

## Data Science – Princípios e Técnicas

Março  
2025



# Data Science



Onde me encontrar:

<https://www.linkedin.com/in/marco-mialaret-junior/>

e

<https://github.com/MatmJr>

# Consumo de API's

# Data Science



---

## Definição:

O termo API (Interface de Programação de Aplicativos) é bastante comum para o profissional de TI. Uma API permite a integração entre aplicações, enviando e recuperando dados de um servidor. Nesta aula vamos explorar o conceito de API e como utilizá-la com Python, destacando suas vantagens em comparação a arquivos CSV para acesso e uso de dados.

# Data Science



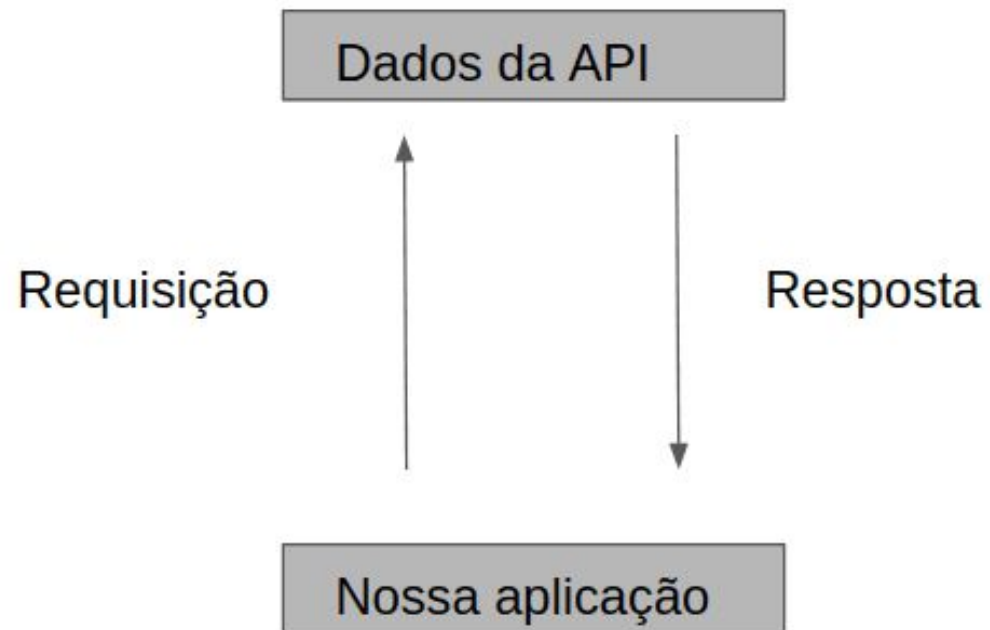
---

Em algumas situações, precisamos de informações específicas que mudam com frequência. Usar uma API permite acessar diretamente os dados necessários, sem baixar arquivos desnecessários, economizando tempo.

A API aparece como uma ponte (em tempo real) entre os dados e nossa aplicação, como mostra a imagem a seguir:

# Data Science

---





ETL

# Data Science



---

## Definição:

ETL (que significa extract-extrair, transform-transformar, load-carregar ) é um processo de integração de dados de longa data usado para combinar dados de várias fontes em um conjunto de dados único e consistente para carregar em um data warehouse, data lake ou outro sistema de destino.



# Data Science

---

Conforme os bancos de dados ganharam popularidade na década de 1970, surgiu o processo ETL, que se tornou essencial para integração e carregamento de dados em projetos de data warehousing.

# Data Science

---

O ETL estabelece a base para análises de dados, fluxos de aprendizado de máquina e é frequentemente utilizado em APIs para integração contínua de informações entre sistemas.

# Data Science

---

Por meio de regras de negócio, o ETL extrai dados de sistemas legados, realiza limpeza para garantir qualidade e consistência e os carrega em bancos de dados, permitindo desde relatórios básicos até análises avançadas que otimizam processos internos e melhoram a experiência do usuário final.

# Data Science



---

## Extrair (Extract)

Os dados são copiados ou exportados de fontes diversas para uma área de preparação. Fontes comuns incluem:

- Bancos de dados SQL ou NoSQL
- APIs externas (dados obtidos via requisições HTTP/REST)
- Arquivos simples
- E-mails
- Páginas da web

# Data Science



## Transformar (Transform)

Os dados obtidos são tratados e processados para atender objetivos analíticos específicos. Principais operações:

- Filtrar, limpar, validar e autenticar dados.
- Realizar cálculos, conversões (unidades, moedas, etc.), edição de textos e estruturação.
- Auditoria de qualidade e conformidade.
- Proteção e criptografia de dados sensíveis conforme regulamentação.

# Data Science



## Carregar (Load)

Dados transformados são enviados ao banco de dados.

- Primeiro ocorre uma carga inicial completa. Posteriormente são realizadas cargas incrementais ou atualizações periódicas.
- O processo geralmente é automatizado, realizado fora do horário comercial para evitar sobrecarga dos sistemas.
- O ETL também pode fornecer dados via APIs para consumo contínuo por outras aplicações, garantindo atualização dinâmica e integrada em tempo real ou quase real.

# Data Science

---

Mais detalhes:

<https://www.ibm.com/br-pt/topics/etl>

# Acessando uma fonte dos dados



# Data Science

---

Para fazer um primeiro exemplo vamos usar API meios de pagamentos do Governo Federal:

<https://dadosabertos.bcb.gov.br/dataset/estatisticas-meios-pagamentos>

# Data Science

## Estatísticas de Meios de Pagamentos



Este conjunto contém diversas estatísticas relacionadas a meios de pagamento. Atualmente está disponibilizado um recurso com informações sobre operações com cartões de pagamento, e outros 2 recursos com diversos instrumentos de pagamentos de varejo utilizados no país.

Estão disponibilizadas estatísticas dos seguintes meios de pagamento:

Instrumentos de periodicidade mensal, com atualização 17 dias após o final do mês de referência:

- Pix (fonte: SPI e documento 1201)
- TED (fonte: CIP-SITRAF e STR)
- Boletos (fonte: CIP-SILOC)
- DOC e TEC (fonte: CIP-SILOC)
- Cheque (fonte: Compe)

Instrumentos de periodicidade trimestral, com atualização 2 meses após o final do trimestre de referência:

- Cartões de crédito, débito e pré-pagos (fonte: documentos 6308 e 6334)
- Pagamentos de convênio, saques, transferências intra-bancárias e débito direto (fonte: documento 6209)



Serão disponibilizadas no futuro novos recursos relacionados a cartões de pagamentos, bem como canais de acesso utilizados em transações financeiras.

Por fim, anualmente são publicadas em uma planilha eletrônica demais estatísticas de meios de pagamentos, acessíveis pelo recurso 'Site do BC - Estatísticas de Meios de Pagamento'.

# Data Science

## Dados e recursos



### Site do BC - Estatísticas de Meios de Pagamento 🔥

Acesso à página de consulta de Estatísticas de Meios de Pagamentos no site do...

Explorar ▾



### Documentação 🔥

Arquivo de documentação do conjunto de dados.

Explorar ▾



### API - Documentação Swagger 🔥

Documentação da API utilizando a interface Swagger. Duas versões da interface...

Explorar ▾



### API - Navegador de Dados 🔥

Interface de navegação nos dados e construção da URL de chamada para todos os...

Explorar ▾



### API - Endpoint OData 🔥

Acesso à API utilizando padrão OData. Open Data Protocol (OData) é um...

Explorar ▾



### Estoque e transações de cartões 🔥

Quantidade de cartões emitidos e ativos e transações realizadas com este...

Explorar ▾



### Meios de Pagamentos Mensais 🔥

Conjunto de informações sobre operações com boletos bancários e de...

Explorar ▾



### Meios de Pagamentos Trimestrais 🔥

Conjunto de informações sobre operações com cartões de pagamento e de...

Explorar ▾

# Data Science

[🏠](#) > [Organizações](#) > [BCB/Decem](#) > [Estatísticas de Meios de ...](#) > [API - Navegador de Dados](#)

## API - Navegador de Dados

[🔗 Ir para recurso](#)

URL: [https://olinda.bcb.gov.br/olinda/servico/MPV\\_DadosAbertos/versao/v1/aplicacao#!/](https://olinda.bcb.gov.br/olinda/servico/MPV_DadosAbertos/versao/v1/aplicacao#!/)

Interface de navegação nos dados e construção da URL de chamada para todos os recursos da API.

### Endereço padrão:

[https://olinda.bcb.gov.br/olinda/servico/MPV\\_DadosAbertos/versao/v1/odata/\[codigo\\_recurso\]?\\$format=json&\[Outros Parâmetros\]](https://olinda.bcb.gov.br/olinda/servico/MPV_DadosAbertos/versao/v1/odata/[codigo_recurso]?$format=json&[Outros Parâmetros])

Os parâmetros disponíveis podem ser visualizados na [documentação](#)

Os códigos dos recursos disponibilizados são:

*Quantidadeetransacoesdecartoes MeiosdePagamentosMensalDA MeiosdePagamentosTrimestralDA*



### Formatos de Retorno:

Por padrão os dados podem ser retornados em 4 formatos:

- json (Padrão)
- xml
- text/csv
- text/html

Não há visões criadas para este recurso ainda.

# Data Science

## Estatísticas de Meios de Pagamento - v1

### Recursos

Título	Descrição
<a href="#">Estoque e transações de cartões</a>	Quantidade de cartões emitidos e ativos e transações realizadas com este instrumento de pagamento.
<a href="#">Meios de Pagamentos Mensais</a>	Conjunto de informações sobre operações com boletos bancários e de transferências de crédito. Dados ficam disponíveis a partir do 17º dia do mês subsequente.
<a href="#">Meios de Pagamentos Trimestrais</a>	Conjunto de informações sobre operações com cartões de pagamento e de transferências de crédito (boletos bancários, cartões de crédito e débito, transferências bancárias). Dados ficam disponíveis 90 dias após o final do trimestre.

Criado por Banco Central do Brasil

# Data Science

Essa API do governo oferece uma interface para facilitar a geração de url's

The screenshot shows a web interface for generating API URLs. The interface is organized into several sections:

- Trimestre:** A text input field with a red asterisk icon and a help icon. The value is "Exemplo: 20221".
- Máximo:** A text input field with a help icon. The value is "100".
- Filtro:** A text input field with a help icon. The value is "Exemplo: Nome eq 'João'".
- Ordenação:** A text input field with a help icon. The value is "Exemplo: Nome asc, Idade desc".
- Saída:** A dropdown menu with a help icon. The selected value is "json".
- Campos:** A list of checkboxes and field names. The fields are: Trimestre (checkbox), Bandeira do cartão (checkbox), Função do cartão (checkbox), Produto do cartão (checkbox), Quantidade de cartões emitidos (checkbox), and Quantidade de cartões ativos (checkbox). The corresponding field names are: trimestre, nomeBandeira, nomeFuncao, produto, qtdCartoesEmitidos, and qtdCartoesAtivos.
- URL de pesquisa:** A text area containing the generated URL: `https://olinda.bcb.gov.br/olinda/servico/MPV_DadosAbertos/versao/v1/odata/Quantidadeetransacoesdecartoes(trimestre=@trimestre)?@trimestre="&$top=100&$format=json`
- Buttons:** Three buttons at the bottom right: "Copiar URL", "Baixar json", and "Executar".

# Gerando a url



# Data Science

Vamos usar uma configuração inicial com 1 resultado

**\* Trimestre** ⓘ 20231

**Máximo** ⓘ 1

**Filtro** ⓘ Exemplo: Nome eq 'João'

**Ordenação** ⓘ Exemplo: Nome asc, Idade desc

**Saída** ⓘ json

**Campos**

<input checked="" type="checkbox"/> Trimestre ⓘ	trimestre
<input checked="" type="checkbox"/> Bandeira do cartão ⓘ	nomeBandeira
<input checked="" type="checkbox"/> Função do cartão ⓘ	nomeFuncao
<input checked="" type="checkbox"/> Produto do cartão ⓘ	produto
<input checked="" type="checkbox"/> Quantidade de cartões emitidos ⓘ	qtdCartoesEmitidos
<input checked="" type="checkbox"/> Quantidade de cartões ativos ⓘ	qtdCartoesAtivos

**URL de pesquisa**

`https://olinda.bcb.gov.br/olinda/servico/MPV_DadosAbertos/versao/v1/odata/Quantidadeetransacoesdecartoes(trimestre=@trimestre)?@trimestre='20231'&$top=1&$format=json&$select=trimestre,nomeBandeira,nomeFuncao,produto,qtdCartoesEmitidos,qtdCartoesAtivos,qtdTransacoesNacionais,valorTransacoesNacionais,qtdTransacoesInternacionais,valorTransacoesInternacionais`



# Data Science

TabelaPivotJSON

Trimestre	Bandeira do cartão	Função do cartão	Produto do cartão	Quantidade de car.	Quantidade de car.	Quantidade de tra.	Va
20231	American Express	Crédito	Intermediário	433549	335542	9107357	161798

Mudando para 1000 resultados:

 Trimestre ⓘ

Máximo ⓘ

Filtro ⓘ

# Montando o repositório

# Engenharia de Software

---

Vamos criar um repositório no github para o projeto.

**Atenção:** o repositório deve conter um README.md que será atualizado em cada aula. E vamos criar o hábito de fazer os nossos commits em branches, quem fizer commit na main terá 7 anos de azar.

# Engenharia de Software

---

Como vocês estão em um computador público façam o login no github em uma aba anônima.

# Engenharia de Software

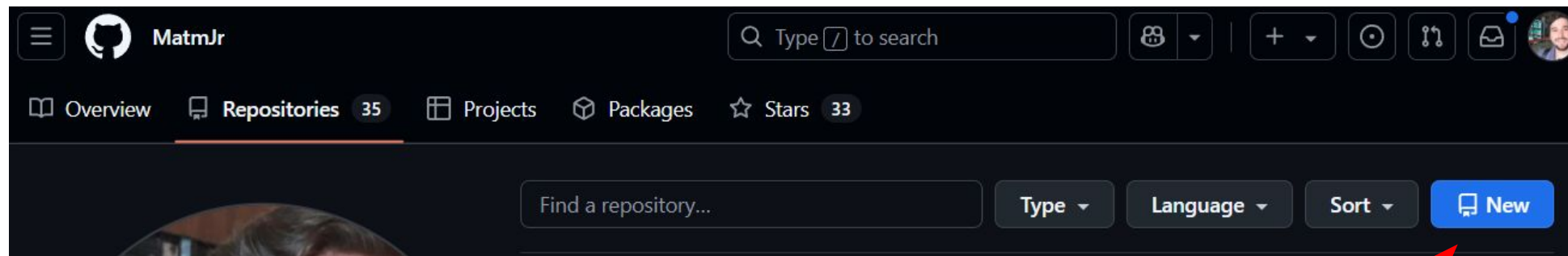
---

Acesse os seus repositórios no github:



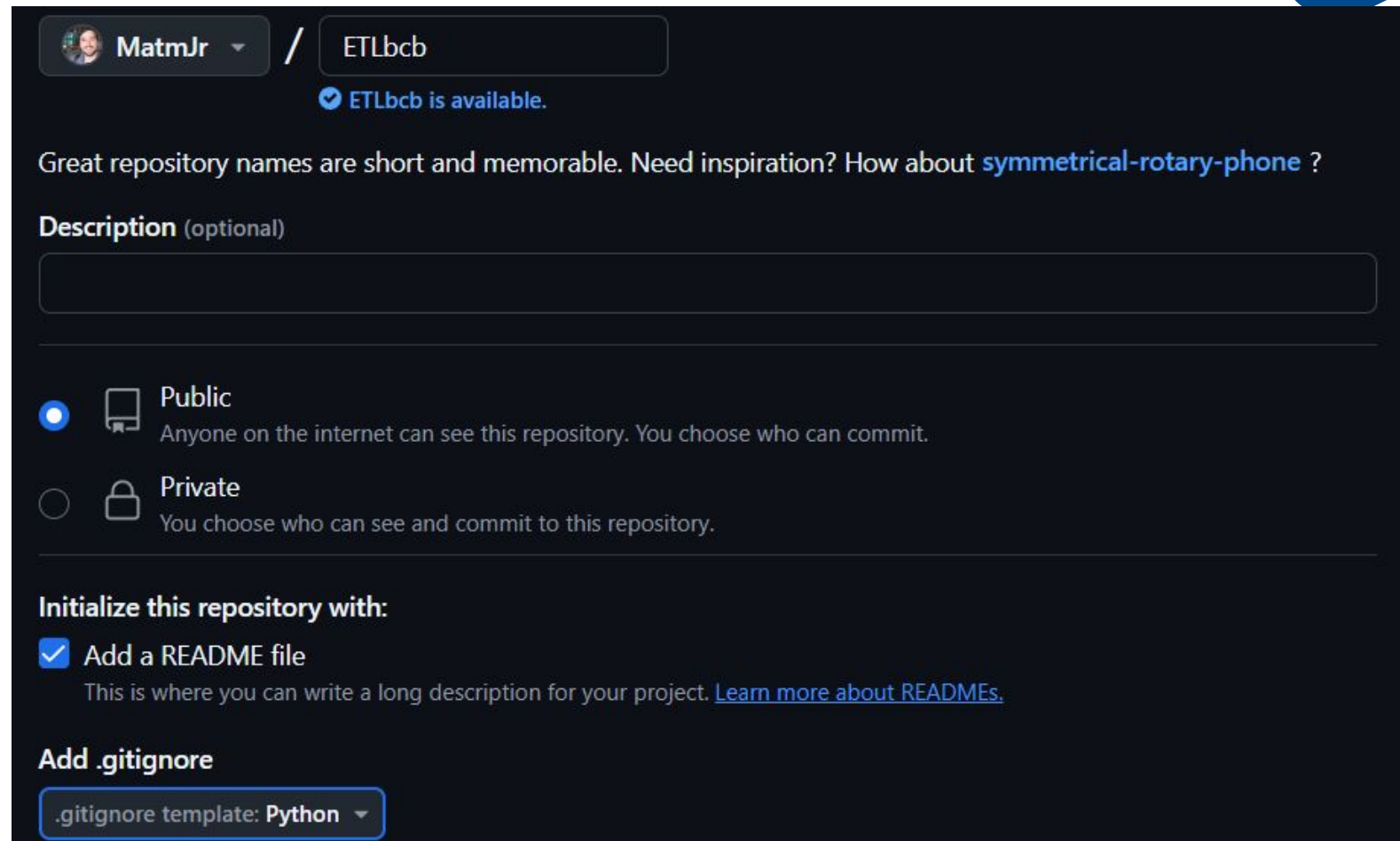
# Engenharia de Software

Clique no botão “Novo”:



# Engenharia de Software

Use as  
configurações  
a seguir:




The screenshot shows the GitHub 'Create new repository' page. At the top, the user 'MatmJr' is logged in, and the repository name 'ETLbcb' is entered. A status message indicates 'ETLbcb is available.' Below this, a prompt suggests repository names should be short and memorable, with an example 'symmetrical-rotary-phone'. A 'Description (optional)' text box is provided. The 'Visibility' section has two options: 'Public' (selected with a blue radio button) and 'Private'. The 'Initialize this repository with:' section has a checked checkbox for 'Add a README file'. At the bottom, the 'Add .gitignore' section shows a dropdown menu with 'Python' selected.


MatmJr / ETLbcb

✔ ETLbcb is available.

Great repository names are short and memorable. Need inspiration? How about [symmetrical-rotary-phone](#) ?

Description (optional)

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

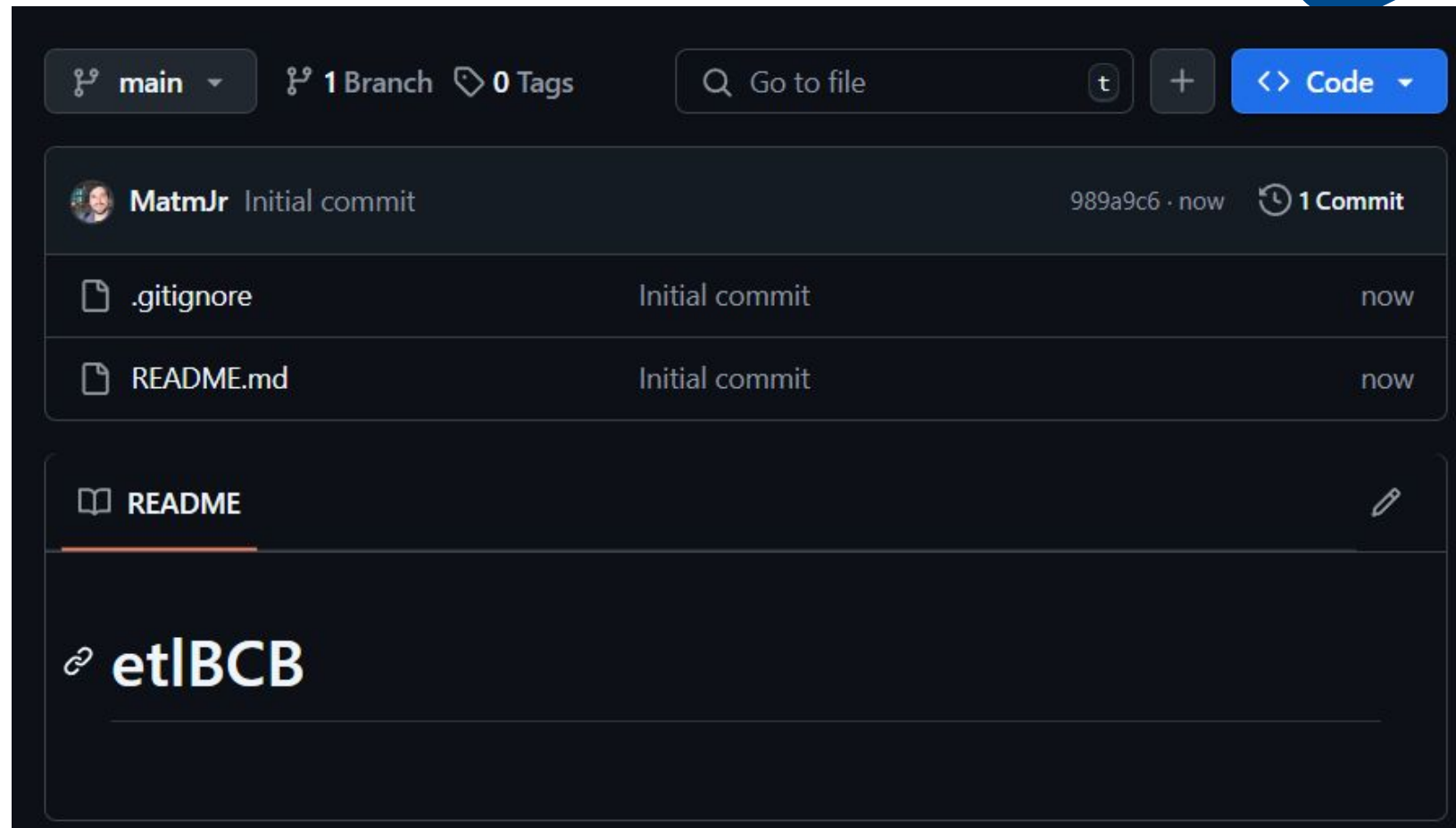
☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **Python**

# Engenharia de Software

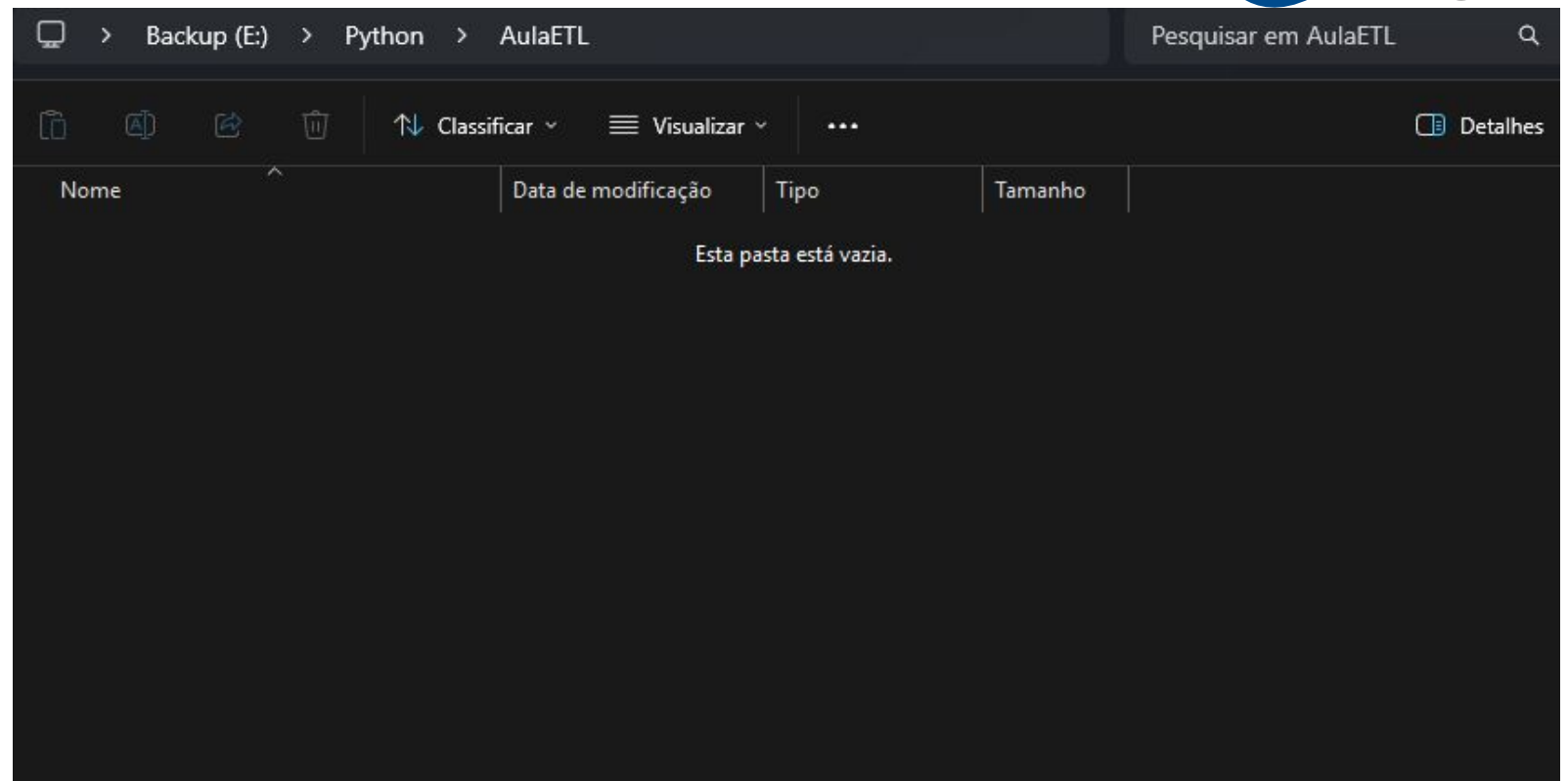
Após criar o repositório, faça um clone na sua máquina:





# Engenharia de Software

Crie uma pasta de trabalho. **Obs:** nos computadores do SENAC as pastas devem ser criadas em Downloads ou Documentos.



# Engenharia de Software



No terminal que abriu digite:

```
PS E:\Python\AulaETL> git clone sua_url_do_github|
```

**OBS.:** Se aparecer um erro dizendo que git não é um comando conhecido, você deve instalar o GIT no computador

<https://git-scm.com/downloads>

# Engenharia de Software

---

Caso dê tudo certo...

```
Cloning into 'etlBCB'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.
```

# Engenharia de Software

se você digitar dir (ou ls no Linux) no terminal:

```
PS E:\Python\AulaETL> dir

Diretório: E:\Python\AulaETL

Mode                LastWriteTime         Length Name
----                -
d-----          20/03/2025   20:00             etlBCB
```

# Engenharia de Software

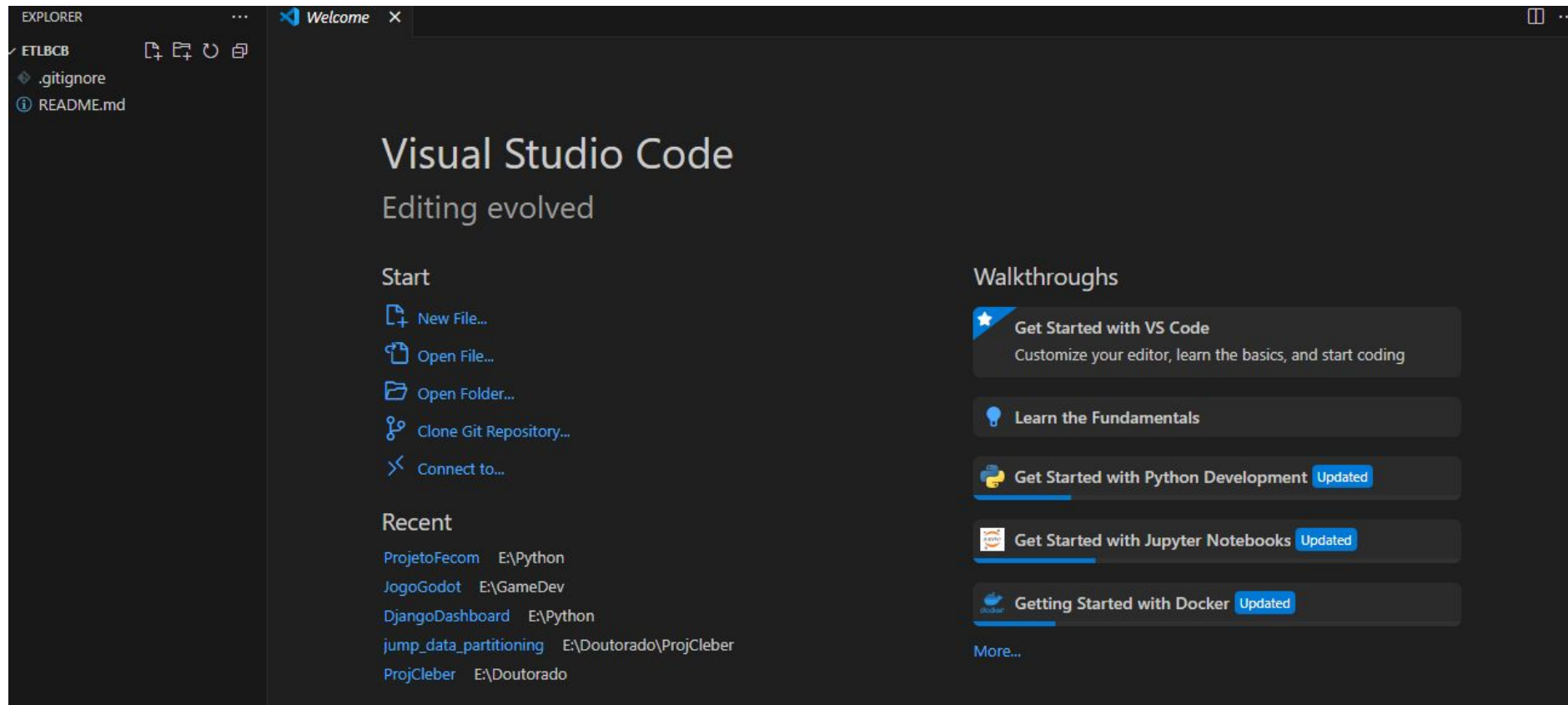
---

Agora digite cd e o nome da pasta que foi criada

```
PS E:\Python\AulaETL> cd etlBCB  
PS E:\Python\AulaETL\etlBCB> |
```

E agora digite: **code** .

# Engenharia de Software



# Engenharia de Software

---



Crie um arquivo chamado requirements.txt e adicione 2 linhas neste documento:

```
requests  
pandas
```

# Engenharia de Software

---

Agora crie um arquivo chamado `main.py` e **execute o processo para criar um venv.**



# Construindo o Código

# Engenharia de Software

---

Vamos criar uma variável para a nossa url:

url =

"[https://olinda.bcb.gov.br/olinda/servico/MPV\\_DadosAbertos/versao/v1/odata/MeiosdePagamentosTrimestralDA\(trimestre=@trimestre\)?@trimestre=%2720051%27&\\$format=json](https://olinda.bcb.gov.br/olinda/servico/MPV_DadosAbertos/versao/v1/odata/MeiosdePagamentosTrimestralDA(trimestre=@trimestre)?@trimestre=%2720051%27&$format=json)"

# Engenharia de Software

---

Precisamos converter a resposta HTTP (que é um texto JSON) em um objeto Python (dict ou list).

Essa etapa é conhecida como desserialização.

Depois vamos transformar o objeto Python obtido em um DataFrame.

# Engenharia de Software

---

```
import requests  
import pandas as pd
```

```
url = "link do slide anterior"
```

```
req = requests.get(url)  
print("Status Code:", req.status_code)  
data = req.json() #desserialização
```

# Engenharia de Software

---

```
df = pd.json_normalize(data["value"]) #DataFrame  
print(df)
```

# Engenharia de Software

---

**Exercício:** Criar uma função que recebe a data como parâmetro, faz uma requisição e retorna um DataFrame.

Salvar o repositório no Github.

# Dúvidas?

---



**Marco Mialaret, MSc**

**Telefone:**

**81 98160 7018**

**E-mail:**

**marcomialaret@gmail.com**

