المشروع الفصلي لمادة برمجة الشبكات بعنوان:

تجريف الويب باستخدام المكتبة beautiful soup

إعداد وتقديم:

ليلى مروان ديب 1978

حليمة مصطفى الضائع 2013

إشراف الدكتور:

مهند عيسى

ملخص

تجريف الويب هو عملية أتمتة استخراج البيانات من مواقع الويب. يمكنك إنشاء برنامج جرف ويب لإخراج شيء ما من صفحة الويب، مثل جمع مراجعات الكتب من نظام أساسي تابع لجهة خارجية، أو تنزيل جميع كلمات الأغاني المفضلة لديك، أو للمتعة فقط كمتصفح. تتوفر العديد من الأدوات الشائعة لجرف الويب ، مثل Beautiful Soup و Scrapy و Selenium وما إلى ذلك. في هذه المقالة سنقوم بشرح جرف الويب باستخدام المكتبة Beautiful Soup وبعدها سنقوم بإنشاء برنامج بايثون وباستخدام المكتبة Beautiful Soup لاستخلاص معلومات حول الوظائف في موقع http://pythonjobs.github.io وبعدها كتابة هذه النتائج في ملف CSV.

Web Scrapping Using beautiful soup

Abstract: Web scraping is the process of automating data extraction from websites. You can create web scrapping software to get something out of the web page, like collect book reviews from a third party platform, download all your favorite song lyrics, or just for fun as a browser. Many popular tools are available for web scraping, such as Beautiful Soup, Scrapy, Selenium, etc. In this article we will explain web scraping using the Beautiful Soup library and then we will create a Python program and using the Beautiful Soup library to extract information about jobs at http://pythonjobs.github.io and then write this informations in a csv file.

مقدمة:

يتحدث الجميع في الوقت الحاضر عن البيانات وكيف تساعد في تعلم الأنماط المخفية والرؤى الجديدة. يمكن أن تساعد المجموعة الصحيحة من البيانات الشركة على تحسين إستراتيجيتها التسويقية ويمكن أن يؤدي ذلك إلى زيادة المبيعات الإجمالية. ودعونا لا ننسى المثال الشعبي الذي يستطيع فيه السياسي معرفة رأي الجمهور قبل الانتخابات. البيانات قوية لكنها لا تأتي بالمجان. دائماً ما يكون جمع البيانات الصحيحة أمراً مكلفاً؛ فكر في الاستطلاعات أو الحملات التسويقية ، إلخ.

الإنترنت عبارة عن مجموعة من البيانات، وباستخدام المجموعة الصحيحة من المهارات، يمكن للمرء استخدام هذه البيانات بطريقة معينة لاكتساب الكثير من المعلومات الجديدة. يمكن دائماً نسخ البيانات ولصقها إلى ملف Excel أو CSV، ولكن هذا أيضاً يستغرق وقتاً طويلاً ومكلفاً.

تجريف على شبكة الإنترنت، حصاد الإنترنت ، أو استخراج البيانات على شبكة الإنترنت هي مصطلحات تعبر عن استخراج البيانات من المواقع. يستطيع برنامج جرف الويب الوصول مباشرة إلى شبكة الويب العالمية باستخدام بروتوكول نقل النص التشعبي أو مستعرض ويب. بينما يمكن إجراء تجريف الويب يدوياً بواسطة المستخدم، يشير المصطلح عادةً إلى العمليات الآلية التي يتم تنفيذها باستخدام روبوت أو زاحف ويب. ويمكن تعريفه أيضاً على أنه شكل من أشكال النسخ يتم فيه جمع بيانات محددة ونسخها من الويب، عادةً إلى قاعدة بيانات محلية مركزية أو جدول بيانات، لاسترجاعها لاحقاً أو تحليلها.

يتضمن تجريف صفحة ويب جلبها واستخراج البيانات منها. الجلب هو تنزيل الصفحة (وهو ما يفعله المتصفح عندما يعرض المستخدم الصفحة). لذلك ، يعد زحف الويب مكوناً رئيسياً في تجريف الويب لجلب الصفحات لمعالجتها لاحقاً. بمجرد إحضارها، يمكن أن يتم الاستخراج. يمكن تحليل محتوى الصفحة أو البحث فيه أو إعادة تنسيقه أو نسخ بياناتها في جدول بيانات أو تحميلها في قاعدة بيانات. عادةً ما تأخذ برامج جرف الويب شيئاً ما من الصفحة، للاستفادة منه لغرض آخر في مكان آخر. من الأمثلة على ذلك البحث عن الأسماء وأرقام الهواتف أو الشركات وعناوين URL الخاصة بها أو عناوين البريد الإلكتروني ونسخها إلى قائمة (جرف جهات الاتصال).

يستخدم التجريف على شبكة الإنترنت لجرف الاتصال، وكعنصر من عناصر التطبيقات المستخدمة لفهرسة الويب، وأيضاً في التعدين على شبكة الإنترنت واستخراج البيانات ورصد تغير الأسعار على الانترنت ومقارنة الأسعار، واستعراض المنتجات (لمشاهدة المنافسة)، وجمع قوائم العقارات، بيانات الطقس المراقبة، واكتشاف تغيير موقع الويب، والبحث، وتتبع التواجد والسمعة عبر الإنترنت، ومزج الويب، وتكامل بيانات الويب.

أتاحت بايثون الاستفادة من جرف الويب حيث قدمت العديد من المكتبات وأهمها هي مكتبة Beautiful Soup وتوابعها الكثيرة وسنتعامل مع هذه المكتبة في المثال العملي.

أدوات وطرائق البحث:

1- مكتبة requests:

هذه المكتبة من المكتبات المشهورة في بايثون وهي خاصة ببروتوكول HTTP وتدعم كل العمليات الخاصة به مثل عمليات POST, GET واستقبال ومعالجة الرد، وهي سهلة الاستخدام ويُمكننا القول إن قوة هذه المكتبة يكمن في سهولتها.

أغلب عمليات تجريف الانترنت لا تستغني عن هذه المكتبة المهمة، ودورها الأساسي في التجريف هو الحصول على مكونات الصفحة الأساسية كصفحة (html).

لتثبيت المكتبة نستخدم أداة pip لذلك ونُنفذ الأمر التالي في الطرفية:

pip install requests

المثال التالي يوضح كيفية قراءة الصفحة الرئيسية لصفحة وظائف البايثون:

```
import requests
page = requests.get('http://pythonjobs.github.io')
contents = page.content
```

في البداية نستورد مكتبة requests ومن ثم نستخدم التابع get لإرسال request والحصول على الرد الناتج من طلب العنوان الذي مررناه للتابع. النتيجة سيتم حفظها في المتغير page والذي هو عبارة عن كائن من نوع Response يتضمن معلومات الرد.

2- مكتبة Beautiful soup:

هي مكتبة مُهمتها تفسير/تحليل/قراءة ملفات html وxml فبدلاً من بناء كود برمجي يعمل على تفسير ومعالجة هذه الملفات، من الممكن استخدام هذه المكتبة لقراءة عناصر ملف html والبحث فيها والحصول على قيم هذه العناصر وبذلك يتم توفير وقت وجهد كبيرين.لو أردنا مثلًا الحصول على جميع الروابط الموجودة في الصفحة الرئيسية لمدونة بايثونات، نستطيع تحقيق ذلك باستخدام مكتبة Requests ومكتبة Beautiful Soup معًا.

لتثبيت مكتبة Beautiful Soup نستخدم مرة أخرى الأداة pip

pip install beautifulsoup4

المثال التالي يوضح كيفية الحصول على جميع الروابط الموجودة في الصفحة الرئيسية لمدونة بايثونات:

```
from bs4 import BeautifulSoup
import requests
req = requests.get("https://pythonat.com")
bs = BeautifulSoup(req.text,"html.parser")
for link in bs.findAll('a'):
      print(link.get("href"))
```

في البداية نستورد مكتبتي requests و beautfulsoup ومن ثم نُرسل طلب لرابط المدونة الرئيسي ونحفظ الرد في المتغير req. إلى الآن نمتلك html الخاص بالصفحة الرئيسية للمدونة فقط ونحتاج الان لاستخراج الروابط منه. نُعرف متغير باسم bs من نوع BeautifulSoup والذي يُمثل المُفسر (Parser) ومن ثم نستخدم التابع findAll للحصول على جميع العناصر من نوع رابط في الصفحة ومن ثم نطبع كل قيمة للسمة href.

عند تشغيل الكود السابق سنحصل على قائمة عديدة من الروابط منها:

https://pythonat.com/tag/machine-learning/

https://pythonat.com/tag/pandas/

https://pythonat.com/feed/

https://pythonat.com/comments/feed/

يُمكن تطوير المثال السابق وتحويله إلى أداة لجلب الروابط وحفظها في ملف خارجي، أو جلب الصور والملفات.

3- مكتبة pandas:

تستخدم هذه المكتبة من أجل إنشاء أطر بيانات خاصة بكل مشروع بحيث يتناسب إطار البيانات مع حاجة المبرمج ويتم عرض إطار البيانات عل شكل جدول مرتب ترتيب واضح ومفهوم وأيضاً تتيح هذه المكتبة حفظ إطار البيانات على شكل ملف csv.

يجب تنزيل هذه المكتبة باستخدام التعليمة التالية:

pip install pandas

المنهجيات العلمية:

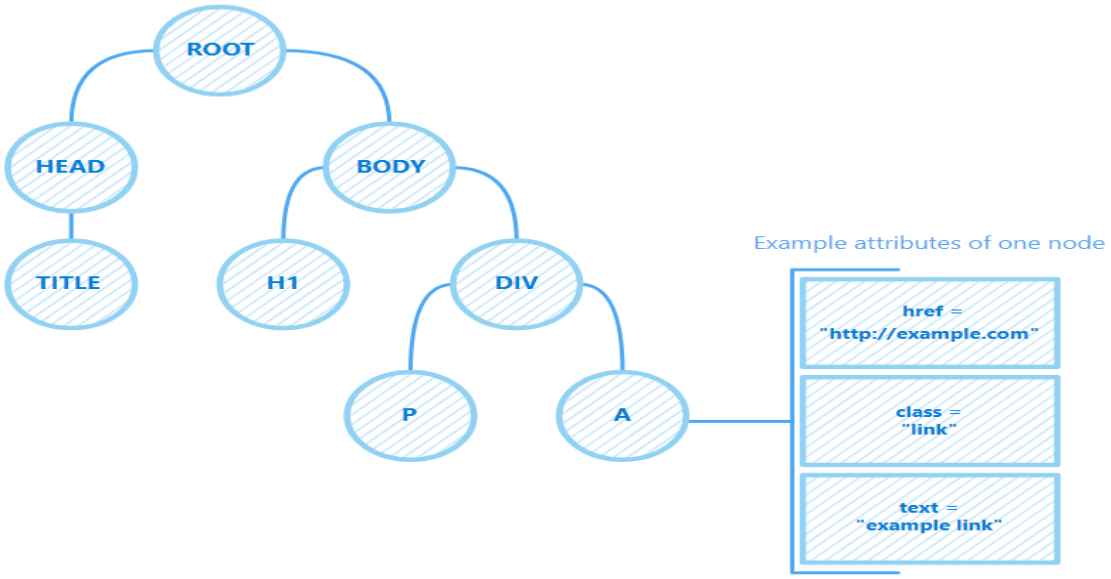
1- ما هو HTML؟

تم تصميم HTML (لغة ترميز النص التشعبي) بحيث يسهل قراءتها آلياً والتحليل. بمعنى آخر، يتبع HTML بنية شجرة مثل بنية العقد (علامات HTML) وسماتها ، والذي يمكننا من التنقل بسهولة برمجياً.

لنبدأ بصفحة نموذجية صغيرة ونوضح هيكلها:

```
<head>
  <title>
  </title>
</head>
<body>
  <div>
    <h1>Introduction</h1>
    <p>some description text: </p>
    <a class="link" href="http://example.com">example link</a>
  </div>
</body>
```

في هذا المثال الأساسي لكود بسيط لصفحة الويب، يمكننا أن نرى أن المستند يشبه بالفعل شجرة بيانات بمجرد النظر إلى المسافة البادئة.



الشكل 1هيكلية صفحةhtml

من الشكل السابق ، يمكننا تبسيط الفكرة بشكل أكبر – إنها شجرة من العقد وتتكون كل عقدة من:

اسم العقدة – ويعرف أيضًا باسم علامة HTML ، على سبيل المثال

الخصائص الطبيعية – قيمة النص والموضع.

خصائص الكلمات الرئيسية – قيم الكلمات الرئيسية مثل class و href وما إلى ذلك.

من خلال هذا الفهم الأساسي، يمكننا أن نرى كيف يمكن أن يساعدنا Python و beautifulsoup في اجتياز هذه الشجرة لاستخراج البيانات التي نحتاجها.

2- تحليل HTML باستخدام BeautifulSoup

Beautifulsoup هي مكتبة بايثون وهي في الأساس أداة محلل HTML. باستخدامها يمكننا التنقل في بيانات HTML لاستخراج / حذف / استبدال عناصر HTML معينة. تأتي BS4 أيضاً بوظائف مفيدة مثل التنسيق المرئي وتنظيف شجرة التحليل.

دعنا نلقي نظرة سريعة على أكثر ميزات beautiful soup فائدة في سياق تجريف الويب. سنبدأ بكود المصدر لصفحة HTML بسيطة:

```
<head>
  <title class="page-title">Hello World!</title>
</head>
<body>
  <div id="content">
    <h1>Title</h1>
    <p>first paragraph</p>
    <p>second paragraph</p>
    <h2>Subtitle</h2>
    <p>first paragraph of subtitle</p>
  </div>
</body>
```

هنا ، لدينا جزء بسيط جداً من بيانات HTML يحتوي على العناصر الأساسية للمقالة: العنوان والعنوان الفرعي وبعض فقرات النص. أولاً، نحتاج إلى اختيار الواجهة الخلفية لمحلل beautifulsoup أو beautifulsoup parser. لا تطبق Beautifulsoup محلل التنقل الشجري الخاص بها وتعتمد بدلاً من ذلك على واحدة من 3 واجهات خلفية متوفرة:

- html.parser – محلل لغة بايثون المدمج، والذي تمت كتابته بلغة بايثون ولكنه بطيئ قليلاً.
- lxml – مكتبة تستند إلى C لتحليل HTML: سريع ، لكن تثبيته أصعب قليلاً.

- html5lib – محلل آخر مكتوب بلغة python يُقصد به أن يكون متوافقًا تمامًا مع html5.

إذا قمنا بتطبيق التابع find_all('div', id=' content ') سيعيد هذا التابع جميع محتوى العقدة div على شكل list مثل العلامة h1 ومحتوياتها والعلامة p ومحتوياتها وهكذا .. ولكن كل علامة تكون على شكل object يمكن الوصول لها.

القسم العملي:

الآن سنبدأ في إنشاء مجرفة الويب الخاصة بنا. موقع الويب الذي سنقوم بجرفه هو لوحة معلومات للوظائف تسرد أحدث وظائف Python. في هذا المشروع سنقوم بجرف:

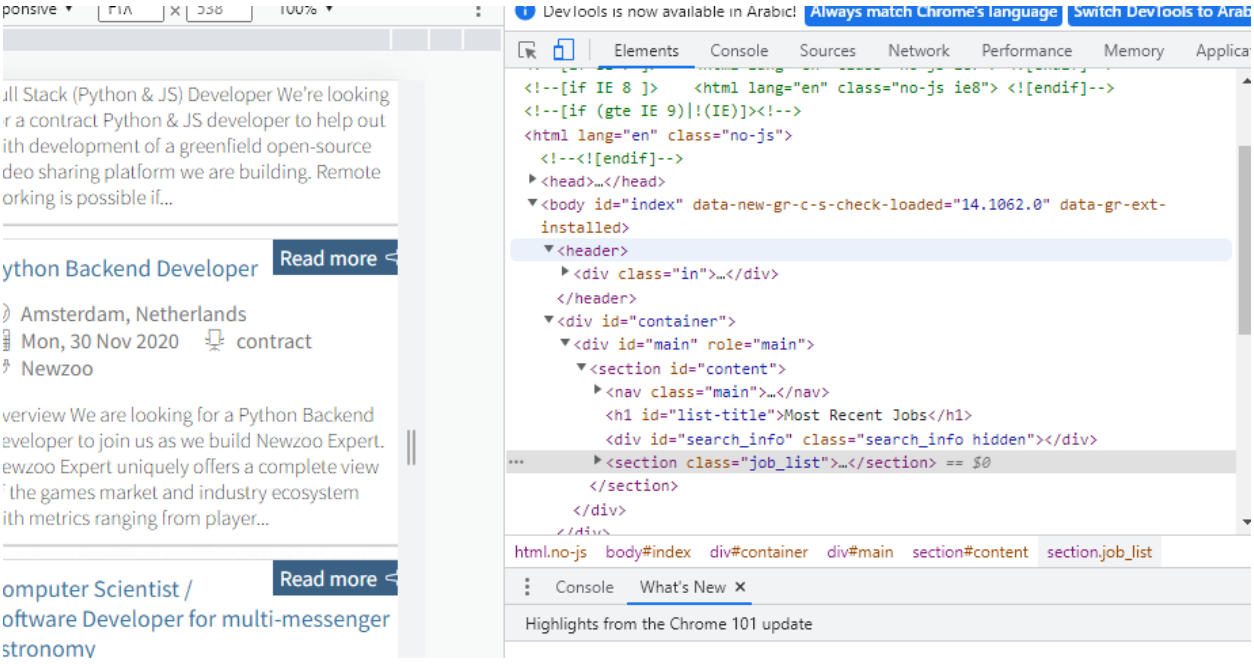
- المسمى الوظيفي
- موقع الوظيفة
- اسم المنظمة

رابط الموقع: http://pythonjobs.github.io/

الخطوة 1: تصفح وفحص موقع الويب / صفحة الويب

تتمثل المهمة الأولى والأهم أثناء تجريف البيانات من أي صفحة ويب في فتح صفحة الويب التي نستخرج منها البيانات ونفحص موقع الويب باستخدام أدوات المطور. يمكننا أيضاً عرض مصدر الصفحة.

الشكل2 واجهة الموقع



الشكل 3 فحص الصفحة

الخطوة 2: استيراد مكتبة Requests

تتيح لنا مكتبة الطلبات إرسال طلب الحصول على خادم الويب.

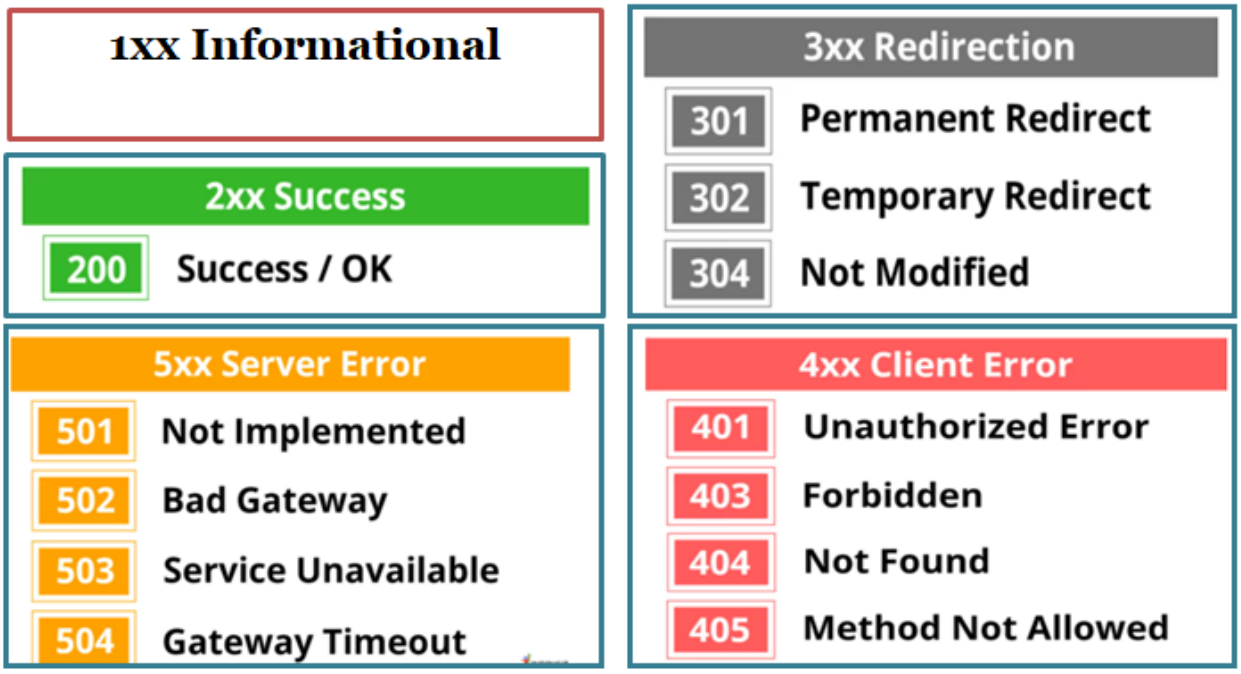
- استيراد مكتبة طلبات Python التي تتعامل مع تفاصيل طلب مواقع الويب من الخادم بتنسيق سهل المعالجة.
- نستخدم التابع request.get (...) للوصول إلى موقع الويب وتمرير عنوان URL
  "http://pythonjobs.github.io/"كبارمتر حتى يعلم التابع الموقع الذي نريد الوصول إليه.
- الوصول إلى الجسم الفعلي لطلب get (القيمة المعادة هي كائن response يحتوي أيضاً على بعض المعلومات الوصفية المفيدة مثل نوع الملف، وما إلى ذلك) وتخزينها في متغير باستخدام السمة content وبالتالي أصبح لدينا صفحة الـ html موجودة ضمن المتغير webpage.

```
import requests
# get() Request
response = requests.get("http://pythonjobs.github.io/")
# Store the webpage contents
webpage = response.content
```

التحقق من رمز الحالة

بمجرد معالجة طلب HTTP بواسطة الخادم، فإنه يرسل استجابة تحتوي على رمز الحالة. يشير رمز الحالة إلى ما إذا تمت معالجة استجابة معينة بنجاح أم لا.

هناك 5 فئات مختلفة من أكواد الحالة:



الشكل 4فئات رمز الحالة

```
print(response.status_code)
```

الخطوة 3: تحليل HTML باستخدام مكتبةBeautifulSoup

BeautifulSoupهي مكتبة Python تُستخدم لتحليل البيانات (البيانات المهيكلة) من مستندات HTML و.XML

- استيراد مكتبة.BeautifulSoup
- نقوم بإنشاء كائن .BeautifulSoup يمثل البارمتر الأول بيانات HTML بينما يمثل البارمتر الثاني المحلل اللغوي.

```
import requests
from bs4 import BeautifulSoup
# get() Request
response = requests.get("http://pythonjobs.github.io/")
# Store the webpage contents
webpage = response.content
```

```
# Check Status Code
print(response.status_code)
```

```
# Create a BeautifulSoup object out of the webpage content
soup = BeautifulSoup(webpage, "html.parser")
```

بمجرد إنشاء كائن BeautifulSoup ، نحتاج إلى استخدام الخيارات المختلفة التي توفرها لنا مكتبة BeautifulSoup للتنقل والعثور على العناصر داخل مستند HTML وكشط البيانات منه.

دعونا نلقي نظرة على الكود وبعد ذلك سوف نفهم مبدأ العمل / المنطق الكامن وراءه.

```
<h1 id="list-title">Most Recent Jobs</h1>
<div id="search_info" class="search_info hidden"></div>
<section class="job_list">
            <div class="job" data-order="0" data-slug="datadog-open-source-software-engineer-python" data-tags="python,django,flask,falcon,celery">
            <a class="go_button"
                href="/jobs/datadog-open-source-software-engineer-python.html">
                Read more <i class="i-right"></i>
            </a>
                    <h1><a href="/jobs/datadog-open-source-software-engineer-python.html">Open Source Software Engineer - Python</a></h1>

    <span class="info"><i class="i-globe"></i> New York City or Remote</span>
    <span class="info"><i class="i-calendar"></i> Thu, 03 Jun 2021</span>
    <span class="info"><i class="i-chair"></i> permanent</span>
    <span class="info"><i class="i-company"></i> Datadog</span>

    <p class="detail"> The Role In this role on our APM (tracing/profiling/debugging) team you will: Write open source code that instruments thousands


            <div class="search_match"></div>
        </div>
            <div class="job" data-order="1" data-slug="realrate-gmbh-senior-python-developer" data-tags="python,back-end,pytorch,data-science,open-sour
            <a class="go_button"
                href="/jobs/realrate-gmbh-senior-python-developer.html">
                Read more <i class="i-right"></i>
            </a>
                    <h1><a href="/jobs/realrate-gmbh-senior-python-developer.html">Senior Python Developer</a></h1>

    <span class="info"><i class="i-globe"></i> remote</span>
    <span class="info"><i class="i-calendar"></i> Sun, 11 Apr 2021</span>
    <span class="info"><i class="i-chair"></i> permanent, part-time possible</span>
    <span class="info"><i class="i-company"></i> RealRate GmbH</span>

    <p class="detail"> RealRate is Hiring Senior Python Developers! RealRate, the Artificial Intelligence rating agency is growing. We're looking for a
```

الشكل كسمات وعلامات html التي سنقرأ من خلالها

```
# The logic
for job in soup.find_all('section', class_='job_list'):
        title = [a for a in job.find_all('h1')]
        for n, tag in enumerate(job.find_all('div', class_='job')):
                company_element = [x for x in tag.find_all('span', class_='info')]
                print("Job Title: ", title[n].text.strip())
```

```
        print("Location: ", company_element[0].text.strip())
        print("Company: ", company_element[3].text.strip())
        print()
```

- في الحلقة الخارجية:

- `for job in soup.find_all('section', class_='job_list'):`

نجد العنصر الأصل ، وهو في هذه الحالة علامة section التي تحتوي على صنف HTML مع اسم الوظيفة ثم نكررها.

- نشكل المتغير title عن طريق list comprehension ويستخدم لتخزين عناوين الوظائف ونستخدم التابع job.find_all('h1') وذلك لأن العنوان في صفحة الـ html موجود ضمن العلامة h1. ويتم استخدام التابع find_all('div', class_='job') للبحث في جميع علامات div التي تحتوي على اسم الفئة job ثم تخزين البيانات في قائمة تحوي عناصر من نوع tuple كل عنصر يحوي العلامة والفهرس الخاص بها وذلك لأننا قمنا بتطبيق التابع enumerate ونضع العلامات في المتغير tag والفهرس الذي يعبر عن رقم الوظيفة في n.
- الحلقة الداخلية:

```
for n, tag in enumerate(job.find_all('div', class_='job'))
```

تحتوي على وظيفتين:

1. ابحث في جميع عناصر div عن الصنف job.
2. احتفظ بعداد كل تكرار بمساعدة تابع enumerate.

داخل الحلقة الداخلية، يخزن company_element باستخدام list comprehension جميع المحتويات الموجودة داخل علامة span مع الصنف info.

- أخيراً ، بمساعدة العداد n الخاص بتابع enumerate ، نقوم باستخراج عناصر علامة title (التي تخزن عناوين الوظائف) بمساعدة فهرسها. يتم استخراج أسماء المواقع والشركات من الفهرس 0 و 3 لقائمة company_element.

- سنقوم بتحسين الخرج باستخدام المكتبة pandas ليتم عرض البيانات على شكل جدول وبعدها سنقوم بكتابة البيانات ضمن ملف csv اسمه jobs. في البداية نقوم باستيراد المكتبة pandas ونشتق غرض اسمه pd وذلك من خلال التعليمة.

```
import pandas as pd
```

ونقوم بإنشاء ثلاث قوائم t للمسمى الوظيفي و l للموقع و c لاسم الشركة وبعدها ضمن حلقة for نستبدل تعلمية الطباعة بتعليمات الاسناد إلى قائمة باستخدام التابع append ونشكل القوائم التي تحوي المعلومات التي قمنا بجرفها من موقع الويب.

```
t=[]
l=[]
c=[]
for job in soup.find_all('section', class_='job_list'):
    title = [a for a in job.find_all('h1')]
    for n, tag in enumerate(job.find_all('div', class_='job')):
        company_element = [x for x in tag.find_all('span', class_='info')]
        t.append(title[n].text.strip())
        l.append(company_element[0].text.strip())
        c.append(company_element[3].text.strip())
```

نقوم بعدها بإنشاء متغير من نوع dictionary يحوي القيم المراد وضعها في إطار البيانات أو الجدول الخاص بمكتبة pandas وبعدها نقوم بتشكيل الجدول بالاعتماد على هذا المتغير:

```
columns={'Job Title':t,'Location':l,'Company':c}
df = pd.DataFrame(columns)
```

الآن نقوم بحفظ الجدول ضمن ملف csv وبعدها عرضه على الخرج باستخدام التعليمات التالية:

```
df.to_csv('Jobs.csv', encoding='utf-8')
df.style.format()
```

النتائج والمناقشة:

- الخرج في الحالة الأولى أي بدون استخدام مكتبة الpandas يظهر لدينا معلومات الوظائف المتاحة على الموقع الحالي وهي المسمى الوظيفي والموقع واسم الشركة :

```
200
Job Title:  Open Source Software Engineer - Python
Location:  New York City or Remote
Company:  Datadog

Job Title:  Senior Python Developer
Location:  remote
Company:  RealRate GmbH

Job Title:  Full Stack (Python & JS) Developer
Location:  Kyiv
Company:  O'Dwyer Software

Job Title:  Python Backend Developer
Location:  Amsterdam, Netherlands
Company:  Newzoo

Job Title:  Computer Scientist / Software Developer for multi-messenger astronomy
Location:  Cascina (Pisa), Tuscany, Italy
Company:  European Gravitational Observatory

Job Title:  Full Stack Engineer
Location:  Sydney, Australia
Company:  Xref

Job Title:  Remote Contractor Senior Django REST Developer
Location:  Washington, D.C., Remote
Company:  AIsoft Development LLC
```

الشكل 6 الخرج البدائي

- بعد استخدام أطر العمل الخاصة بمكتبة pandas يصبح الخرج بالشكل التالي:

Out[10]:

	Job Title	Location	Company
0	Open Source Software Engineer - Python	New York City or Remote	Datadog
1	Senior Python Developer	remote	RealRate GmbH
2	Full Stack (Python & JS) Developer	Kyiv	O'Dwyer Software
3	Python Backend Developer	Amsterdam, Netherlands	Newzoo
4	Computer Scientist / Software Developer for multi-messenger astronomy	Cascina (Pisa), Tuscany, Italy	European Gravitational Observatory
5	Full Stack Engineer	Sydney, Australia	Xref
6	Remote Contractor Senior Django REST Developer	Washington, D.C., Remote	AIsoft Development LLC

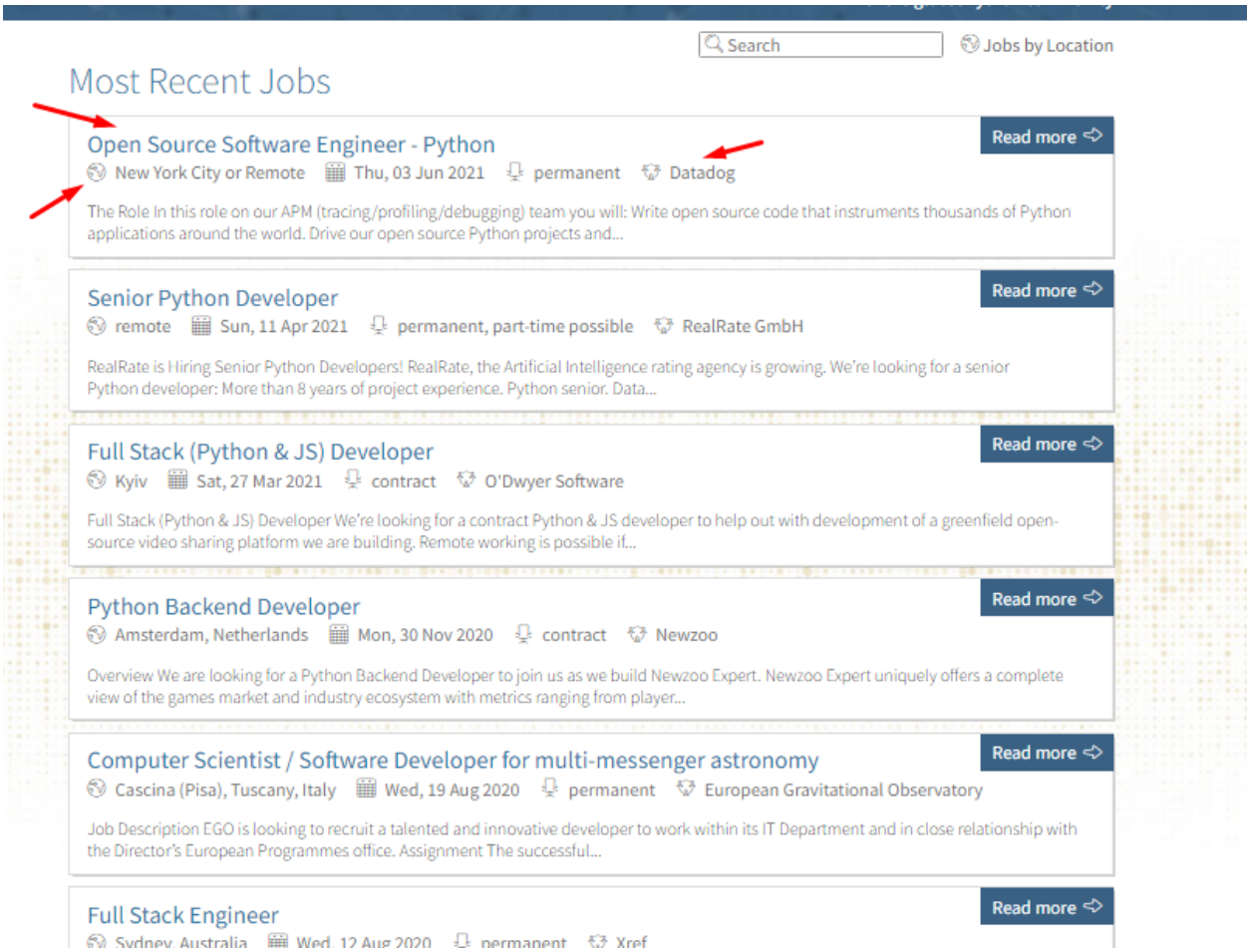
الشكل 7 الخرج باستخدام مكتبة pandas

والملف الناتج يحوي المعلومات التي تم تجريفها وهي موضحة بالشكل:

	A	B	C	D	E
1		Job Title	Location	Company	
2	0	Open Source Software Engineer - Python	New York City or Remote	Datadog	
3	1	Senior Python Developer	remote	RealRate GmbH	
4	2	Full Stack (Python & JS) Developer	Kyiv	O'Dwyer Software	
5	3	Python Backend Developer	Amsterdam, Netherlands	Newzoo	
6	4	Computer Scientist / Software Developer for multi-messenger a	Cascina (Pisa), Tuscany, It	European Gravitational Observatory	
7	5	Full Stack Engineer	Sydney, Australia	Xref	
8	6	Remote Contractor Senior Django REST Developer	Washington, D.C., Remot	AIsoft Development LLC	
9					
10					
11					

الشكل 8ملف الـ csv

في هذا الوقت كانت هذه هي الوظائف المتوفرة وإذا رغبنا رؤيتها مباشرة من الموقع نفتح نافذة الموقع كما في الشكل:



الشكل 9الوظائف المتاحة في الموقع

الاستنتاجات والتوصيات:

تجريف الويب هي عملية مفيدة جداً ولكنها تعتبر غير قانونية لأنها تسبب عبئ على المواقع وضغط على السيرفرات كون الكود يقوم بتصفح مئات الصفحات بالموقع في أقل من ثانية وهذا ما قد يؤدي إلى بعض المشاكل التقنية في الموقع. أيضاً إن الكود الخاص بتجريف موقع ويب معين لا يمكن تطبيقه على موقع آخر لأن هيكلية المواقع تختلف من حيث السمات والعلامات المستخدمة وأيضاً الكود المكتوب الخاص بموقع معين قد لا يصلح للعمل مع نفس الموقع بعد فترة زمنية معينة في حال قام أصحاب الموقع بتحديث موقعهم وتغيير هيكليته. يمكن تطوير هذا الكود ليتم استخلاص معلومات أكثر حول الوظائف وأيضاً يمكن تطويره للبحث عن وظيفة معينة ضمن الوظائف الموجودة.

المراجع:

1- https://www.pluralsight.com/guides/extracting-data-html-beautifulsoup
2- https://www.simplilearn.com/learn-basics-of-web-scraping-in-python-free-course-skillup
3- https://cj-mayes.com/2022/02/10/web-scraping-with-beautifulsoup-python/
4- https://en.wikipedia.org/wiki/Web_scraping
5- https://github.com/vprusso/youtube_tutorials/tree/master/web_scraping_and_automation/beautiful_soup
6- محاضرات الدكتور مهند عيسى, برمجة الشبكات, جامعة تشرين، 2022
7- https://docs.python-requests.org/
8- https://www.dataquest.io/blog/tutorial-an-introduction-to-python-requests-library/
9- https://www.crummy.com/software/BeautifulSoup/bs4/doc/
10- https://pandas.pydata.org/