imageView)
            img.setImageBitmap(bitmap)
        } catch (e: IOException) {
            e.printStackTrace()
        }
    }
}
```

enviarImagem

```
private fun enviarImagem() {
    if (filePath != null) {
        val progressDialog = ProgressDialog(this)
        progressDialog.setTitle("Enviando...")
        progressDialog.show()
        val ref: StorageReference = FirebaseStorage.getInstance().getReference()
            .child("images/" + UUID.randomUUID().toString())

        ref.putFile(filePath!!).addOnSuccessListener {
            progressDialog.dismiss()
            Toast.makeText(this@MainActivity, "Imagem enviada", Toast.LENGTH_SHORT).show()
        }
        .addOnFailureListener { e -> progressDialog.dismiss()
            Toast.makeText(this@MainActivity, "Erro ao enviar " + e.message, Toast.LENGTH_SHORT).show()
        }
        .addOnProgressListener { taskSnapshot ->
            val progress = (100.0 * taskSnapshot.bytesTransferred / taskSnapshot.totalByteCount)
            progressDialog.setMessage("Enviando " + progress.toInt() + "%")
        }
    }
}
```