

Criação de API com Python

Prof. Dr. Daniel Trevisan Bravo



Conceitos iniciais

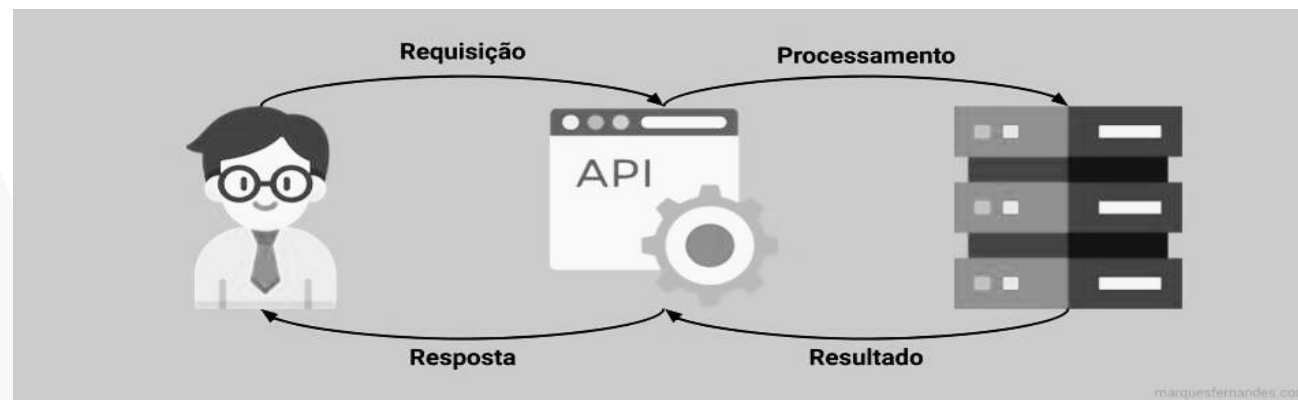
- Objetivo: ilustrar o passo a passo de uma das maneiras para se criar APIs com Python
- **Principal desafio:** criar as próprias APIs em Python para que sejam utilizadas para a resolução de problemas que possam ser solucionados por meio de algoritmos de ML
- **Solução mais adequada:** utilizar ferramentas e pacotes do Python para a criação e utilização das APIs.

O que é API?

"Application Programming Interface"

que significa em tradução para o português "Interface de Programação de Aplicativos"

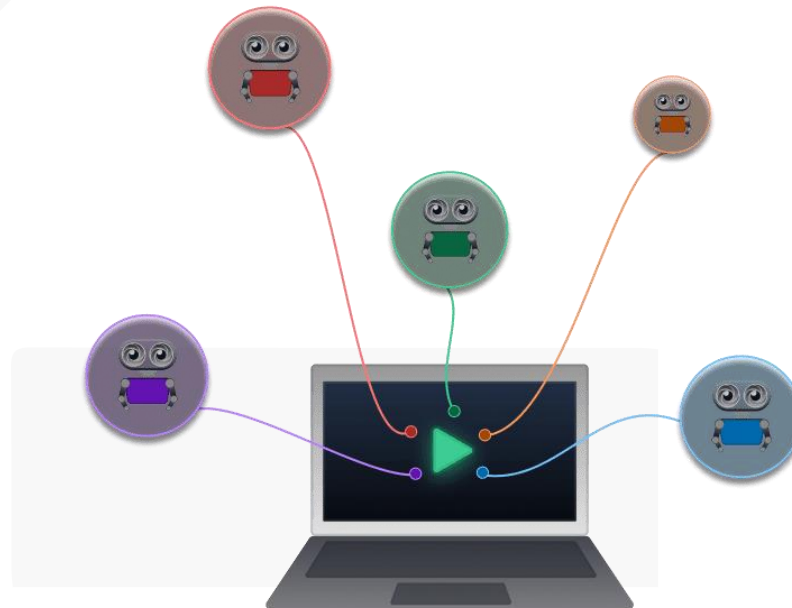
- **API** é um termo para designar uma interface de comunicação que um sistema oferece para que outros acessem suas funções, dados e recursos sem que o software ou plataforma externa precise saber como eles foram implementados.
- Trata-se de um **conjunto de rotinas e padrões** muito utilizados na web para facilitar a integração entre diferentes sites e aplicativos. O Google Maps, por exemplo, fornece uma API para que outros produtos utilizem os mapas em seus serviços.



Plataforma Replit

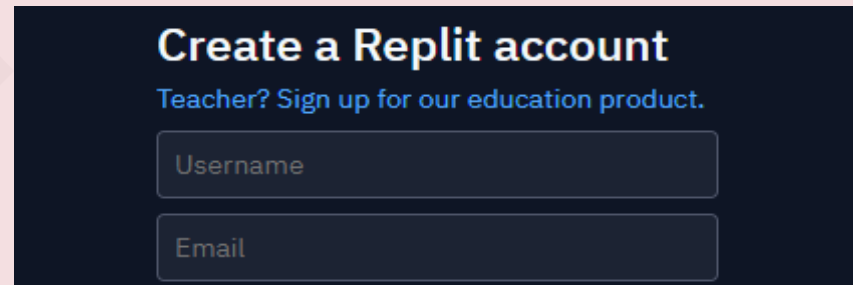


- Ambiente de desenvolvimento online para programar pelo navegador, o Replit permite que os usuários escrevam código e criem aplicativos e sites usando um navegador. O site também possui vários recursos colaborativos, incluindo capacidade para edição multiusuário em tempo real com um feed de bate-papo ao vivo. Ele suporta mais de 50 linguagens de programação e marcação, como Java , Python e HTML , permitindo que os usuários criem aplicativos, API's, etc...

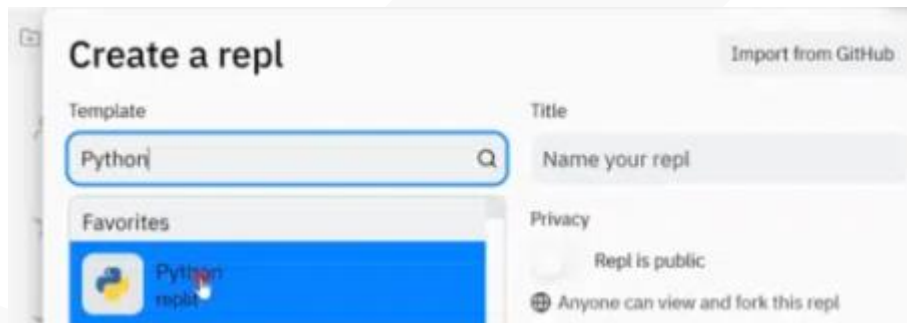


Exemplo - Criação de API

- **Passo 1:** Criar uma conta no site da Replit
<https://replit.com/>



The screenshot shows a dark-themed form titled "Create a Replit account". Below the title is a link: "Teacher? Sign up for our education product." There are two input fields: "Username" and "Email".



The screenshot shows the "Create a repl" form. It has a "Template" dropdown menu with "Python" selected. Below it is a "Favorites" section with a "Python repl" button. To the right, there is a "Title" field with the placeholder "Name your repl" and a "Privacy" section with a "Repl is public" checkbox and the text "Anyone can view and fork this repl".

- **Passo 2:** Clicar em + Create Repl, em seguida vai poder selecionar a linguagem que no nosso caso será Python e por fim dar um nome para a criação.

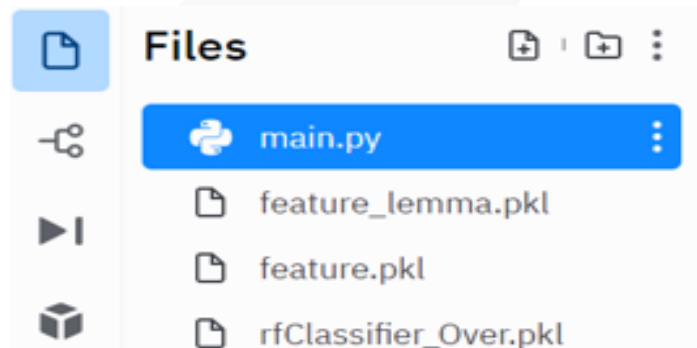
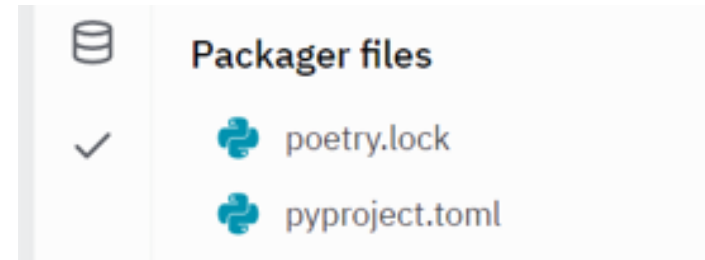
- **Passo 3:** Depois de criar você vai notar que já tem um editor de Python para colocar seu código.



The screenshot shows the Replit editor interface. On the left, there is a file explorer with "main.py" selected. The editor area shows a single line of code: "1 | Not sure what to do? Run some examples (start typing to dismiss)". On the right, there is a "Console" tab with the text "Python 3.8.2 (defa)" and a prompt ">".

Exemplo - Criação de API

- **Passo 4:** Baixar as bibliotecas necessárias dentro do próprio replit, basta ir em Packages e depois escrever o nome da biblioteca que deseja instalar.



- **Passo 5:** Subir no replit todos os arquivos necessários para a criação da API. Ex: arquivos txt, xlsx, modelos treinados de ML, etc...

- **Passo 6:** Agora só colocar o código dentro do Replit, e usar a biblioteca "flask" que serve para criação de páginas no Python, a partir dela que é gerado a URL que poderá ser chamada.



API para Análise de Sentimentos

Arquivos dos modelos
treinados de ML

Código de entrada

URL a ser usada
para chamar a API

The screenshot displays a Replit workspace for a sentiment analysis API. The interface is divided into three main sections: a file explorer on the left, a code editor in the center, and a console/output pane on the right.

File Explorer (Left): Shows a project named "main.py" with several files listed below it: `feature_lemma.pkl`, `feature.pkl`, `rfClassifier_Over.pkl`, `rfClassifier_Puro.pkl`, `vectorizer_lemma.pk`, and `vectorizer.pk`. These are labeled as "Arquivos dos modelos treinados de ML". Below these are "Packager files" including `poetry.lock` and `pyproject.toml`, which are labeled as "Pacotes".

Code Editor (Center): Displays the `main.py` file, which contains Python code for a Flask application. The code imports `Flask`, `jsonify`, `request`, `re`, `cleantext`, `nltk`, `stopwords`, `word_tokenize`, `joblib`, `pickle`, and `numpy`. It defines a `Flask` app and a `mostra` function that returns sentiment labels based on input values.

Console/Output (Right): Shows the URL `https://ClassifTweets.danieltb3006.repl.co` in the browser address bar, labeled as "URL a ser usada para chamar a API". Below the URL, the text "API está no ar" is displayed. The console output shows a warning message: "Use a production WSGI server instead. * Debug mode: off * Running on all addresses (0.0.0.0) WARNING: This is a development server. Do not use it in a production deployment. * Running on http://127.0.0.1:5000 * Running on http://172.18.0.16:5000 (Press CTRL+C to quit)".

Testando a API

Após rodar o código dentro do Replit, basta copiar a URL e utilizar na aplicação desejada para chamar a API, nesse caso utilizamos o Jupyter Notebook para fazer a requisição!

```
In [1]: import requests
```

```
In [2]: texto = 'A @ClaroBR tem internet lixo que toda hora cai. Serviço de bosta'
link = 'https://ClassifTweets.danielb3006.repl.co/classifica_tweet?texto='+texto

requisicao = requests.get(link)

print(requisicao)
print(requisicao.json())

dicionario = requisicao.json()
```

```
<Response [200]>
{'Sentimento': 'Negative'}
```