

Enterprise Connection

MAIS QUE AMIGOS FRIENDS

01

Principais Características

ALGORITMO

LINGUAGEM ESCOLHIDA:

Escolhemos Python para desenvolver nosso algoritmo.

BIBLIOTECAS

Usamos algumas bibliotecas: profanity-check, matplotlib, nltk , alt-profanity-check, wordcloud, re, collections.

02

MAIS QUE AMIGOS FRIENDS



Desenvolvimento

O Algoritmo tem como objetivo indentificar as palavras chaves de um texto e organiza-las em uma imagem, salvando essa imagem como PNG.

MAIS QUE AMIGOS FRIENDS

Algoritmo

```
def generate_cloud_words(text: str, qty_key_words: int) → None:  
    nltk.download('stopwords')  
  
    text = text.split()  
  
    stopwordsNltk = stopwords.words('portuguese')  
  
    words = []  
  
    for word in text:  
        if word not in stopwordsNltk and word ≠ '' and len(word) ≥ 3 and predict([word]) ≠ 1:  
            words.append(word)  
  
    count = Counter(words)  
    most_common = count.most_common(qty_key_words)      You, 3 days ago • Chore: implement script.  
  
    words = [x[0] for x in most_common]  
  
    z = lambda x: re.compile('#').sub('', re.compile('rt @').sub('@', x, count=1).strip())  
    words = z(str(words).replace("'", ""))  
  
    wordCloud = WordCloud(width=2400, height=1600, margin=0).generate(words)  
    plt.figure(figsize=(20, 11))  
    plt.imshow(wordCloud, interpolation='bilinear')  
    plt.axis("off")  
    plt.margins(x=0, y=0)  
    wordCloud.to_file("cloud.png")  
  
    display(most_common)
```

01

```
def display(most_common: List) → None:  
    print(f'\nPalavras capturadas:\n')  
    t = PrettyTable(['Palavra', 'Frequência'])  
  
    for data in most_common:  
        t.add_row([data[0], data[1]])  
  
    print(t)  
  
def main() → None:  
    qty_key_words = int(input('Quantas palavras chave deseja obter: '))  
  
    with open('to-read.txt', mode='r', encoding='utf8') as f:  
        text = f.read()  
  
    text = text.lower()  
  
    trash = ['.', ',', ';', ':', '?', '!', '\'', '\"', '\"', '\"', '\"', '(', ')', '}', '{', '[', ']', '@', '#', '%', '^', '&', '*', '/', '\\\\', '+', '\n']  
  
    for t in trash:  
        text = text.replace(t, " ")  
  
    remove_digits = str.maketrans('', '', digits)  
    text = text.translate(remove_digits)  
  
    words = str(len(text.split()))  
    letters = str(len(text))  
  
    print(f"\nO texto contém {words} palavras e {letters} letras.\n")  
    generate_cloud_words(text, qty_key_words)  
  
if __name__ == "__main__":  
    main()
```

02

04

Obrigado!

Washington Sampaio Vieira - RM86423

Jorge Carriel Nunes - RM81323

Camila Figueira - RM88302

Juan Versolato Lopes - RM88516

Lucas dos Santos Melo - RM86429