Buscando Empresas com Web Scraping

A inspiração desse projeto surgiu após uma conversa com uma diretora que tinha interesse de conhecer as empresas do estado de Sergipe, sejam elas de pequeno, médio ou grande porte.

PROPÓSITO

Praticar técnicas de Web Scraping para extrair dados de sites e gerar o dataset com as informações solicitadas para o cliente final. Assim como treinar o uso da biblioteca pandas e a linguagem programação Python.

Objetivo

Módulo 1

- Entender o que é webscrapping
- Como gerar dados a partir da extração

Módulo 2

- Utilizar API
- Visualizar dados através do mapa

Antes de continuar

Recomendo a leitura deste documento para quem tem algum conhecimento em programação e de preferência em Python.
Caso tenha interesse de testar em sua máquina, certifique que tem os pacotes utilizados neste documento instalados em sua máquina.

É muito importante que tenha na pasta do seu notebook o arquivo geckodriver, baixe a versão compatível com seu pc aqui: https://github.com/mozilla/geckodriver/releases, ou, faça um clone desta documentação que se encontra também no github através desta URL: https://github.com/rafaelandradeslv/busca_empresas

Introdução

Assim como um pirata dos mares se aventura nos mares procurando saquear navios, existe o pirata web que navega por sites para extrair informações de forma autônoma para obter os dados de seu interesse.

Estou lhe convidando a entender um pouco do que aprendi ao estudar as técnicas de web scraping, vou tentar introduzir o passo a passo em código até alcançar o objetivo final.

Dados a serem coletados

Após o conhecimento dos dados solicitados, iremos em busca para obtê los.

- Razão social
- Nome fantasia
- CNPJ
- Porte nominal
- Setor
- Atividade primária
- Localização
- Endereço
- Cep

- Quantidade de funcionários
- Sócios administradores

O nosso destino será a busca por empresas de Sergipe. Depois de procurar por sites que pudessem fornecer os dados acima, essa url foi encontrada: **https://www.econodata.com.br/lista-empresas/SERGIPE.**

Acesse o site acima e veja a sua estrutura de tabelas. Em seguida, clique em qualquer empresa para visualizar a estrutura dos dados que serão extraídos. E agora Com o conhecimento do site que contém os dados desejados, vamos para o código.

Importando Pacotes

```
import pandas as pd
import numpy as np

import requests
import json
import time

from bs4 import BeautifulSoup as bsp
from selenium import webdriver
from selenium.webdriver.firefox.options import Options
```

Funções

NOTA: As funções a seguir, foram criadas com intuito de organizar a escrita do código.

```
# função dedicada aos dados de contato
def add_list(var, lista):
    soup = bsp(var.text, 'html.parser')
    soup = soup.get_text()
    lista.append(soup)
    return lista
```

```
# FUNÇÃO QUE BUSCA OS ELEMENTOS E ADICIONA OS DADOS AO DATAFRAME
def busca_adiciona_elementos():
    lista = []
    contador = 1
    # percorrer "DADOS DE CONTATO"
    while contador <= 8:
        dados_de_contato = driver.find_element_by_xpath(f'//*[@id="companyDetailsForm"]/section/div/div/div[1]/div[4]/div/div[2]/table/tbod
        add_list(dados_de_contato, lista)
        contador += 1

    contador = 1
    # percorrer "PORTE ESTIMADO"
    while contador <= 3:
        dados_porte_estimado = driver.find_element_by_xpath(f'//*[@id="companyDetailsForm"]/section/div/div/div[1]/div[5]/div/div[2]/table/
        add_list(dados_porte_estimado, lista)
        contador += 2

    # percorrer "SOCIEDADE"
    dados_sociedade = driver.find_element_by_xpath('//*[@id="companyDetailsForm"]/section/div/div/div[1]/div[6]/div/div[2]/div/div[2]/p[1]'
    soup = bsp(dados_sociedade.text, 'html.parser')
    soup = soup.get_text()
    lista.append(soup)

    df.loc[len(df)+1] = lista

    return lista
```

Criando Dataset Base

```
# dicionário para criar a tabela
name_columns = dict()
name_columns['nome_fantasia'] = ''
name_columns['setor'] = ''
```

```
name_columns['cnpj'] = ''
name_columns['atividade_primaria'] = ''
name_columns['fundacao'] = ''
name_columns['localizacao'] = ''
name_columns['endereco'] = ''
name_columns['cep'] = ''
name_columns['porte_medio'] = ''
name_columns['qnt_funcionarios'] = ''
name_columns['socios_administradores'] = ''
```

```
# CRIANDO O DATASET

# criando o dataset
df = pd.DataFrame(name_columns.items())
df = df.transpose() # transforma as linhas em colunas

# após criar o dataset
df.columns = df.iloc[0] # renomeia as colunas
df = df.drop(df.index[0]) # remove o primeiro índice a primeira linha
df
```

Primeira tentativa

Seria tudo perfeito se de primeira o código a seguir tivesse funcionado, mas na verdade descobri que a vida de um pirata não é tão fácil como parece, se é que parece alguma coisa rsrs.

Para melhor prática e entendimento recomendo que descomente as linhas de código a seguir e tente perceber o problema, e depois parta para a solução que desenvolvi a seguir , caso tenha tentado e não percebeu o erro, fique a vontade para me enviar uma mensagem no LinkedIn dizendo: Cadê o erro ?

```
# ESTADO = 'SERGIPE'
# url = f"https://www.econodata.com.br/lista-empresas/{ESTADO}"

# option = Options()
# option.headless = True # opção para não abrir o navegador externo
# driver = webdriver.Firefox() # para ativar o headless passar options=option
# driver.get(url)

# time.sleep(2)
# teste = driver.find_element_by_xpath('//*[@id="j_idt10:j_idt11"]/div[1]/table')
# soup = bsp(teste.text, 'html.parser')

# try:
#     contador = 1
#     while contador <= 10:

#         time.sleep(2)
#         driver.find_element_by_xpath(f'//*[@id="j_idt10:j_idt11_data"]/tr[{contador}]/td[1]/a/span[1]').click()

#         time.sleep(1)
#         busca_adiciona_elementos() # função busca elementos e adiciona itens no dataframe

#         driver.get(url)
#         time.sleep(2)
#         contador += 1

# except:
#     print('código finalizado: Error')

# driver.quit()
# df
```

Segunda tentativa

PIRATA WEB :

- Eu vim pegar esses dados e só saio daqui com ele!

A partir desse pensamento, foi construído a seguinte solução :

É claro que os dados alvo são os de Sergipe, mas quem disse que não podemos coletar de outros estados caso tenham interesse?

```
# Preencher com hífen os nomes dos estado que tiver espaço (RIO-DE-JANEIRO)
listaDeEstado = ['SERGIPE', 'RIO-DE-JANEIRO', 'SAO-PAULO']
url = f"https://www.econodata.com.br/lista-empresas/{listaDeEstado[0]}"
```

Na primeira tentiva não utilizei a lib requests, então recoori a ela em busca de uma solução possível e gerei o código a seguir.

```
r = requests.get(url, auth=('user', 'pass'))
html = r.text

soup = bsp(html, 'html.parser')
html = soup.table.find_all('a')
```

NOTA: Rode a variável html para visualizar o resultado

Gerando o Dataset

```
option = Options()
option.headless = True
driver = webdriver.Firefox(options=option) # se desejar ver o navegador abrir e fechar, retire "options=option"

print(f'Quantidade de itens : {len(html)}')
contador = 0
while contador < len(html):
    print(f"Contador : {contador}")
    try:
        response = html[contador].get('href')
        site = 'https://www.econodata.com.br' + response

        driver.get(site)

        busca_adiciona_elementos() # função para buscar elementos e adicionar ao dataframe

        contador += 1
        driver.quit()
    except:
        contador += 1
        driver.quit()
        driver = webdriver.Firefox(options=option)

driver.quit()
```

Salvando Dataset em CSV

```
#salvando o dataset para não repetir o código sempre
df.to_csv('EmpresasSergipe.csv', encoding='utf-8', index=False)
```

Conclusão

Vimos como foi possível extrair dados de sites como um pirata web e terminamos este primeiro módulo salvando os dados em CSV.

OBSERVAÇÃO: Uma etapa da coleta de dados foi concluída, fica a seu critério e dever de casa dar continuidade a implementação de novas funcionalidades e busca de dados.

Próximo módulo:

No próximo módulo pretendo trazer a visualização das empresas no mapa.

Referências

- https://www.selenium.dev/documentation/webdriver/
- https://www.econodata.com.br/lista-empresas/SERGIPE
- https://beautiful-soup-4.readthedocs.io/en/latest/
- https://docs.python-requests.org/en/master/
- https://pandas.pydata.org/pandas-docs/dev/reference/