

# Azure Function

## I. Instalando a extensão Azure Functions Core Tools
























A primeira coisa a ser feito é verificar se já existe no computador (Servidor) a ferramenta para testar as Azure Functions na máquina local.

Nome da ferramenta: **azure-functions-core-tools**

Caso não esteja instalado, abaixo está o link para baixá-lo em conformidade com o sistema operacional.

Link: <https://github.com/Azure/azure-functions-core-tools/releases>

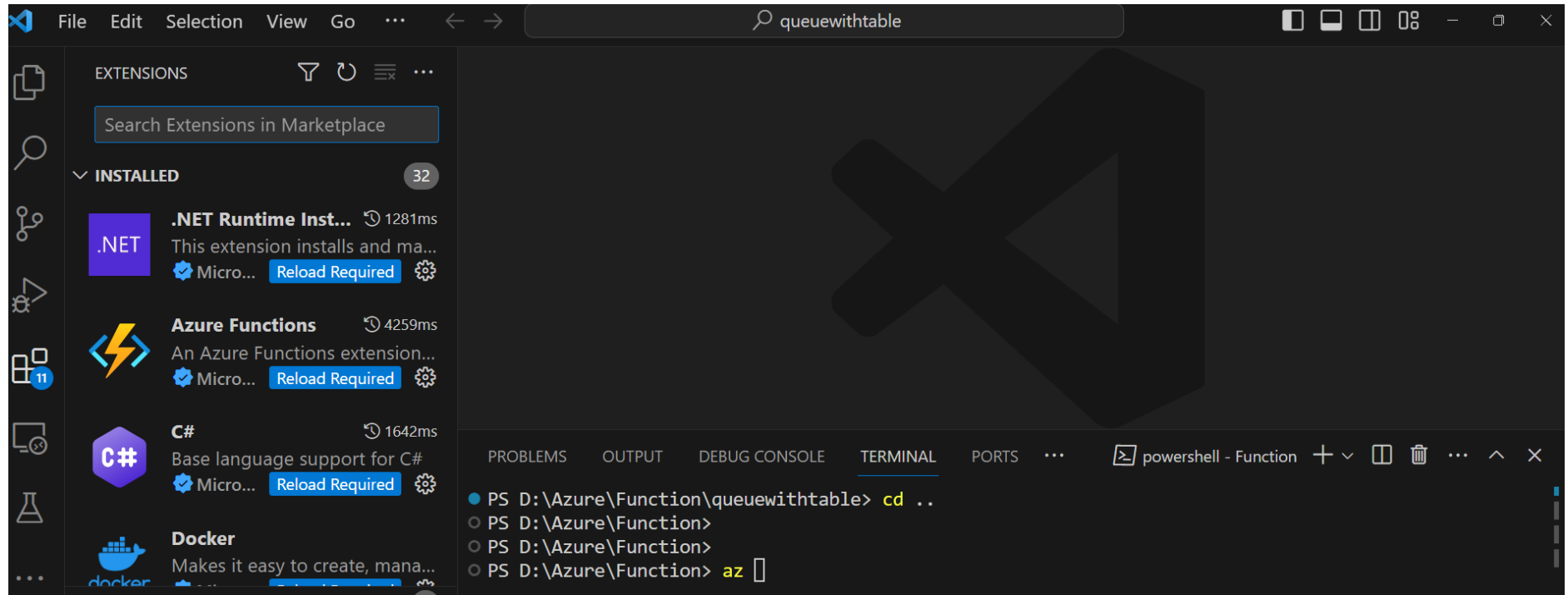
### ▼ Assets 23

 Azure.Functions.Cli.linux-x64.4.0.5390.zip	200 MB	last week
 Azure.Functions.Cli.linux-x64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.min.win-arm64.4.0.5390.zip	60.9 MB	last week
 Azure.Functions.Cli.min.win-arm64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.min.win-x64.4.0.5390.zip	64.3 MB	last week
 Azure.Functions.Cli.min.win-x64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.min.win-x86.4.0.5390.zip	60.8 MB	last week
 Azure.Functions.Cli.min.win-x86.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.osx-arm64.4.0.5390.zip	195 MB	last week
 Azure.Functions.Cli.osx-arm64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.osx-x64.4.0.5390.zip	201 MB	last week
 Azure.Functions.Cli.osx-x64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.win-arm64.4.0.5390.zip	234 MB	last week
 Azure.Functions.Cli.win-arm64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.win-x64.4.0.5390.zip	238 MB	last week
 Azure.Functions.Cli.win-x64.4.0.5390.zip.sha2	64 Bytes	last week
 Azure.Functions.Cli.win-x86.4.0.5390.zip	234 MB	last week
 Azure.Functions.Cli.win-x86.4.0.5390.zip.sha2	64 Bytes	last week
 func-cli-4.0.5390-x64.msi	207 MB	last week
 func-cli-4.0.5390-x86.msi	204 MB	last week
 Microsoft.Azure.Functions.CoreTools.4.0.5390.nupkg	25.2 MB	last week
 Source code (zip)		last week
 Source code (tar.gz)		last week

# Azure Function

## II. Instalando a extensão Azure Function no Visual Code

Antes de mais nada, precisamos instalar a extensão do Azure Function no Visual Code.



Como podemos verificar na imagem acima, a extensão já se encontra instalada, mas se não tiver, é só pesquisar por essa extensão e quando aparecer, solicitar a instalação dela dentro do Visual Code.

## Azure Function

### III. Criando o ambiente no anaconda para este trabalho

Antes de executar a criação de uma Azure Function, criar um ambiente (env) no anaconda.

`conda create --name {nome do ambiente da sua preferência} {python==3.8}` --> Não é necessário, mas querendo, verificar a versão da sua preferência.

`conda activate {nome_do_seu_ambiente}` → Após, ativar o ambiente.

Você pode instalar pacotes adicionais no seu ambiente virtual usando o Conda. Por exemplo:

`conda install nome_do_pacote`

```
5 from urllib3.exceptions import InsecureRequestWarning
6 import urllib3
7 import uuid
8 import json
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   AZURE: ACTIVITY LOG

#	
base	C:\ProgramData\anaconda3
FunctionAppCosmosDBTrigger	C:\ProgramData\anaconda3\envs\FunctionAppCosmosDBTrigger
CosmosDB	C:\Users\sid-j\.conda\envs\CosmosDB
EnvioArqBlobAzure	C:\Users\sid-j\.conda\envs\EnvioArqBlobAzure
ProjetoETL	C:\Users\sid-j\.conda\envs\ProjetoETL
Projeto_ETL	C:\Users\sid-j\.conda\envs\Projeto_ETL
Treino_Python	C:\Users\sid-j\.conda\envs\Treino_Python
dags	C:\Users\sid-j\.conda\envs\dags

○ (venv) PS D:\EstudosCasa\Python\Azure\Function\CosmosDBTrigger> █

## Azure Function

Caso necessário, instalar os pacotes necessários dentro do seu ambiente que você acabou de criar.

Por exemplo:

```
azure-functions  
  
import logging  
  
import azure.functions as func  
  
  
import azure.cosmos.cosmos_client as cosmos_client  
from urllib3.exceptions import InsecureRequestWarning  
  
import urllib3  
  
import uuid  
  
import json
```

pacote para você tanto ter os dados que vem do CosmosDB como para criar documento no CosmosDB

## Azure Function

### IV. Criando a estrutura de uma Azure Function Localmente

Abra o Visual Code e dentro dele abra o terminal.

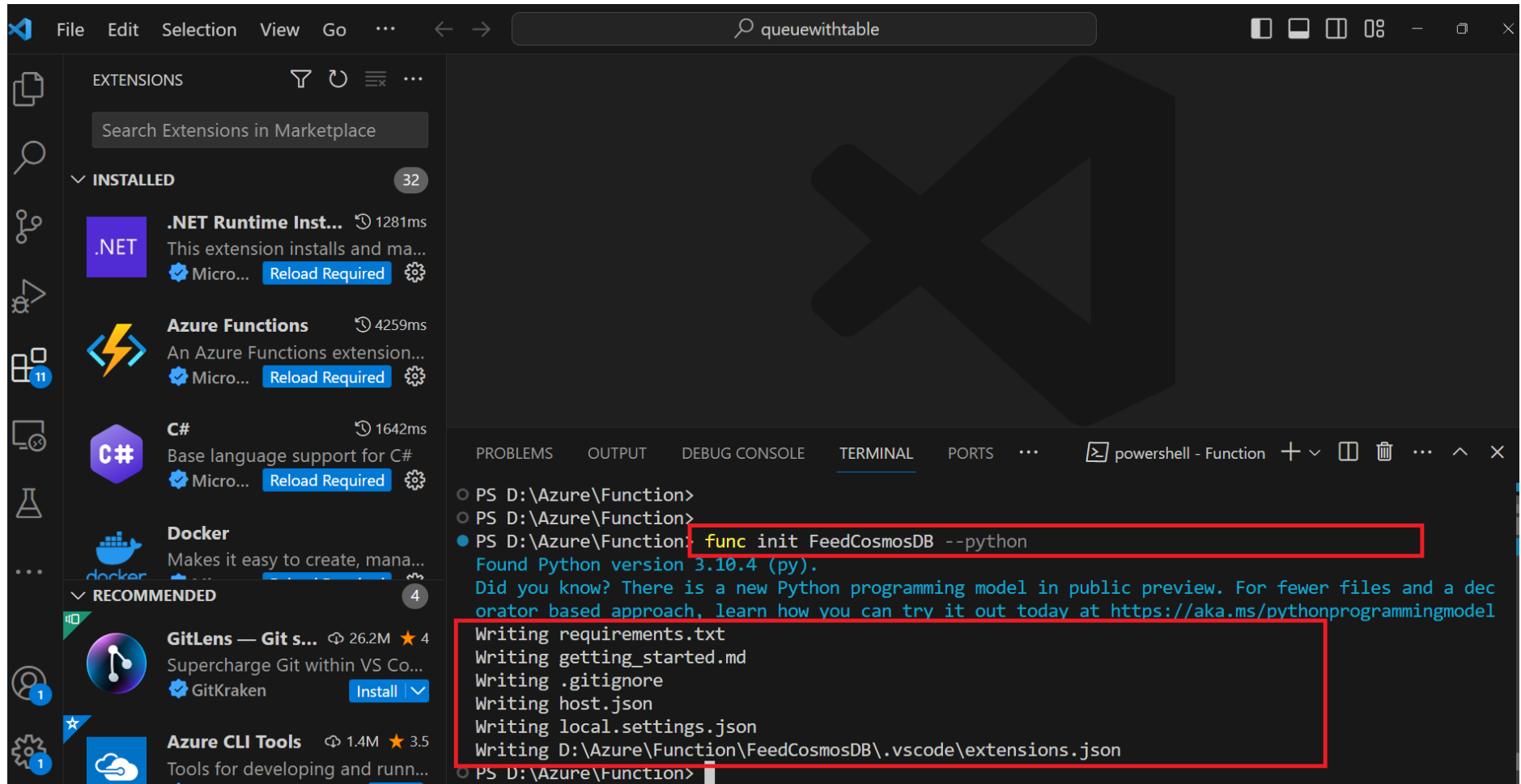


Dentro do terminal cria ou vai até o diretório do seu projeto Azure Function (Esta pasta você pode criar via cmd do VSCode).

Neste exemplo, estou dentro do diretório “D:\Azure\Function”, onde eu irei criar uma aplicação de Azure Function.

## Azure Function

Estando já no diretório onde você deseja criar a Azure Function, agora é digitar o comando em destaque na imagem abaixo.

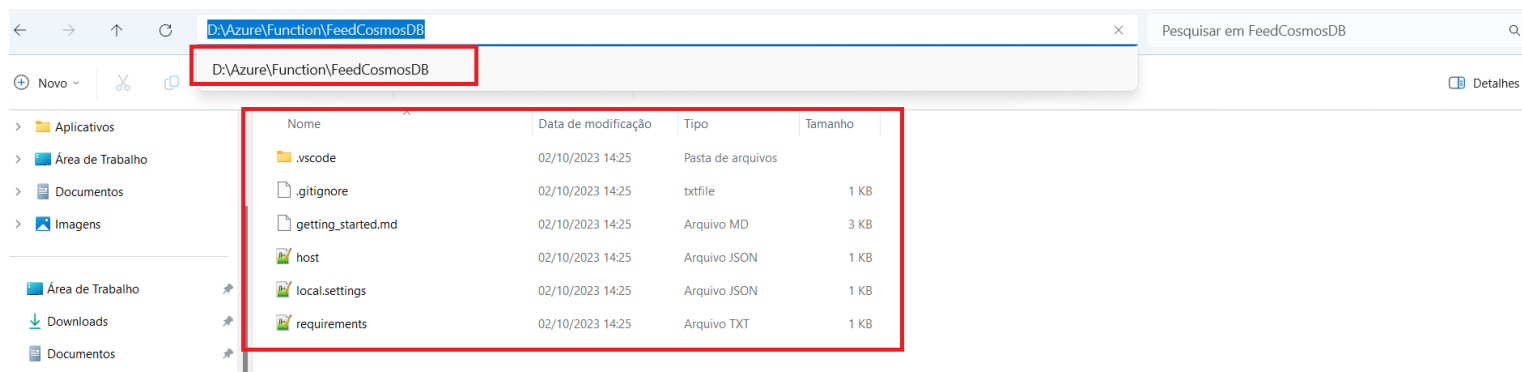


Comando: `func init {nome do projeto} {tipo da linguagem de programação que você quer desenvolver}`, neste caso estou desenvolvendo em Python.

Após, ele irá criar alguns arquivos dependentes para o funcionamento desta Function.

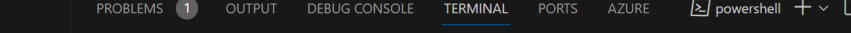
Estes arquivos estão no 2º destaque desta imagem acima.

## Azure Function



## V. Criando Azure Function

Comando para criar um tipo de Azure Function `{func new}`



The screenshot shows a PowerShell terminal window with the following content:

```
PS D:\Azure\Function\FeedCosmosDB> func new
Use the up/down arrow keys to select a template:
Azure Blob Storage trigger
Azure Cosmos DB trigger
Durable Functions activity
Durable Functions entity
Durable Functions HTTP starter
Durable Functions orchestrator
Azure Event Grid trigger
```

The terminal window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (active), PORTS, and AZURE. The PowerShell icon and name are visible in the title bar.

Ao executar este comando acima, a extensão do o Azure Function apresentará uma lista de opções de eventos que podem ser criados na Azure Function.

## Azure Function

Lista:

1. Azure Blob Storage trigger
2. Azure Cosmos DB trigger
3. Durable Functions activity
4. Durable Functions entity
5. Durable Functions HTTP starter
6. Durable Functions orchestrator
7. Azure Event Grid trigger
8. Azure Event Hub trigger
9. HTTP trigger
10. Kafka output
11. Kafka trigger
12. Azure Queue Storage trigger
13. RabbitMQ trigger
14. Azure Service Bus Queue trigger
15. Azure Service Bus Topic trigger
16. Timer trigger

### 1. Azure Cosmos DB trigger

Para este tipo, você deve escolher o número 2.

1. Azure Blob Storage trigger
  2. Azure Cosmos DB trigger
  3. Durable Functions activity
  4. Durable Functions entity
  5. Durable Functions HTTP starter
  6. Durable Functions orchestrator
  7. Azure Event Grid trigger
  8. Azure Event Hub trigger
  9. HTTP trigger
  10. Kafka output
  11. Kafka trigger
  12. Azure Queue Storage trigger
  13. RabbitMQ trigger
  14. Azure Service Bus Queue trigger
  15. Azure Service Bus Topic trigger
  16. Timer trigger
- Choose option: 2

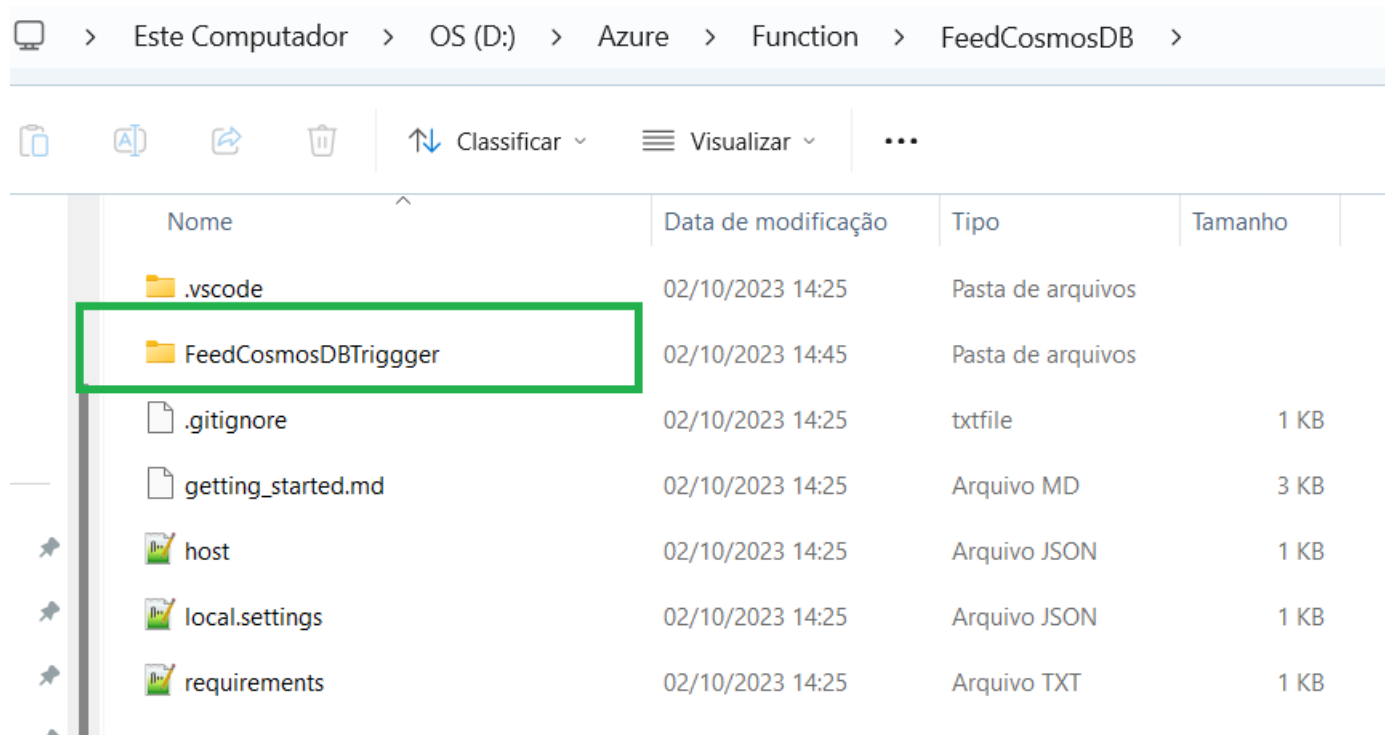
Após, será solicitado que você dê o nome para esta Function.



## Azure Function



```
Azure Cosmos DB trigger
Function name: [CosmosTrigger] FeedCosmosDBTriggerger ←
Writing D:\Azure\Function\FeedCosmosDB\FeedCosmosDBTriggerger\__init__.py
Writing D:\Azure\Function\FeedCosmosDB\FeedCosmosDBTriggerger\function.json
The function "FeedCosmosDBTriggerger" was created successfully from the "Azure Cosmos DB trigger" te
mplate.
Did you know? There is a new Python programming model in public preview. For fewer files and a dec
orator based approach, learn how you can try it out today at https://aka.ms/pythonprogrammingmodel
○ Writing D:\Azure\Function\FeedCosmosDB\FeedCosmosDBTriggerger\function.json
○ The function "FeedCosmosDBTriggerger" was created successfully from the "Azure Cosmos DB trigger" te
○ mplate.
Did you know? There is a new Python programming model in public preview. For fewer files and a dec
orator based approach, learn how you can try it out today at https://aka.ms/pythonprogrammingmodel
```

No diretório do projeto Azure Function que você criou anteriormente será criado uma subpasta com o nome que você acabou de digitar no terminal acima.

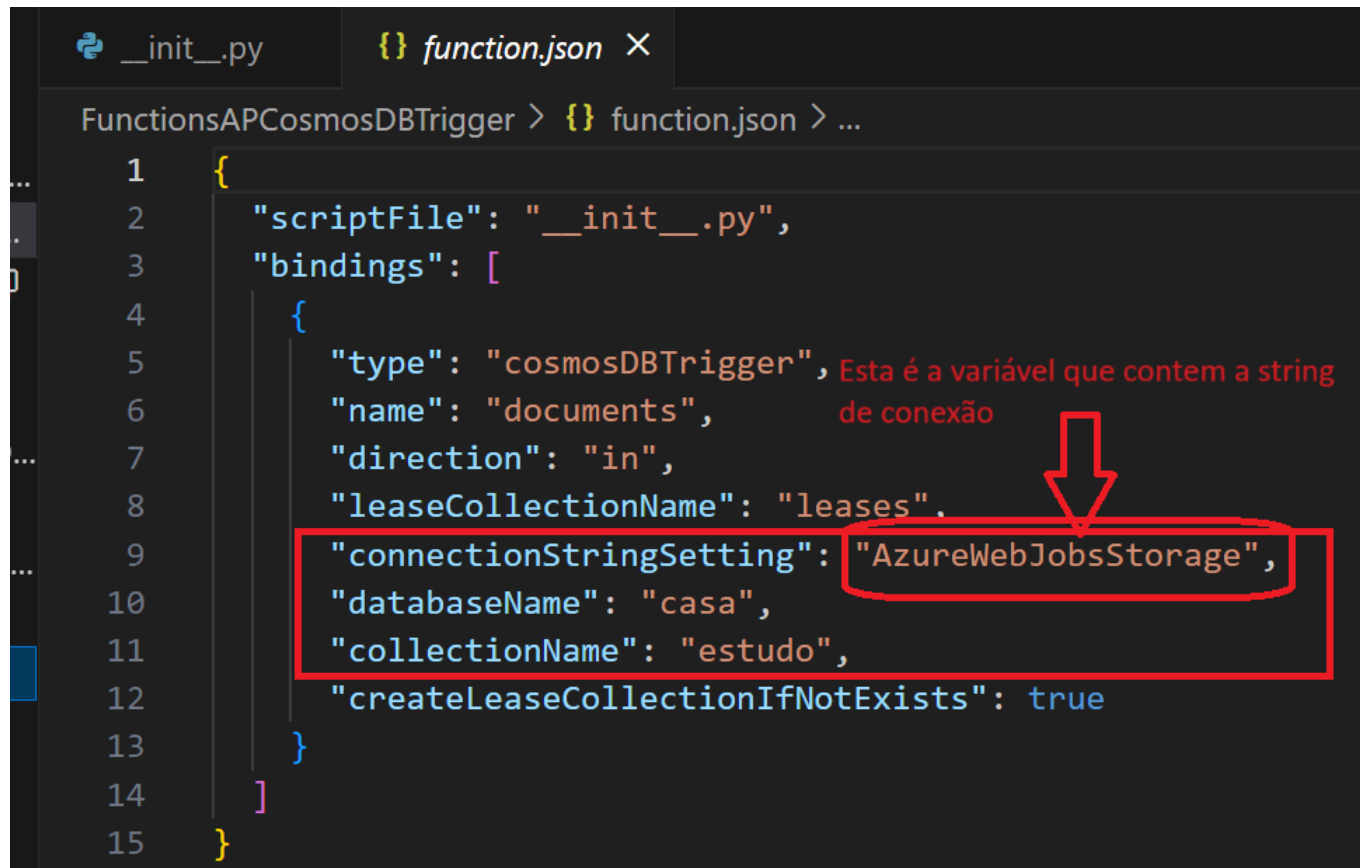


## Azure Function

Com estes arquivos.

Nome	Data de modificação	Tipo	Tamanho
 __init__	02/10/2023 14:45	JetBrains PyCharm ...	1 KB
 function	02/10/2023 14:50	Arquivo JSON	1 KB

Dentro do arquivo Function.json, você irá encontrar os parâmetros de conexão com o seu banco CosmosDB.



The screenshot shows the VS Code editor with the `function.json` file open. The file content is as follows:

```
1 {
2   "scriptFile": "__init__.py",
3   "bindings": [
4     {
5       "type": "cosmosDBTrigger",
6       "name": "documents",
7       "direction": "in",
8       "leaseCollectionName": "leases",
9       "connectionStringSetting": "AzureWebJobsStorage",
10      "databaseName": "casa",
11      "collectionName": "estudo",
12      "createLeaseCollectionIfNotExists": true
13    }
14  ]
15 }
```

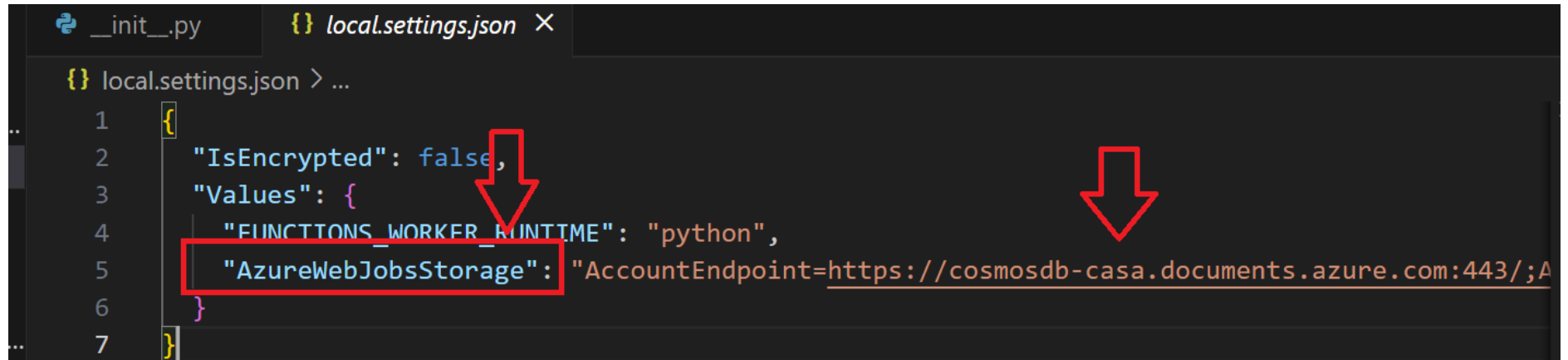
Annotations in the image include:

- A red arrow pointing to the `"connectionStringSetting": "AzureWebJobsStorage",` line, with the text: "Esta é a variável que contém a string de conexão".
- A red rectangle highlighting the entire binding object from line 9 to line 12.

## Azure Function

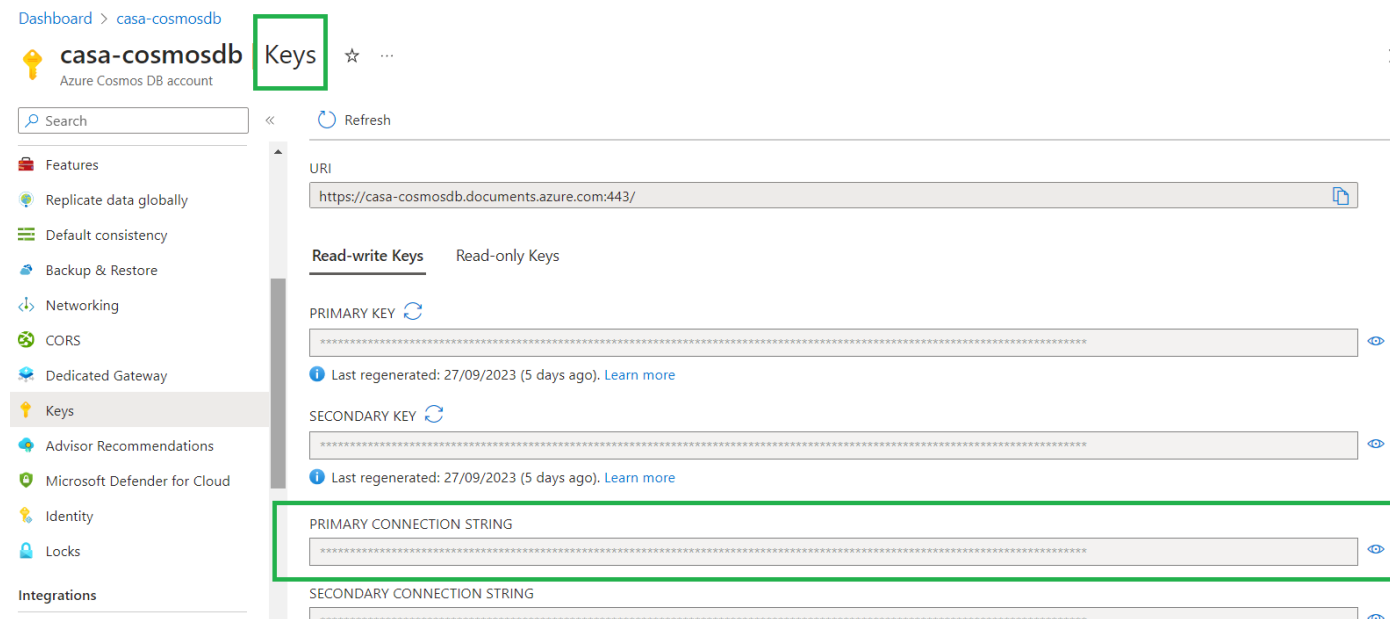
Onde você irá colocar:

**connectionStringSetting** = A variável que vem do arquivo **local.Settings.json**



```
{
  "IsEncrypted": false,
  "Values": {
    "FUNCTIONS_WORKER_RUNTIME": "python",
    "AzureWebJobsStorage": "AccountEndpoint=https://cosmosdb-casa.documents.azure.com:443/;A"
  }
}
```

Conexão que você pega no painel de configuração do seu CosmosDb criado no Azure.



Dashboard > casa-cosmosdb

**Keys** ☆ ...

Search Refresh

URI  
https://casa-cosmosdb.documents.azure.com:443/

Read-write Keys Read-only Keys

PRIMARY KEY  
Last regenerated: 27/09/2023 (5 days ago). [Learn more](#)

SECONDARY KEY  
Last regenerated: 27/09/2023 (5 days ago). [Learn more](#)

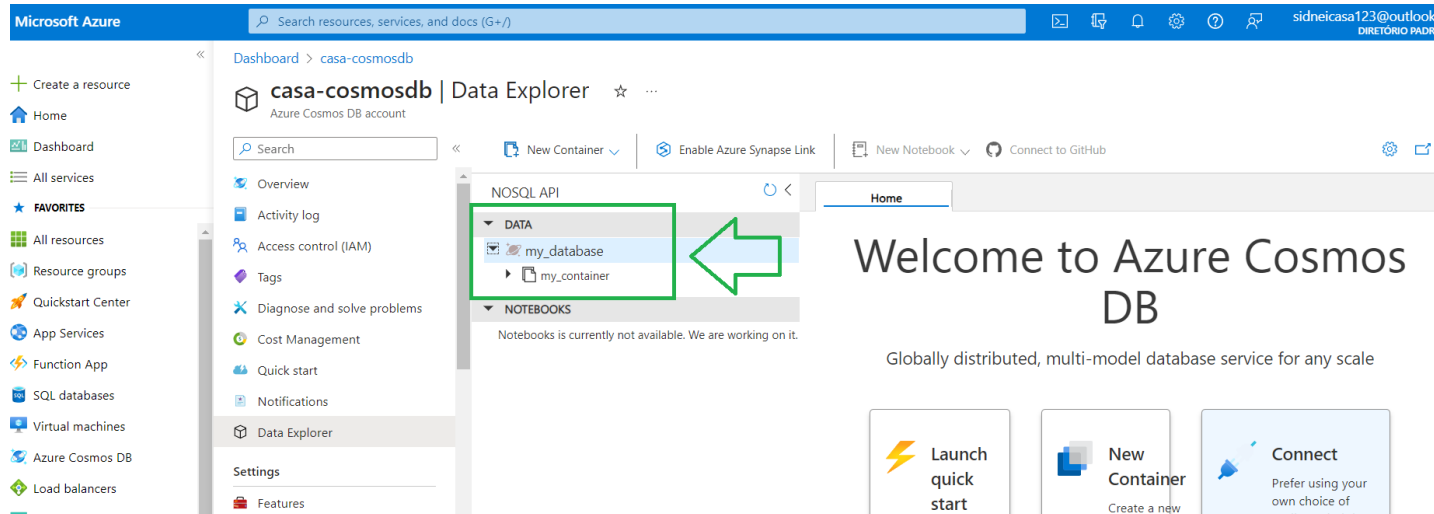
PRIMARY CONNECTION STRING

SECONDARY CONNECTION STRING

## Azure Function

**databaseName** = O nome da base criada no CosmosDB

**collectionName** = O nome do seu container base criada no CosmosDB



## Azure Function

### VI. Executando o projeto Azure Function

Após as devidas configurações, agora é executar a function.

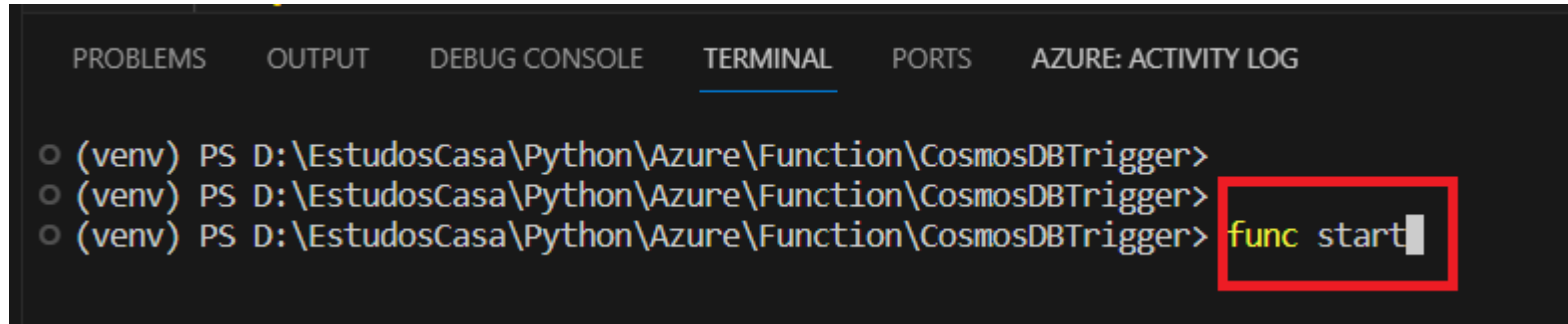
O arquivo é o `__init__.py`, ele é o arquivo man da function.

```
__init__.py • {} local.settings.json
FunctionsAPCosmosDBTrigger > __init__.py > main
1  import logging
2  import azure.functions as func
3
4  import azure.cosmos.cosmos_client as cosmos_client
5  from urllib3.exceptions import InsecureRequestWarning
6  import urllib3
7  import uuid
8  import json
9
10 url = "https://cosmosdb-casa.documents.azure.com:443/"
11 key = "V4mulwX1m5bvWASsx6nMhgsGmLprTIWiffjUIY9JfYtHoKVZwHiXENuMbRuj2MDSeCCrBRP2K0XNAACDbYT81XA=="
12 database_name = "casa"
13 container_name = "estudo2"
14
15 def registrar():
16     client = cosmos_client.CosmosClient(url, key)
17     database = client.get_database_client(database_name)
18     container = database.get_container_client(container_name)
19
20     data = {
21         "id": "3.0",
22         "nome": "Sidnei Lima Santos",
23         "idade": 40,
24         "cidade": "São Paulo"
25     }
26
27     result = container.upsert_item(data)
28     logging.info(result)
29
30 def main(documents: func.DocumentList, context: func.Context) -> str:
31     if "deleted" in documents or documents[0]["deleted"]:
32         logging.info('Document id: %s', documents[0]['id'])
33         nome = documents[0]["nome"]
34         logging.info('Nome é: %s', nome)
35     registrar()
```

\*Neste código eu estou registrando os dados obtidos da alteração do documento do container estudo para o container estudo2.

## Azure Function

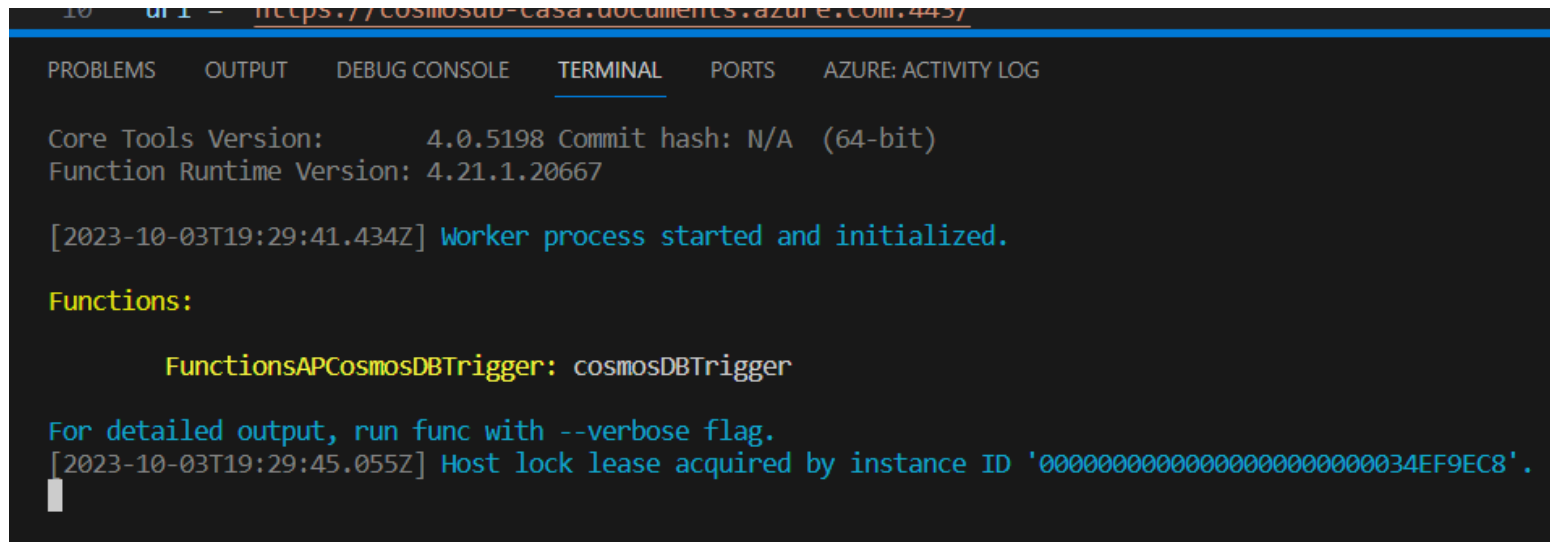
Para executar uma Azure Function, melhor executar via cmd, desta forma:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE: ACTIVITY LOG

○ (venv) PS D:\EstudosCasa\Python\Azure\Function\CosmosDBTrigger>
○ (venv) PS D:\EstudosCasa\Python\Azure\Function\CosmosDBTrigger>
○ (venv) PS D:\EstudosCasa\Python\Azure\Function\CosmosDBTrigger> func start
```

onde o mesmo será executado, não dando erro, o mesmo ficará desta forma:



```
10 url = https://cosmosdb-casa.documents.azure.com:443/

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE: ACTIVITY LOG

Core Tools Version:      4.0.5198 Commit hash: N/A (64-bit)
Function Runtime Version: 4.21.1.20667

[2023-10-03T19:29:41.434Z] Worker process started and initialized.

Functions:

    FunctionsAPCosmosDBTrigger: cosmosDBTrigger

For detailed output, run func with --verbose flag.
[2023-10-03T19:29:45.055Z] Host lock lease acquired by instance ID '00000000000000000000000034EF9EC8'.
```

Para realizar o stop, é só clicar em **CTRL + C** dentro do terminal, este comando irá interromper a execução.

## Azure Function

### VII. Publicação da Azure Function para o portal da Azure

Agora que finalizamos o nosso projeto de Azure Function localmente, vamos publicá-lo no portal do Azure para que o mesmo possa executar automaticamente pelo serviço da Azure Cloud.

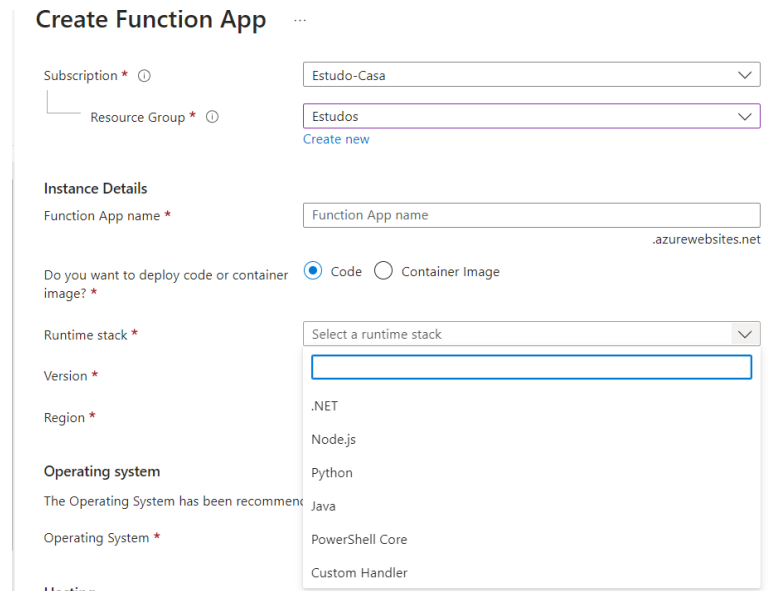
Passo a passo para a realização da publicação.

1. Criar uma function App no portal da Azure Cloud.
  - 1.1. Entrar no portal e escolher o serviço Function App



Function App

#### 1.2. Basic

The screenshot shows the 'Create Function App' form in the Azure portal. The form is titled 'Create Function App' with a three-dot menu icon to its right. It contains several sections: 'Subscription' with a dropdown menu showing 'Estudo-Casa'; 'Resource Group' with a dropdown menu showing 'Estudos' and a 'Create new' link; 'Instance Details' with a 'Function App name' text box and a '.azurewebsites.net' domain; 'Do you want to deploy code or container image?' with radio buttons for 'Code' (selected) and 'Container Image'; 'Runtime stack' with a dropdown menu showing 'Select a runtime stack'; 'Version' with a text box; 'Region' with a dropdown menu; 'Operating system' with a text box and a note 'The Operating System has been recommended'; and 'Operating System' with a dropdown menu showing 'PowerShell Core' and 'Custom Handler'. The form is partially obscured by a blue bar at the bottom.

## Azure Function

Subscription = Sua subscrição

Resource Group = Seu grupo de recurso

Function App Name = Criar um nome para sua Function App

Do you want to deploy code or container images? = Aqui você pode escolher entre código ou container. (Normalmente é código)

Runtime stack = Aqui você selecionará a linguagem de programação desse repositório de Functions. (Cada repositório tem uma única ligação atrelada a ela)

Version = Aqui você escolherá a versão do pacote de programação da sua preferência.

Region = Aqui você escolherá a região da sua preferência. (Escolha a região que está o seu Blob Storage caso exista, assim, a Function App usará esse repositório para registrar o seus logs.)

Operating System = Escolha o sistema operacional que a suas functions irá rodar. (Escolha sempre que possível o sistema operacional Linux, menos custo na hora de executar)

Hosting options and plans = Escolha “ Consumption (Serverless)” Optimized for serverless and event-driven workloads.

### 1.3. Storage

#### Create Function App ...

Basics Storage Networking Monitoring Deployment Tags Review + create

##### Storage

When creating a function app, you must create or link to a general-purpose Azure Storage account that supports Blobs, Queue, and Table storage.

Storage account \*

(New) estudos8257

[Create new](#)

Aqui, se você escolheu a mesma região que está a sua Storage, então ela irá aparecer para vc vinculá-la a sua Function App.

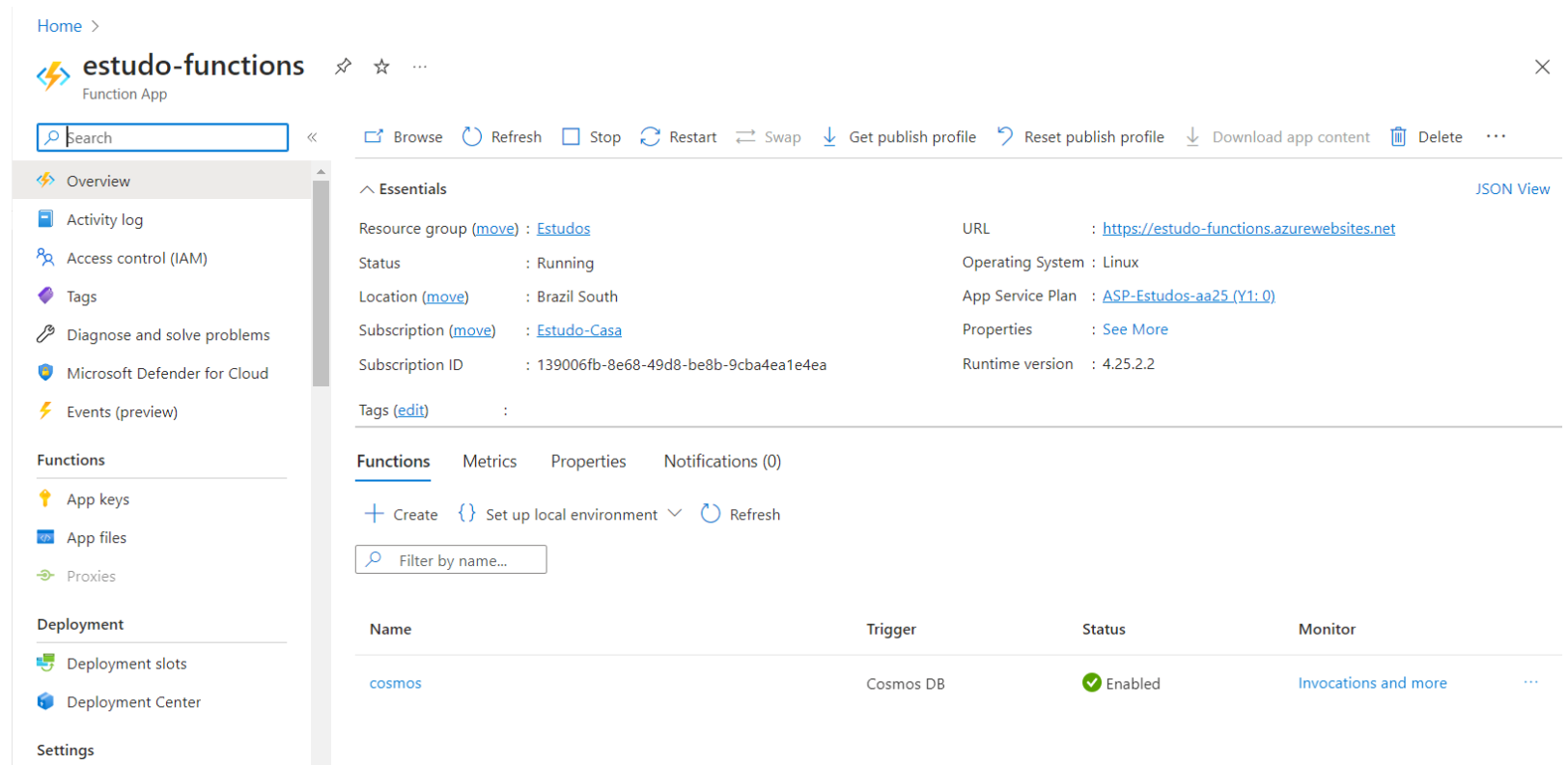
Feito isso, vai até o fim para criá-la.



# Azure Function

## 1.4. Sua Function App após ser criada

Ao final da criação você deverá ter esse ambiente.




Na imagem acima, eu já tenho uma Azure Function publicada.(Essa function está construída em Python)

**Obs.:** O nome da função principal de dentro do seu código Azure Function será o nome da Azure Function no portal da Azure Cloud.

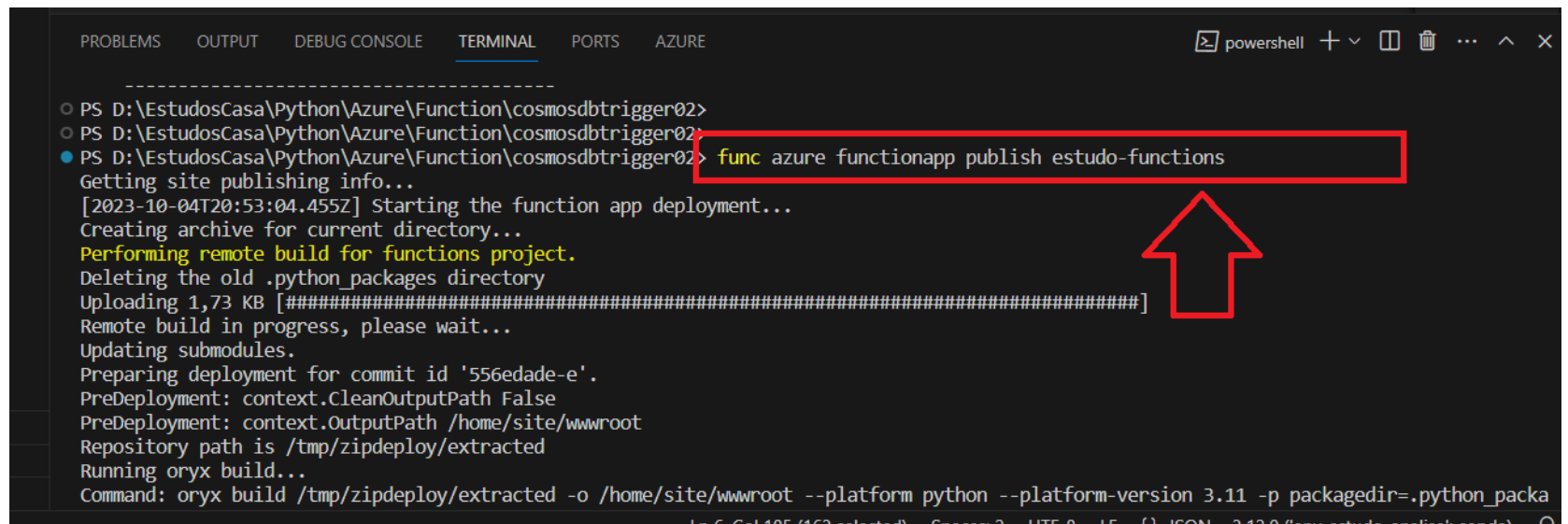
O nome na imagem acima é Cosmos, certo? Porquê?

## Azure Function

```
@app.cosmos_db_trigger(arg_name="azcosmosdb", container_name="estudo",  
                        database_name="casa", connection="conncosmosdb")  
  
def cosmos(azcosmosdb: func.DocumentList):
```



### 1.5. Como Subir a Azure Function criada localmente?

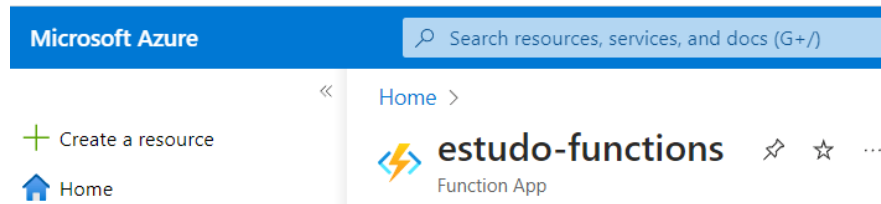


```
-----  
PS D:\EstudosCasa\Python\Azure\Function\cosmosdbtrigger02>  
PS D:\EstudosCasa\Python\Azure\Function\cosmosdbtrigger02>  
PS D:\EstudosCasa\Python\Azure\Function\cosmosdbtrigger02> func azure functionapp publish estudo-functions  
Getting site publishing info...  
[2023-10-04T20:53:04.455Z] Starting the function app deployment...  
Creating archive for current directory...  
Performing remote build for functions project.  
Deleting the old .python_packages directory  
Uploading 1,73 KB [#####]  
Remote build in progress, please wait...  
Updating submodules.  
Preparing deployment for commit id '556edade-e'.  
PreDeployment: context.CleanOutputPath False  
PreDeployment: context.OutputPath /home/site/wwwroot  
Repository path is /tmp/zipdeploy/extracted  
Running oryx build...  
Command: oryx build /tmp/zipdeploy/extracted -o /home/site/wwwroot --platform python --platform-version 3.11 -p packagedir=.python_packa
```

Comando: `func azure functionapp publish estudo-functions`

`{estudo-function}` é o nome da Function App criada no portal da Azure Cloud.

## Azure Function



### Atenção:

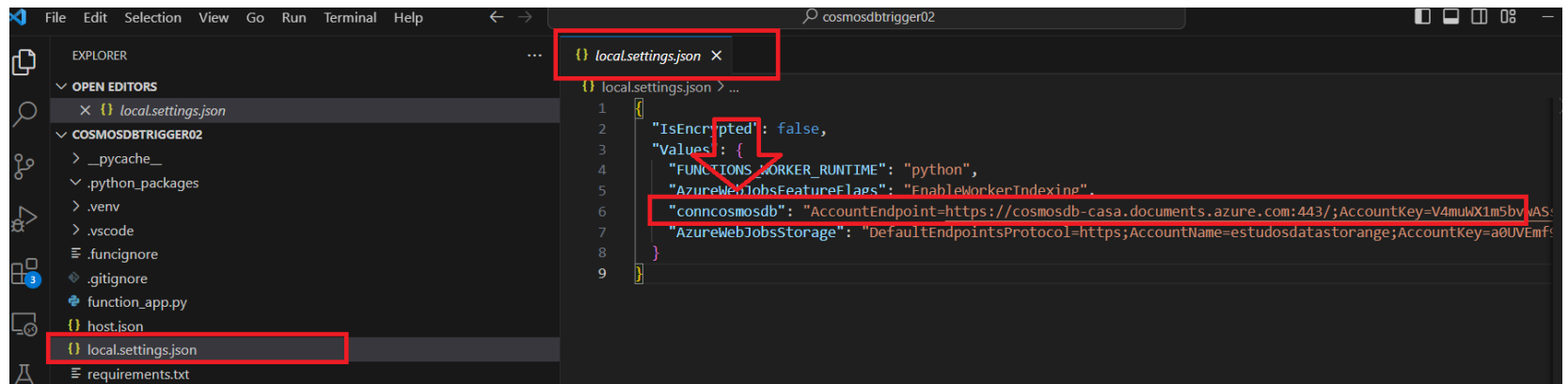
Todos os pacotes que você está utilizando na sua Azure Function, precisam estar registrada no arquivo **requirements.txt**

A screenshot of a code editor with a dark background. At the top, there are two tabs: 'function\_app.py' and 'requirements.txt'. The 'requirements.txt' tab is active. The code in the editor is as follows:

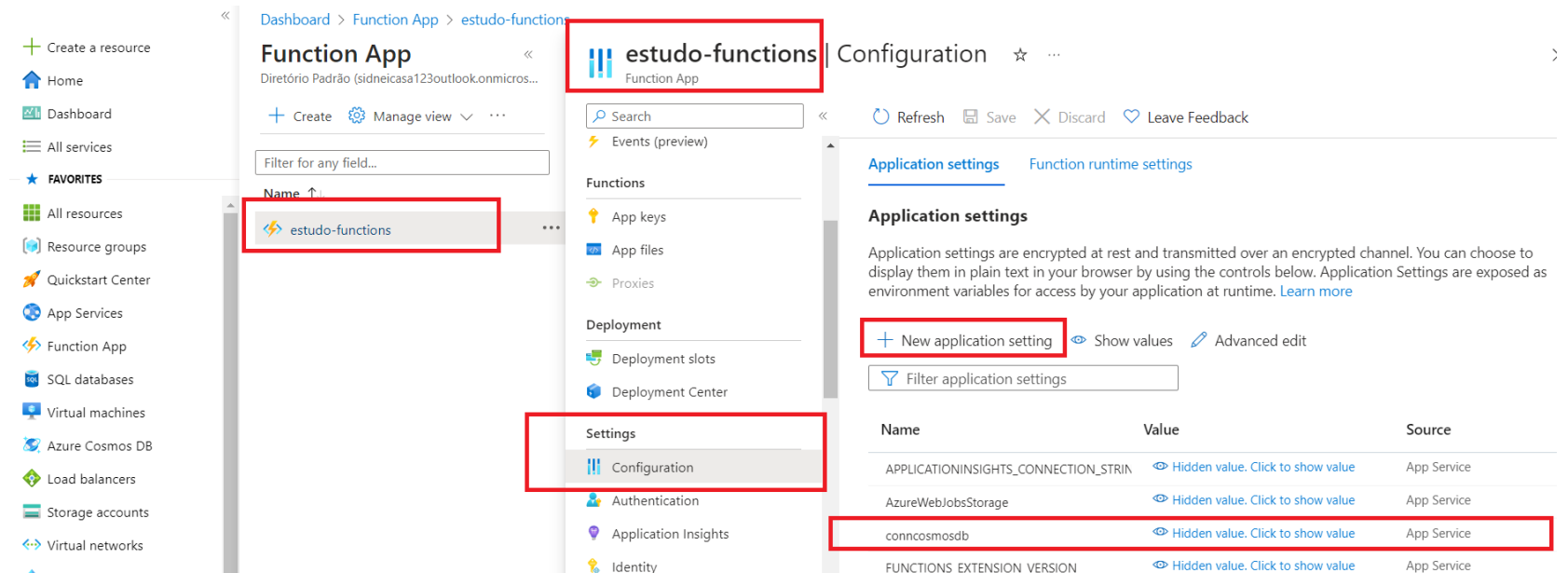
```
1 # Do not include azure-functions-worker in this file
2 # The Python Worker is managed by the Azure Functions platform
3 # Manually managing azure-functions-worker may cause unexpected issues
4
5 azure-functions
6 azure-cosmos
```

A CONEXÃO COM O BANCO, COSMOS DB ETC... DEVEM SER REGISTRADA NA MÃO NO PORTAL DA AZURE CLOUD EM CONFIGURAÇÃO, NA HORA DA PUBLICAÇÃO, ESTAS INFO DE CONEXÃO NÃO SÃO CRIADAS AUTOMATICAMENTE POR SEGURANÇA.

## Azure Function



VOCÊ PRECISA CONFIGURAR NO PORTAL EM CONFIGURATION.



## Azure Function

Estando tudo certo, ao final da construção da Azure Function no portal da Azure Cloud, você deve receber essa mensagem no final do processo.

```
root@0)
Creating placeholder blob for linux consumption function app...
SCM_RUN_FROM_PACKAGE placeholder blob scm-latest-estudo-functions.zip located
Uploading built content /home/site/artifacts/functionappartifact.squashfs for linux consumption function app...
Resetting all workers for estudo-functions.azurewebsites.net
Deployment successful. deployer = Push-Deployer deploymentPath = Functions App ZipDeploy. Extract zip. Remote build.
Remote build succeeded!
[2023-10-04T20:53:39.087Z] Syncing triggers...
Functions in estudo-functions:
  cosmos - [cosmosDBTrigger]

PS D:\EstudosCasa\Python\Azure\Function\cosmosdbtrigger02>
```

Qualquer problema, analise o motivo.

### 1.6. Processamento automático no portal da Azure Cloud

Sempre que for criada a Azure Function no portal da Azure Cloud, ao ser criada, ele já imediatamente fica no estado de habilitada, e a Function App fica em Running, ou seja em execução.

Ou seja, assim que subir, é só realizar a ação devida para analisar o seu processamento.

#### Atenção:

O resultado no Monitor demora de aparecer, em média, por volta de 5 minutos.

## Azure Function

The screenshot shows the 'cosmos | Monitor' interface for a function. The left sidebar contains navigation links: Overview, Developer, Code + Test, Integration, Monitor (selected), and Function Keys. The main content area has tabs for 'Invocations' and 'Logs'. Under 'Invocations', there are two summary cards: 'Success Count' showing 3 successes and 'Error Count' showing 1 error, both for the 'Last 30 Days'. Below these is the 'Invocation Traces' section, which includes a description, a link to 'Run query in Application Insights', a 'Refresh' button, and a 'Filter invocations' input field. A table displays the twenty most recent function invocation traces.

Date (UTC)	Success	Result Code	Duration (ms)	Operation Id
<a href="#">2023-10-04 21:23:19.845</a>	✓ Success	0	35	8c022f03bc466ad6f632922e576b42c2
<a href="#">2023-10-04 21:15:14.576</a>	✓ Success	0	34	2953b3dc9fdc5acd6d7089ab80a252db
<a href="#">2023-10-04 21:13:34.696</a>	✗ Error	0	28	ff4a2d12d08140d807f68595252e4afd
<a href="#">2023-10-04 21:11:54.809</a>	✓ Success	0	88	25f5ebb82194c6422b60e8fb39b33fdf

## VIII. Conclusão

Bom, com essa Trigger do Cosmos DB, você pode pegar os dados alterados ou inseridos, inseri-lo em um outro contêiner, em um outro Cosmos DB, inserir no Blob Storage etc...

## Azure Function