TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**



**BÁO CÁO THỰC TẬP NHẬN THỨC**

**TÊN ĐỀ TÀI**

**CRAWL DỮ LIỆU TỰ ĐỘNG TRÊN WEBSITE**

**Nhóm: 3**

**Sinh viên thực hiện:**

1. Thành viên: Dương Việt Huy Hoàng Lớp : 45K21.2
2. Thành viên: Nguyễn Văn Hoàng Vương Lớp : 45K21.2

**Giảng viên hướng dẫn:** Nguyễn Thành Thủy

**Đà Nẵng, 6/2021**

**LỜI CAM ĐOAN**

Nhóm xin cam đoan đề tài: “*Crawl dữ liệu tự động trên web*” là một đề tài nghiên cứu độc lập của nhóm với sự hướng dẫn từ GVHD: TS.Nguyễn Thành Thủy. Đề tài là một quá trình nghiên cứu, học tập mà chúng em đã thực hiện và làm việc nhóm với nhau một cách nghiêm túc.

Trong quá trình nghiên cứu đề tài, ngoài việc học trong khóa học, nhóm chúng em có tham khảo thêm một số tài liệu nhằm củng cố kiến thức, chúng em cam kết sẽ ghi chú rõ ràng đối với những tài liệu tham khảo bên ngoài.

Nhóm chúng em xin cam đoan. Nếu phát hiện có gì sai sót trong đề tài, nhóm sẽ chịu hoàn toàn trách nhiệm.

# MỤC LỤC

[MỤC LỤC ii](#_Toc75097604)

[LỜI MỞ ĐẦU 1](#_Toc75097605)

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 2](#_Toc75097606)

[1.1. Tổng quan về đề tài 2](#_Toc75097607)

[1.1.1. Mục tiêu nghiên cứu của đề tài 2](#_Toc75097608)

[1.1.2. Nhiệm vụ của đề tài 2](#_Toc75097609)

[1.2. Tổng quan về khóa học 2](#_Toc75097610)

[CHƯƠNG 2. LÝ THUYẾT TRONG KHÓA HỌC 3](#_Toc75097611)

[2.1. Ngôn ngữ lập trình python 3](#_Toc75097612)

[2.1.1. Khái niệm Python 3](#_Toc75097613)

[2.1.2. Crawl dữ liệu với Python 4](#_Toc75097614)

[2.2. Crawl dữ liệu 4](#_Toc75097615)

[2.2.1. Giới thiệu về crawl dữ liệu 4](#_Toc75097616)

[2.2.2. Các khái niệm trong khóa học 5](#_Toc75097617)

[CHƯƠNG 3. TRIỂN KHAI THỰC HÀNH & KẾT QUẢ 6](#_Toc75097618)

[3.1. Thực hành trong khóa học 6](#_Toc75097619)

[3.1.1. Tổng quan về khóa học Web Scraping 6](#_Toc75097620)

[3.1.2. Beautifulsoup & Request 7](#_Toc75097621)

[3.1.3. Trích xuất URL 7](#_Toc75097622)

[3.1.4. Web Scraping Craigslist – Tiêu đề 10](#_Toc75097623)

[3.1.5. Web Scraping Craigslist – Trình gói chi tiết công việc 12](#_Toc75097624)

[3.1.6. Web Scraping Craigslist – Trang mô tả công việc 14](#_Toc75097625)

[3.1.7. Thu thập thông tin & chỉnh sửa các trang tiếp theo 17](#_Toc75097626)

[3.1.8. Lưu CSV 19](#_Toc75097627)

[3.2. Bài thực hành cuối khóa 22](#_Toc75097628)

[CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 26](#_Toc75097629)

[TÀI LIỆU THAM KHẢO 27](#_Toc75097630)

DANH MỤC HÌNH ẢNH

[Hình 3. 1 Giao diện Colab 6](#_Toc76738270)

[Hình 3. 2 Giao diện Pycharm 6](#_Toc76738271)

[Hình 3. 3 Cài đặt beautifulsoup & request trên colab 7](#_Toc76738272)

[Hình 3. 4 Trang web crawl dữ liệu 7](#_Toc76738273)

[Hình 3. 5 “a href” chứa liên kết đến các thẻ 9](#_Toc76738274)

[Hình 3. 6 Các liên kết ở các thẻ “a” được trích xuất 9](#_Toc76738275)

[Hình 3. 7 Tiêu đề của từng đối tượng 10](#_Toc76738276)

[Hình 3. 8 Titles sau khi trích xuất 11](#_Toc76738277)

[Hình 3. 9 Địa chỉ của từng đối tượng 11](#_Toc76738278)

[Hình 3. 10 Địa chỉ của các đối tượng sau khi trích xuất 12](#_Toc76738279)

[Hình 3. 11 Lớp chung của một đối tượng 13](#_Toc76738280)

[Hình 3. 12 Job Title, Location, Date, Link của từng đối tượng 14](#_Toc76738281)

[Hình 3. 13 Description của từng đối tượng 15](#_Toc76738282)

[Hình 3. 14 Attributes của từng đối tượng 16](#_Toc76738283)

[Hình 3. 15 Job Attributes, Job Description của một đối tượng 16](#_Toc76738284)

[Hình 3. 16 Code thực thi crawl nhiều trang 17](#_Toc76738285)

[Hình 3. 17 Đối tượng next để chuyển trang 18](#_Toc76738286)

[Hình 3. 18 Chạy kết quả toàn bộ dữ liệu ở tất cả các trang 19](#_Toc76738287)

[Hình 3. 19 Kết quả cho khi chạy code Total Jobs: 149 21](#_Toc76738288)

[Hình 3. 20 Kết quả được lưu về file csv 22](#_Toc76738289)

[Hình 3. 21 Kết quả chạy code 25](#_Toc76738290)

[Hình 3. 22 Kết quả lưu về file csv 25](#_Toc76738291)

**DANH MỤC CÁC TỪ VIẾT TẮT**

**SEO** : Search Engine Optimization

**URL** : Uniform Resource Locator

**API** : Application Programming Interface

**HTML** : **Hypertext Markup Language**

**CSV** : Comma Separated Values

**SQL** : Structured Query Language

# LỜI MỞ ĐẦU

1. **Mục tiêu nghiên cứu của đề tài**

* Đề tài này tập trung nghiên cứu về cách crawl dữ liệu từ một trang web
* Biết cách thu thập và trích xuất dữ liệu với *Web Scraping*

1. **Nhiệm vụ chính của đề tài**

* Học khóa học *Learn Web Scraping with Python from Scratch*
* Thực hành các hướng dẫn trong khóa học
* Làm bài kiểm tra cuối khóa học hoặc crawl dữ liệu từ một trang web do GVHD chỉ định

1. **Kết cấu của đề tài**

Đề tài được tổ chức gồm phần mở đầu, 4 chương bao gồm:

* Mở đầu
* **Chương 1**: **Tổng quan về đề tài**
* **Chương 2: Lý thuyết trong khóa học**
* **Chương 3: Triển khai thực hành & kết quả**
* **Chương 4: Kết luận và hướng phát triển**

# TỔNG QUAN VỀ ĐỀ TÀI

## Tổng quan về đề tài

### Mục tiêu nghiên cứu của đề tài

Thông qua đề tài đã lựa chọn, nhóm đặt mục tiêu nắm bắt được các kiến thức cơ bản của ngôn ngữ lập trình python, khái niệm về crawl dữ liệu và thực hiện trích xuất dữ liệu với kỹ thuật web scraping đã được học. Đồng thời có thể áp dụng để thực hành vào một trang web đơn giản. Cùng với đó nâng cao khả năng tự học và làm việc nhóm.

### Nhiệm vụ của đề tài

* Nghiên cứu các kiến thức trong khóa học *Learn Web Scraping with Python from Scratch* trên Udemy
* Thực hiện theo đúng kế hoạch thực tập đã được lên sẵn
* Xem các video trên khóa học, nắm được các kiến thức về crawl dữ liệu với web scraping. Tải các file hướng dẫn về để thực hành
* Thực hiện bài kiểm tra tổng kết khóa học hoặc crawl dữ liệu ở một trang web được GVHD chỉ định
* Viết báo cáo tổng kết và kết quả đạt được của đề tài, nộp tài khoản chung của khóa học cho GVHD

## Tổng quan về khóa học

Khóa học *Learn Web Scraping with Python from Scratch* trên Udemy là một khóa học hướng dẫn cách trích xuất dữ liệu từ một trang web bất kỳ trở nên dễ dàng và thuận tiện hơn. Ban đầu khi muốn lấy dữ liệu từ một trang web, ta mở trình duyệt web và tìm dữ liệu cần, cuối cùng sao chép và dán vào tệp khác trên máy. Tuy nhiên khi cần mở nhiều trang để trích xuất nhiều dữ liệu, bằng cách thủ công như trên có thể gây nhàm chán và lãng phí thời gian. Và đó là khi thao tác tìm kiếm web tự động ra đời nhằm cho quá trình trích xuất dữ liệu trở nên hiệu quả hơn, đó cũng là mục đích của khóa học.

# LÝ THUYẾT TRONG KHÓA HỌC

## Ngôn ngữ lập trình python



Hình 2. Ngôn ngữ lập trình Python

### Khái niệm Python

Python là ngôn ngữ lập trình hướng đối tượng đa năng có cấu trúc dữ liệu cấp cao mạnh mẽ và hệ thống thư viện lớn. Bắt đầu được thiết kế bởi Guido Van Rossum và ra mắt vào năm 1991 khi ông muốn tạo ra một ngôn ngữ lập trình đơn giản, dễ hiểu. Nó dễ dàng để tìm hiểu và đang nổi lên như một trong những ngôn ngữ lập trình nhập môn tốt nhất cho người lần đầu tiếp xúc với ngôn ngữ lập trình.

Python hiện có 2 version phổ biến đó chính là python 2 và python 3. Python có [cấu trúc dữ liệu](https://quantrimang.com/cau-truc-du-lieu-va-giai-thuat) cấp cao mạnh mẽ và cách tiếp cận đơn giản nhưng hiệu quả đối với lập trình hướng đối tượng. Cú pháp lệnh của python là điểm cộng vô cùng lớn vì sự rõ ràng, nó giúp người dùng tạo ra những chương trình hay với số lượng dòng code tối thiểu và phát triển ứng dụng trong nhiều lĩnh vực, ở hầu hết các nền tảng.

### Crawl dữ liệu với Python

Lý do nhóm chọn Python làm ngôn ngữ thực hiện crawl:

1. Python đơn giản và nhất quán

Python cho phép viết code ngắn gọn và dễ đọc. Đối với các thuật toán phức tạp thì việc crawl với python là giải pháp đơn giản cho phép việc đọc hiểu code một cách dễ dàng hơn. Nhóm sẽ tập trung được thời gian, trí lực vào việc tìm hiểu các dòng code để crawl dữ liệu thay vì gặp rắc rối với kỹ thuật của ngôn ngữ.

1. Python có nhiều lựa chọn về thư viện và framework

Python có một lượng thư viện khổng lồ do cộng đồng lập trình viên chia sẻ. Việc sử dụng các framework/thư viện có sẵn phục vụ crawl như beautifulsoup, requests, pandas, scrapy,…có thể hiểu là các code được viết sẵn mà ta có thể sử dụng ngay để crawl dữ liệu. Từ đó rút ngắn được thời gian học và tăng hiệu quả tiếp thu của khóa học.

## Crawl dữ liệu

### Giới thiệu về crawl dữ liệu

Crawl dữ liệu hay còn gọi là cào dữ liệu là một thuật ngữ quen thuộc trong ngành marketing và SEO. Nó được sử dụng phổ biến trong các công cụ tìm kiếm như Google, Bing, Baidu, Yahoo…Công việc chính của crawl là thu thập dữ liệu từ một trang web bất kỳ. Rồi tiến hành phân tích mã nguồn HTML để đọc dữ liệu. Và lọc ra theo yêu cầu người dùng hoặc dữ liệu mà Search Engine yêu cầu.

Trong crawl dữ liệu chia thành 2 phần là *Web crawler* và *Web scraping*:

* Web crawler được hiểu là thuật ngữ kỹ thuật dùng để chỉ quá trình thu thập nội dung và dữ liệu từ các website trên mạng internet theo các đường links cho trước. Giúp thu thập thông tin của những trang web đó về cho công cụ tìm kiếm
* Web scraping là thuật ngữ kỹ thuật để chỉ quá trình sử dụng bot để trích xuất nội dung và dữ liệu từ một trang web. Thông tin thu thập được sẽ xuất thành định dạng hữu ích hơn cho người dùng. Có thể là bảng tính hoặc API

Và trong đề tài nghiên cứu lần này nhóm sẽ tiến hành crawl dữ liệu với web scraping bằng việc thu thập và trích xuất những thông tin cần thiết từ một trang web bất kỳ

### Các khái niệm trong khóa học

1. Thư viện Beautifulsoup & Request

* Thư viện Beautifulsoup: là một thư viện của python cho phép chúng ta lấy dữ liệu từ HTML đơn giản và hiệu quả. Cụ thể trong khóa học là thư viện Beautifulsoup 4 với Python 3
* Thư viện Request: là một thư viện đơn giản của python cho phép thực hiện các tác vụ như gửi request từ client lên server và nhận response (server trả kết quả về cho client) một cách đơn giản

1. URL, HTML

* URL: là địa chỉ của một tài nguyên duy nhất trên Web. Mỗi URL hợp lệ sẽ trỏ đến một tài nguyên duy nhất, tài nguyên đó có thể là trang HTML, hình ảnh, video, file PDF, …
* HTML: là ngôn ngữ đánh dấu siêu văn bản, giúp người dùng tạo bố cục và cấu trúc các thành phần trong trang web làm trang web trở thành hệ thống hoàn chỉnh và là ngôn ngữ để hiện thị nội dung cho người truy cập trang web trên World Wide Web.

1. Thư viện Pandas: là một thư viện mã nguồn mở trong python hỗ trợ đắc lực cho các thao tác với dữ liệu. Là một công cụ cho phép đọc, ghi dữ liệu giữa bộ nhớ và nhiều định dạng file: csv, text, excel, sql,…

# TRIỂN KHAI THỰC HÀNH & KẾT QUẢ

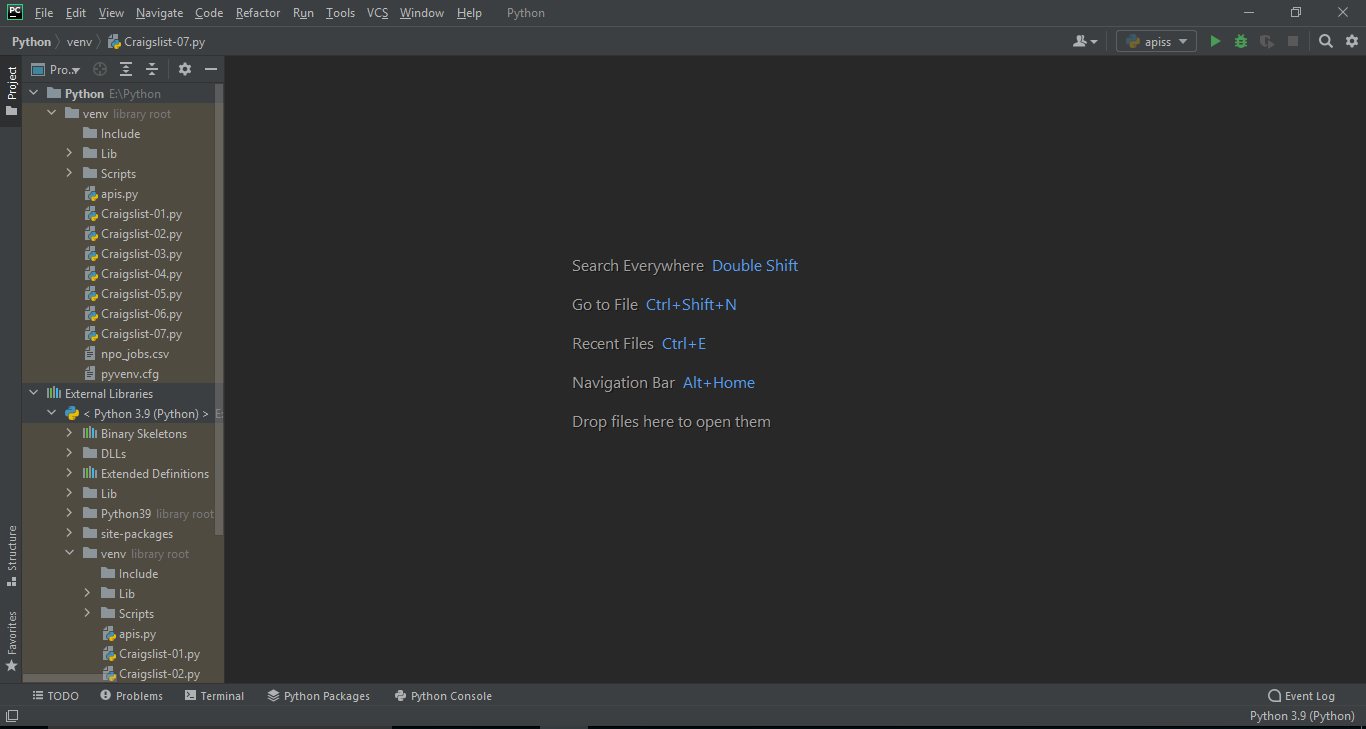
## Thực hành trong khóa học

### Tổng quan về khóa học Web Scraping

Trong khóa học này nhóm tiến hành chạy code trên hai nền tảng là Google Colab và Pycharm (hoặc VS Code)



Hình 3. Giao diện Colab

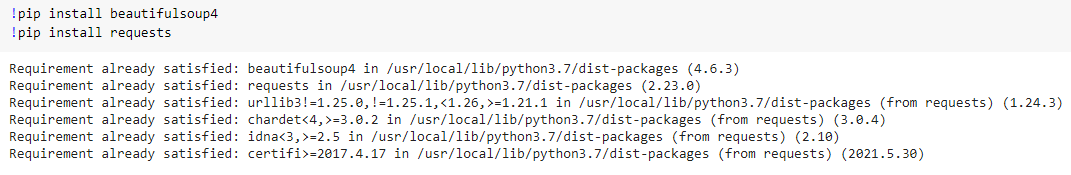


Hình 3. Giao diện Pycharm

### Beautifulsoup & Request

Để cài đặt beautifulsoup và request nhóm sử dụng câu lệnh:

“Pip install beautifullsoup4” và “Pip install requests”. Ngoài ra đối với Colab thì phải thêm “!” vào trước câu lệnh



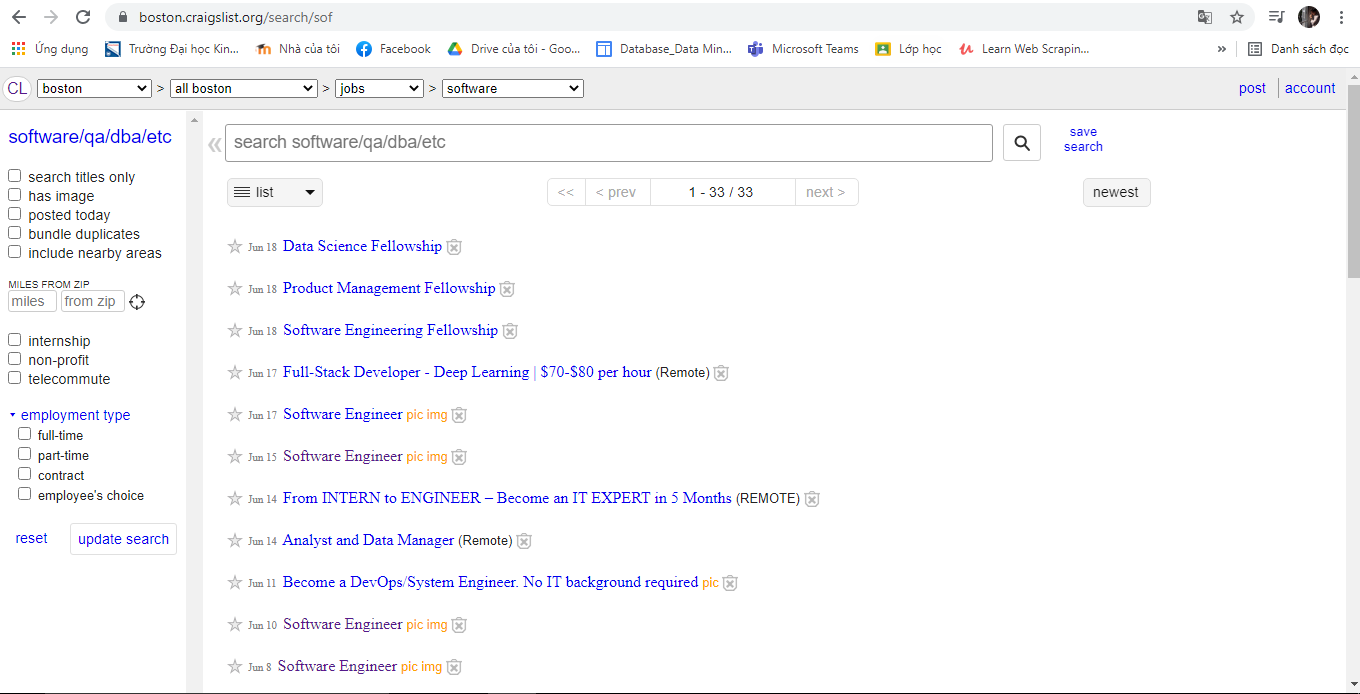
Hình 3. Cài đặt beautifulsoup & request trên colab

Sau khi cài đặt beautifulsoup và request ta tiến hành chạy nó để sử dụng

Với dòng lệnh →

### Trích xuất URL

Nhóm chọn trang web để crawl dữ liệu:

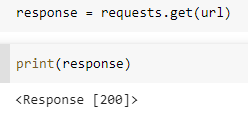


Hình 3. Trang web crawl dữ liệu

Đầu tiên nhóm tiến hành lưu địa chỉ trang web trên bằng URL:



Tiếp theo là tạo phản hồi từ trang web tới máy bằng đoạn code:



Khi in kết quả phản hồi, đầu ra 200 có nghĩa là yêu cầu URL đã thực hiện thành công và internet vẫn đang hoạt động tốt.

Tiếp theo là giải nén mã nguồn của trang web vào bộ nhớ, dữ liệu sau khi giải nén sẽ ở dạng văn bản:

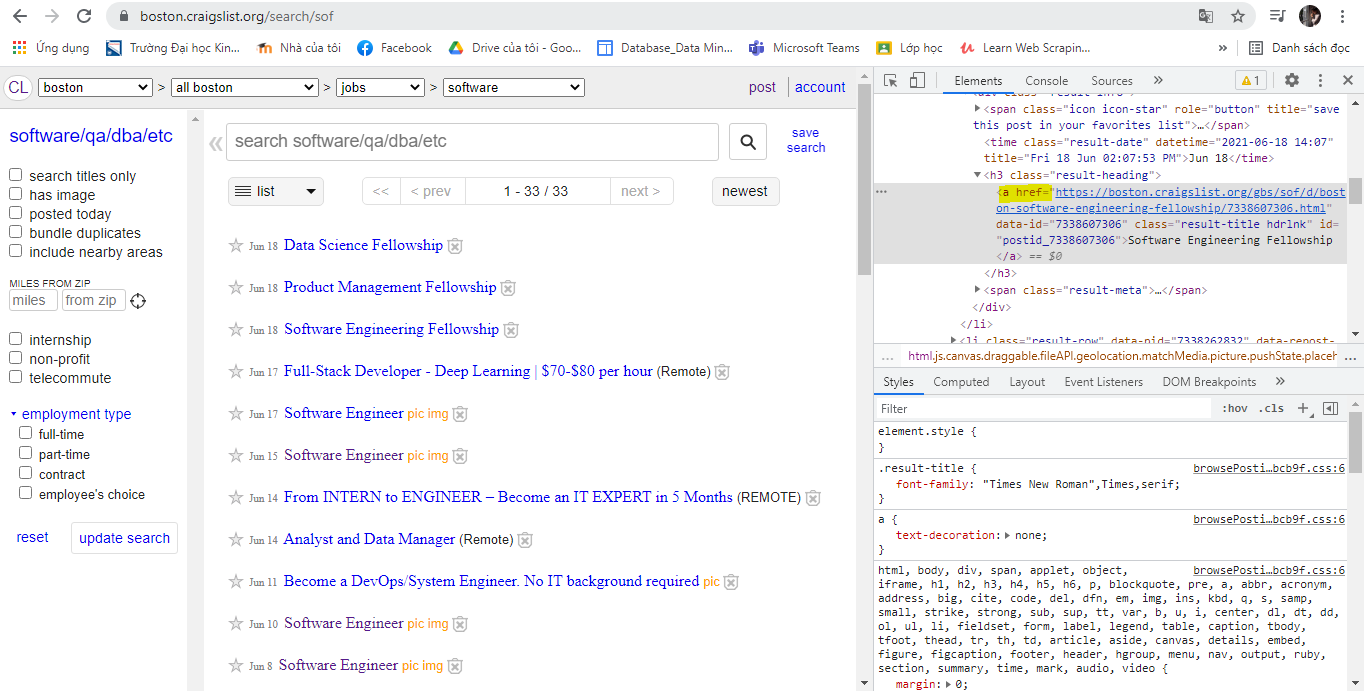


Sau đó chuyển mã nguồn vào soup để sắp xếp cấu trúc dữ liệu của trang web dưới dạng văn bản html:

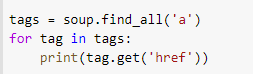


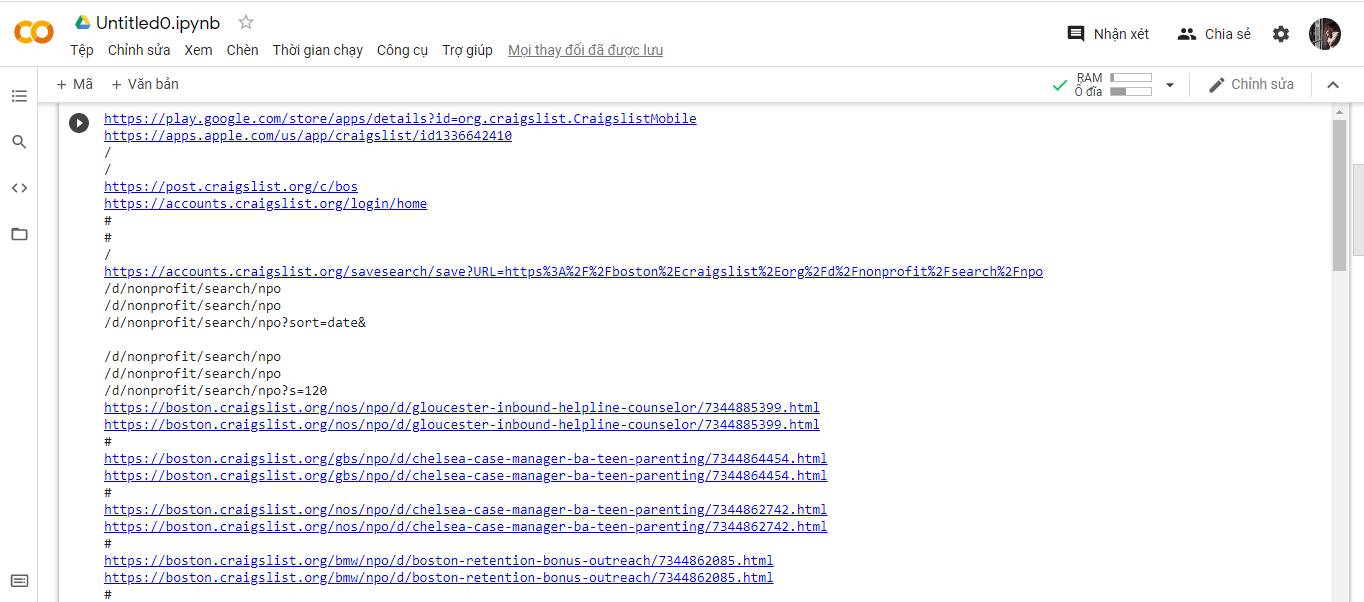
“html.parser” có chức năng cấu trúc lại bố cục và kiểu thụt lề của đánh dấu kết quả

Cuối cùng nhóm đã có thể trích xuất các thẻ html, cụ thể là đường link tới các đối tượng ở một trang web:



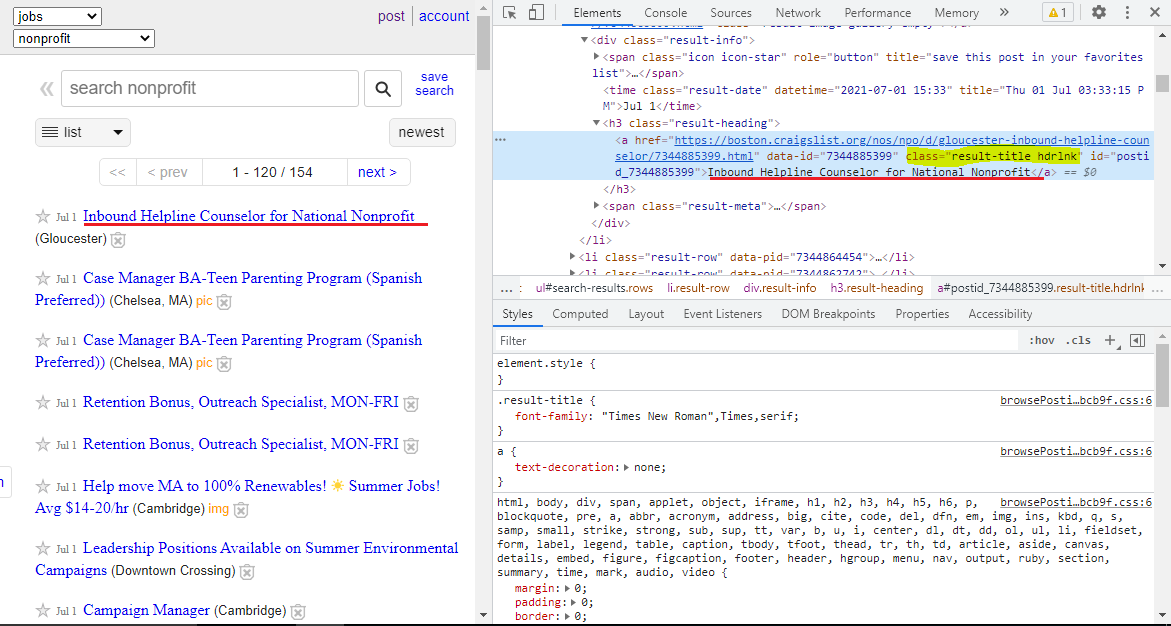
Hình 3. “a href” chứa liên kết đến các thẻ

Trích xuất các thẻ “a” có cùng thuộc tính “href” bằng cách sử dụng vòng lặp



Hình 3. Các liên kết ở các thẻ “a” được trích xuất

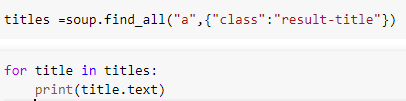
### Web Scraping Craigslist – Tiêu đề



Hình 3. Tiêu đề của từng đối tượng

Để trích xuất title (tiêu đề) của các đối tượng trong trang web thì cần phải chú ý tới thuộc tính ***class=“result-title”***

Sau đó trích xuất titles có cùng thuộc tính bằng đoạn code:



titles = tất cả đối tượng ở soup đã được xử lí ở phần 3.1.3 có cùng lớp “a” và thuộc tính ***class=“result-title”***

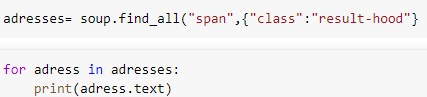


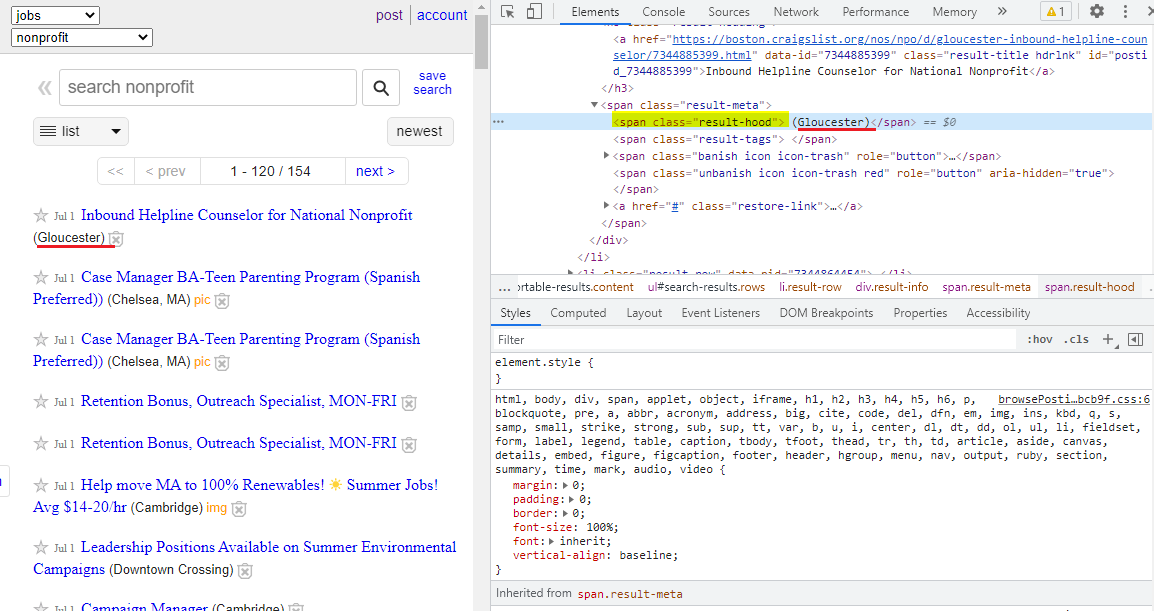
Sau đó sử dụng vòng lặp for và in ra tất cả các title



Hình 3. Titles sau khi trích xuất

Tiếp theo là một đoạn code tương tự trích xuất địa chỉ của các title có cùng lớp “span” và thuộc tính ***class=“result-hood”***





Hình 3. Địa chỉ của từng đối tượng

Và dưới đây là kết quả đạt được khi chạy

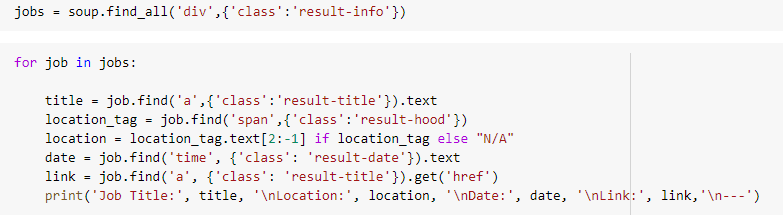


Hình 3. Địa chỉ của các đối tượng sau khi trích xuất

### Web Scraping Craigslist – Trình gói chi tiết công việc

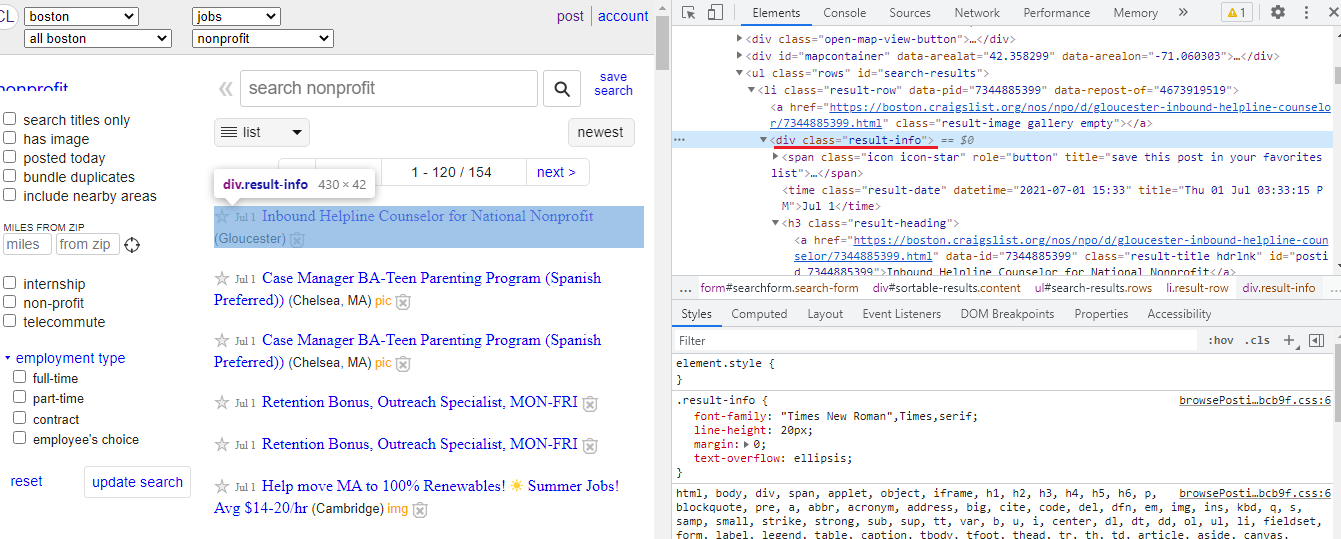
Đến với phần này, nhóm sẽ crawl dữ liệu bên ngoài của từng đối tượng trong trang web một cách thống nhất với nhau

Dưới đây là đoạn code giúp thực hiện điều này



Jobs (thể hiện chung cho nguyên một đối tượng) bao gồm title, location, date, link.

Để biết được phần lớp chung của một đối tượng, ta bôi đen đối tượng 🡪 click chuột trái 🡪 kiểm tra. Sẽ ra được lớp chung cần tìm, bên dưới là kết quả:



Hình 3. Lớp chung của một đối tượng

Tiếp theo sử dụng vòng lặp for để crawl các dữ liệu của một đối tượng đã được bôi xanh ở trên:



Trích xuất title của đối tượng có cùng lớp “a” và thuộc tính ***class=“result-title”*** lưu vào text (văn bản)

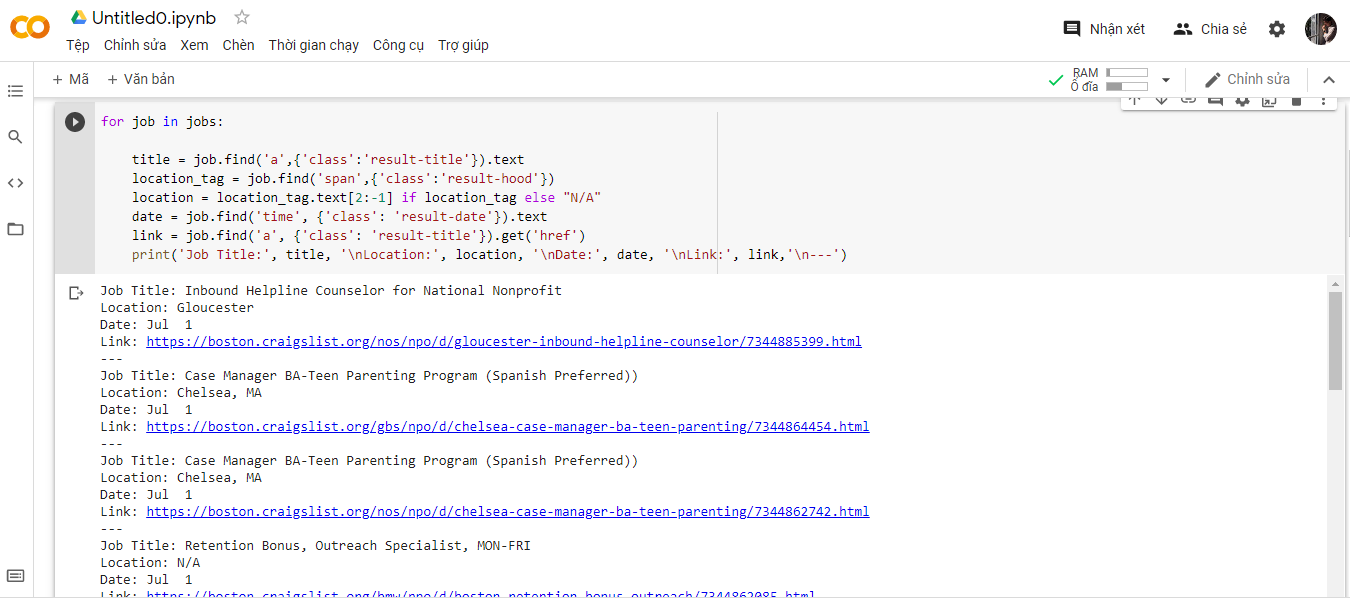


Trích xuất location của đối tượng có cùng lớp “span” và thuộc tính ***class=“result-hood”*** , tuy nhiên để kết quả in ra đẹp và dễ nhìn, ta sử dụng lệnh .text[2: -1] để loại bỏ những khoảng trống và ngoặc tròn không cần thiết ở hai đầu location. Sẽ có đối tượng có hoặc không có location nên ta sử dụng lệnh if 🡪 nếu có thì in ra văn bản 🡪 nếu không có thì “N/A”.



Trích xuất link của đối tượng có cùng lớp “a” và thuộc tính ***class=“result-title”*** và lưu link bằng get (‘href’).

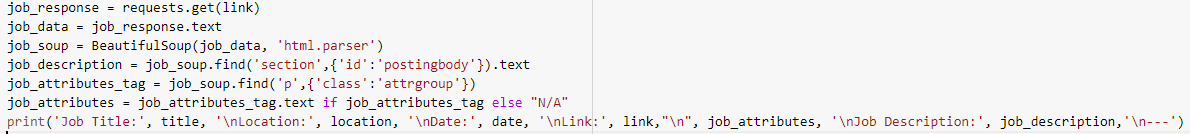
Và đây là kết quả khi chạy code:



Hình 3. Job Title, Location, Date, Link của từng đối tượng

### Web Scraping Craigslist – Trang mô tả công việc

Ở Phần trước, nhóm đã tiến hành việc crawl dữ liệu bên ngoài của từng đối tượng và đối với phần này, nhóm sẽ thực hiện việc crawl dữ liệu bên trong của từng đối tượng bằng đoạn code được bổ sung như sau:



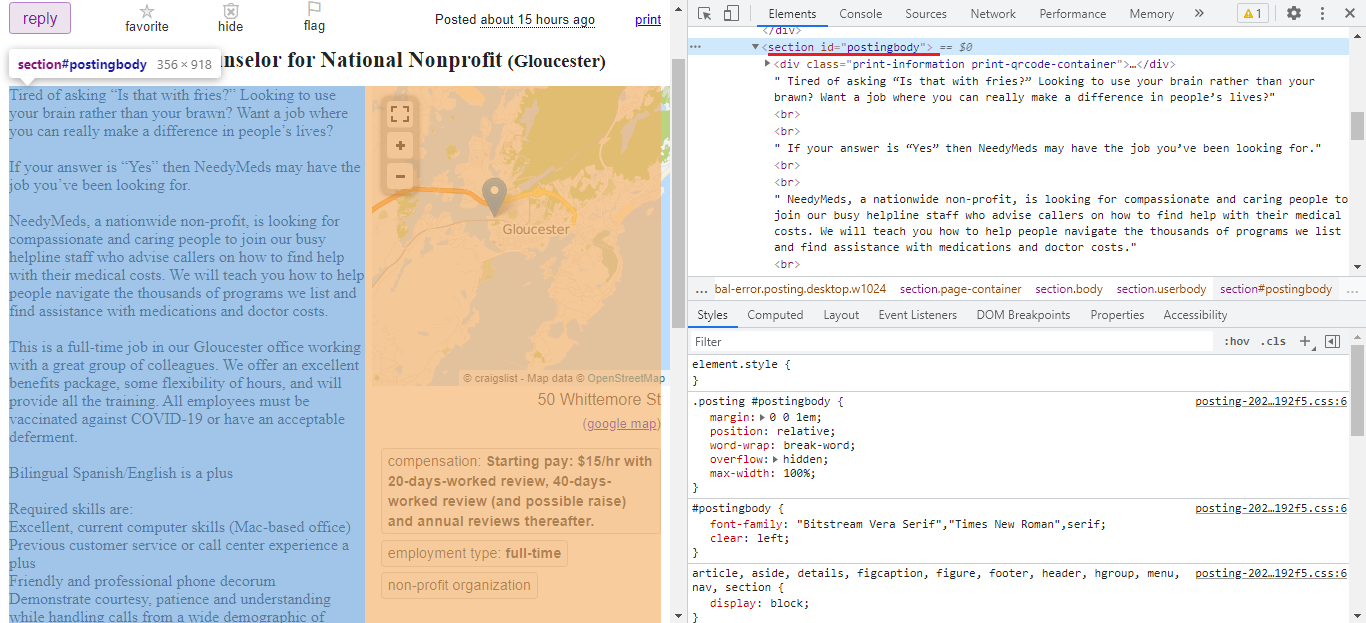
job\_response được sử dụng để liên kết với link của từng đối tượng đã được crawl. Job\_data và job\_soup giúp thu thập và lưu dữ liệu vào thư viện BeatifulSoup dưới dạng văn bản html đã được chỉnh sửa.



Tiếp theo là trích xuất mô tả công việc(description) của từng đối tượng có lớp “section” và thuộc tính ***id=“postingbody”*** và lưu vào text



Để tìm kiếm description trên ta truy cập vào đối tượng 🡪 click chuột trái ở phần mô tả công việc 🡪 kiểm tra. Sẽ ra được lớp cần tìm, bên dưới là kết quả:

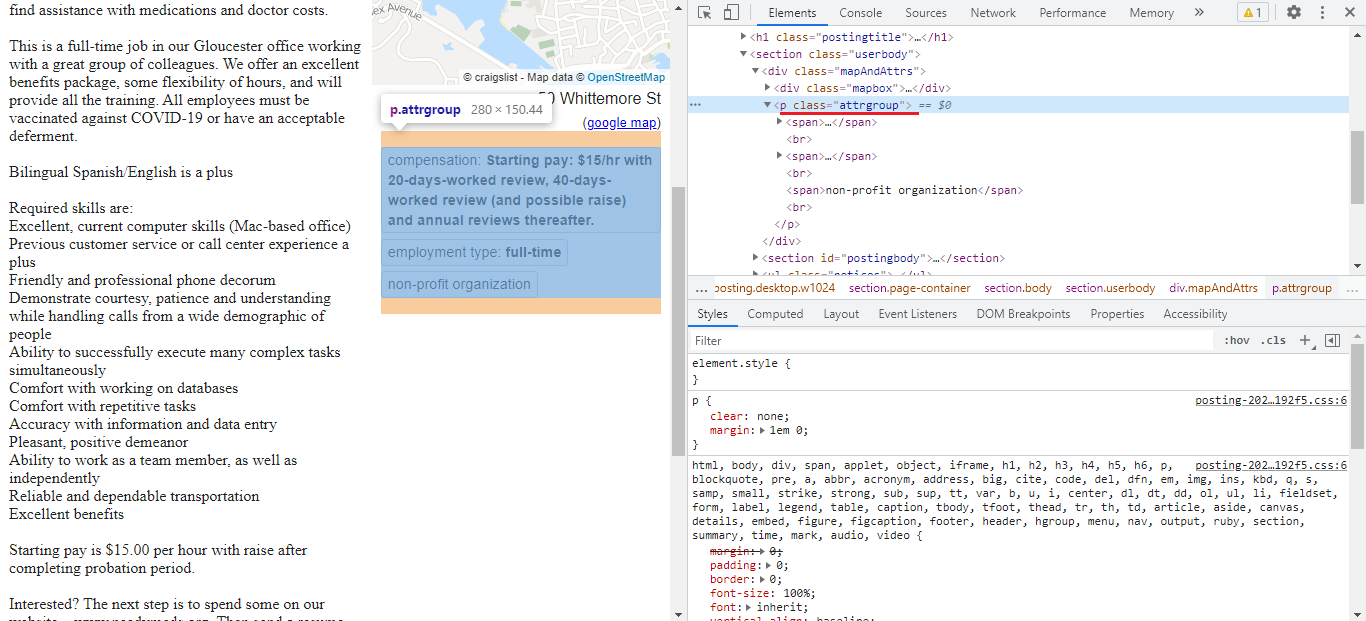


Hình 3. Description của từng đối tượng

Cuối cùng là trích xuất các thuộc tính công việc như compensation, employment type,… của từng đối tượng có lớp “p” và thuộc tính ***class=“attrgroup”.*** Sẽ có đối tượng có hoặc không có attributes nên ta sử dụng lệnh if 🡪 nếu có thì in ra văn bản text 🡪 nếu không có thì “N/A”.

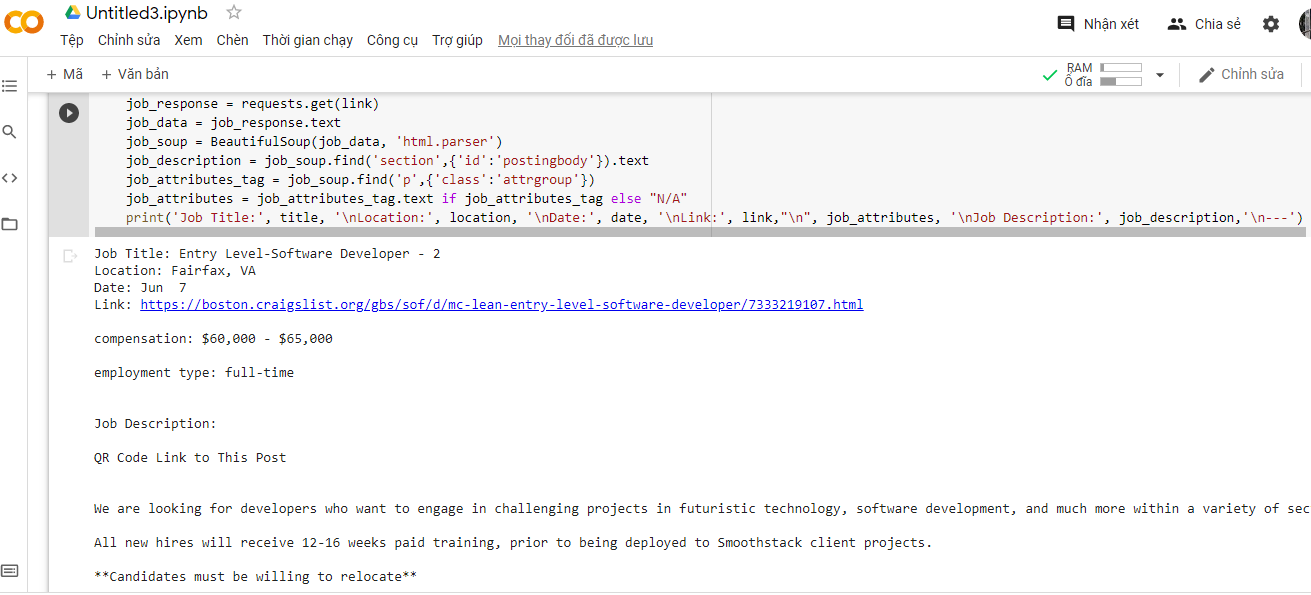


Để tìm kiếm attributes trên ta truy cập vào đối tượng 🡪 click chuột trái ở phần thuộc tính công việc 🡪 kiểm tra. Sẽ ra được lớp cần tìm, bên dưới là kết quả:



Hình 3. Attributes của từng đối tượng

Đây là kết quả khi ta chạy code:



Hình 3. Job Attributes, Job Description của một đối tượng

### Thu thập thông tin & chỉnh sửa các trang tiếp theo

Vậy là nhóm đã thực hiện xong việc crawl dữ liệu trong một trang. Ở phần kiến thức này nhóm sẽ tiến hành việc lấy dữ liệu của tất cả các trang trong một trang web. Ta có đoạn code như sau:

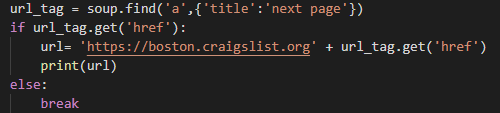


Hình 3. 16 Code thực thi crawl nhiều trang

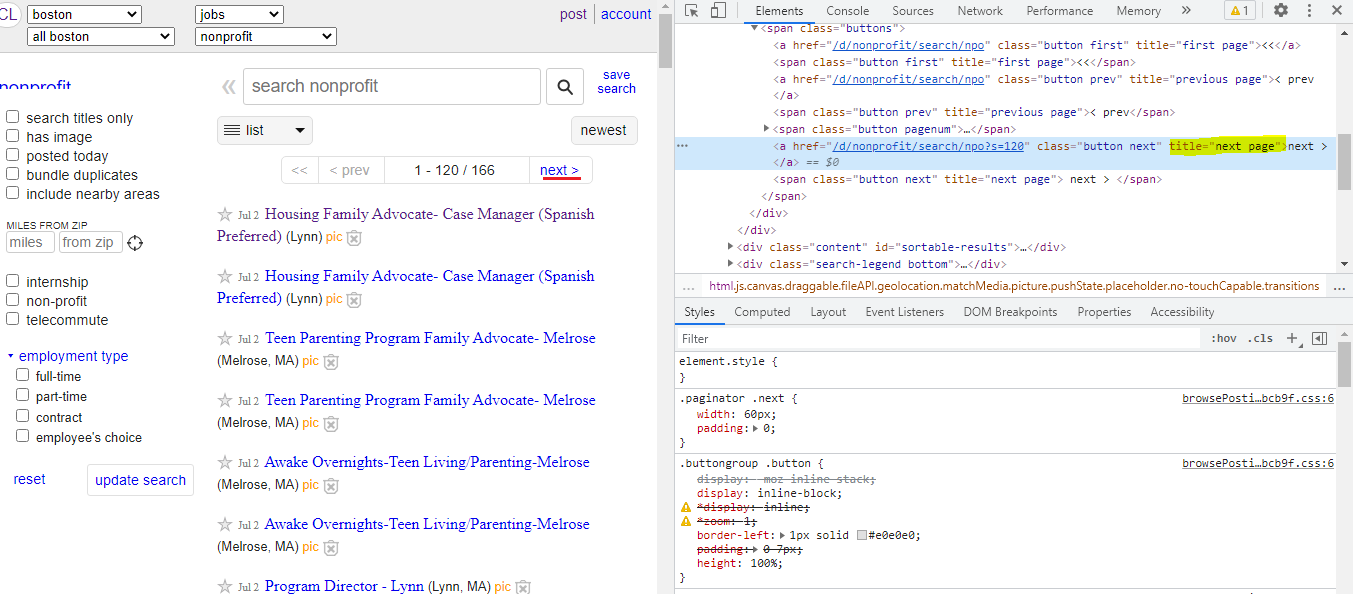
Nhóm đưa toàn bộ code để crawl dữ liệu từ một trang vào vòng lặp *While True:* để có thể chuyển trang cho đến khi yêu cầu dừng.

Thêm *job\_no = 0* và *job\_no+=1* vào trong vòng lặp để có thể đếm số trang đã crawl được.

Để có thể chuyển sang các trang kế tiếp, nhóm nhập code:



Url\_tag là đối tượng chuyển trang có lớp “a” và thuộc tính ***title=“next page”.*** Dễ dàng tìm thấy đối tượng này bằng cách click chuột trái ở “next” 🡪 kiểm tra. Sẽ ra được lớp cần tìm, bên dưới là kết quả:



Hình 3. Đối tượng next để chuyển trang

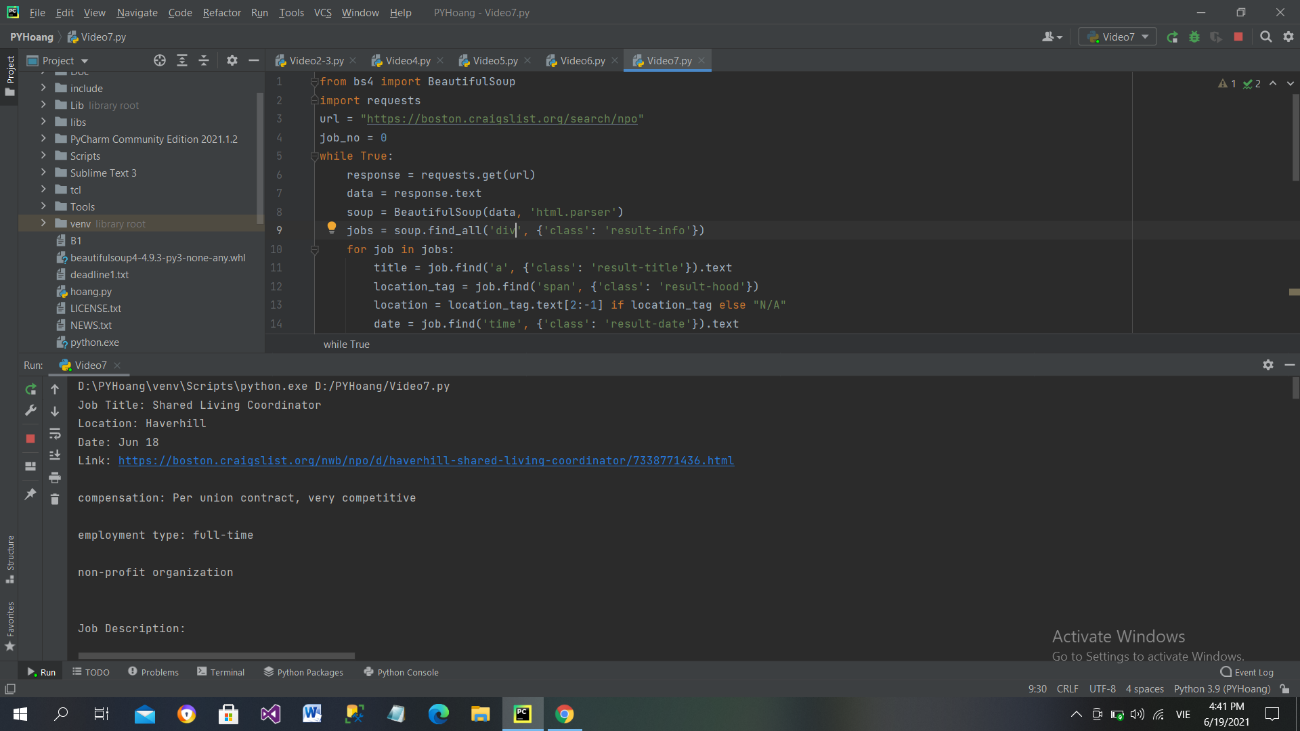
Sau đó dùng câu lệnh “if” với điều kiện gán địa chỉ url\_tag bằng địa chỉ của link trang tiếp theo.

Ở bên trong câu lệnh if, ta đổi url = url của trang chủ + địa chỉ url\_tag.



Sau mỗi lần thay đổi url thành url của trang tiếp theo, kết hợp với vòng lặp *While True:* sẽ giúp thực hiện việc crawl dữ liệu ở từng trang một cho đến khi hết toàn bộ các trang. Đây được gọi là crawl dữ liệu tự động. Sau khi hết trang sẽ *break* để kết thúc vòng lặp

Và ta có kết quả của chạy code ở hình bên dưới:



Hình 3. Chạy kết quả toàn bộ dữ liệu ở tất cả các trang

### Lưu CSV

Nội dung cuối cùng của khóa học là lưu các dữ liệu về file csv (execl) bằng cách chạy đoạn code sau:

from bs4 import BeautifulSoup

import requests

import pandas as pd

url = "https://boston.craigslist.org/search/npo"

npo\_jobs = {}

job\_no = 0

while True:

response = requests.get(url)

  data = response.text

  soup = BeautifulSoup(data,'html.parser')

  jobs = soup.find\_all('div',{'class':'result-info'})

  for job in jobs:

   title = job.find('a',{'class':'result-title'}).text

    location\_tag = job.find('span',{'class':'result-hood'})

    location = location\_tag.text[2:1] if location\_tag else "N/A"

    date = job.find('time', {'class': 'result-date'}).text

    link = job.find('a', {'class': 'result-title'}).get('href')

    job\_response = requests.get(link)

    job\_data = job\_response.text

    job\_soup = BeautifulSoup(job\_data, 'html.parser')

    job\_description = job\_soup.find('section',{'id':'postingbody'}) .text

    job\_attributes\_tag = job\_soup.find('p',{'class':'attrgroup'})

    job\_attributes = job\_attributes\_tag.text if job\_attributes\_tag else "N/A"

    job\_no += 1

    npo\_jobs[job\_no] = [title, location, date, link, job\_attributes, job\_description]

    #print('Job Title:', title, '\nLocation:', location, '\nDate:', date, '\nLink:', link,"\n", job\_attributes, '\nJob Description:', job\_description,'\n---')

  url\_tag = soup.find('a',{'title':'next page'})

    if url\_tag.get('href'):

     url= 'https://boston.craigslist.org' + url\_tag.get('href')

      print(url)

    else:

      break

print("Total Jobs:", job\_no)

npo\_jobs\_df = pd.DataFrame.from\_dict(npo\_jobs, orient = 'index', columns = ['Job Title','Location','Date', 'Link', 'Job Attributes', 'Job Description'])

npo\_jobs\_df.to\_csv('npo\_jobs.csv')

Để thực hiện lưu dữ liệu về file csv. Nhóm tiến hành cài đặt thư viện Pandas.

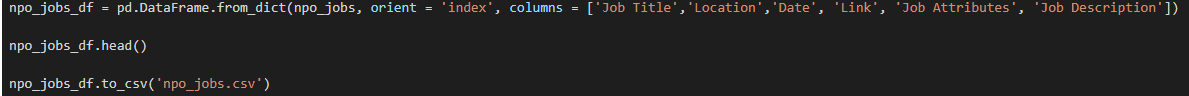
Pandas như chúng ta đã biết là một công cụ cho phép đọc, ghi dữ liệu giữa bộ nhớ và nhiều định dạng file: csv, text, excel, sql … bằng dòng lệnh



Trước khi vào vòng lặp While, tạo một thư viện trống tên là npo\_jobs = {}. Sau mỗi lần crawl từ một trang thì dữ liệu sẽ được lưu lại thư viện này. Ở bên trong vòng lặp nhóm khai báo giá trị cho thư viện lần lượt như sau:



Tiếp theo nhóm thêm dòng lệnh dưới đây:



Dòng đầu tiên(npo\_jobs\_df) có chức năng tạo khung cho thư viện. Ta sử dụng “pd.DataFrame.from\_dict” để xử lý dữ liệu dễ dàng hơn và chỉ định từ điển npo\_jobs để chuyển đổi sang một khung dữ liệu pandas.

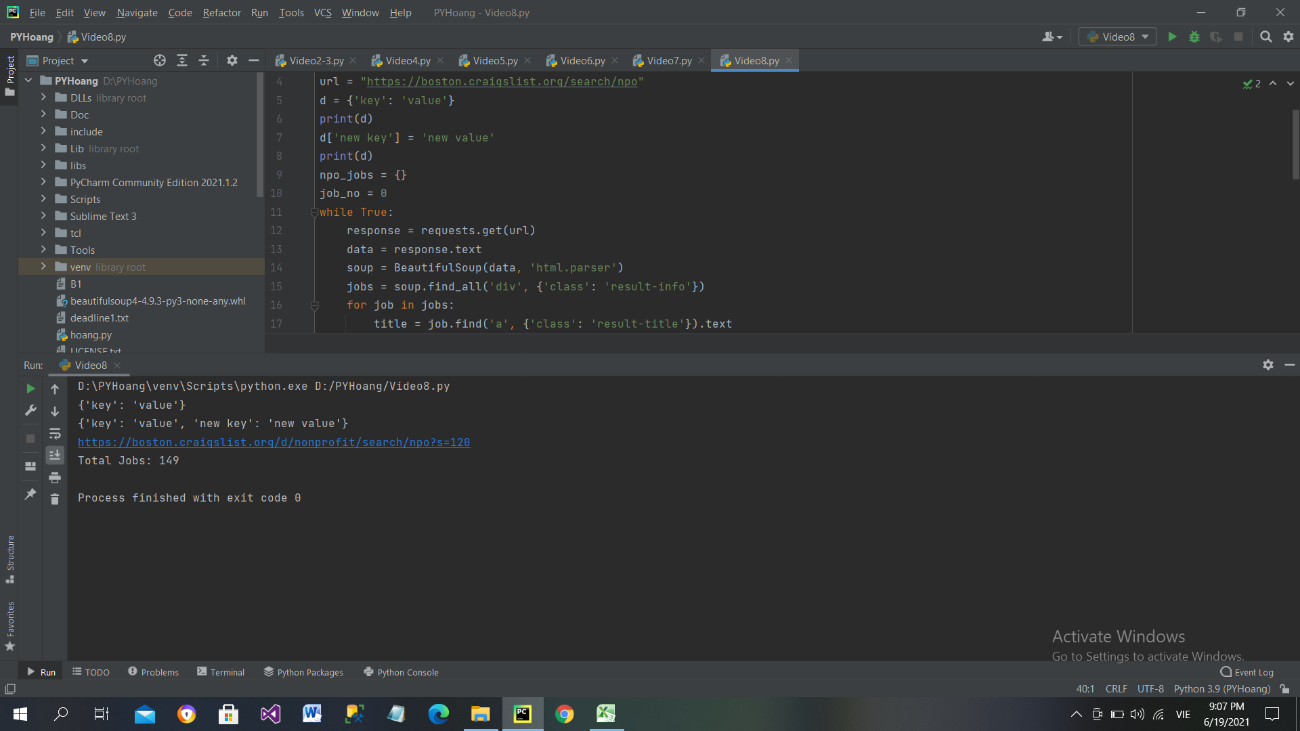
Ta có Orient = ‘index’ có nghĩa là mỗi khóa và giá trị tạo thành một hàng. Colums (tên của các cột).



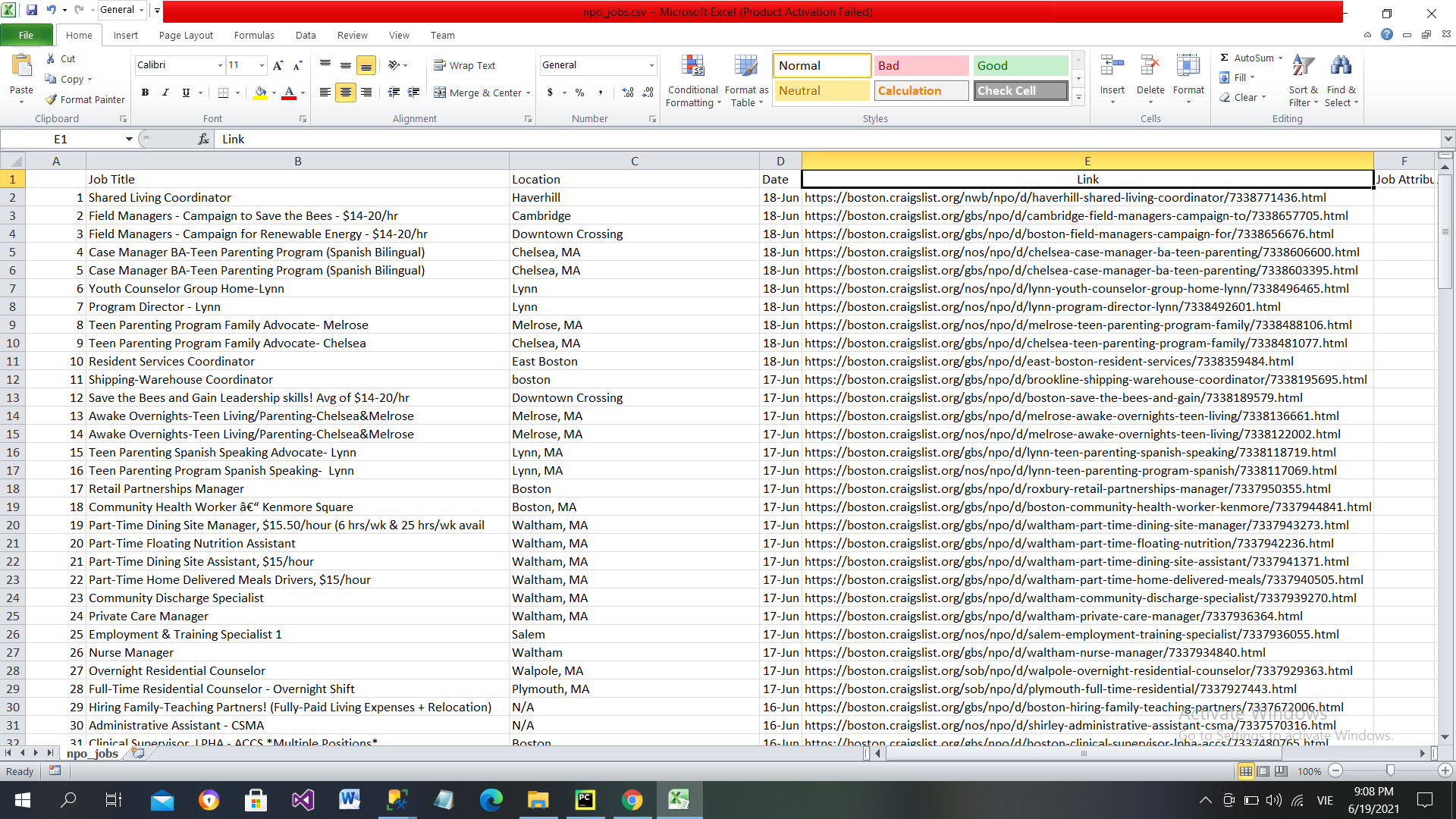
Có chức năng kiểm tra khung dữ liệu vừa được đưa vào.



Là tạo file csv (Tên file + đuôi csv)



Hình 3. Kết quả cho khi chạy code Total Jobs: 149



Hình 3. Kết quả được lưu về file csv

## Bài thực hành cuối khóa

**Đề:**

Dựa trên những gì bạn đã nghiên cứu trong khóa học, Hãy phân loại danh sách API trên trang này và xuất kết quả của bạn thành tệp CSV.

https://www.programmableweb.com/category/all/apis *hoặc là:*

https://www.programmableweb.com/category/tools/api

Các chi tiết cần crawl dữ liệu từ mỗi hàng trong bảng:

• Tên API

• URL API

• Danh mục API

• Mô tả API

Trên máy của bạn, hãy viết mã và kiểm tra nó để đảm bảo rằng nó đang hoạt động. Sau đó, sao chép, dán mã cuối cùng của bạn

Code:

from bs4 import BeautifulSoup

import requests

import pandas as pd

url = "https://www.programmableweb.com"

url\_category = "/category/all/apis"

npage = 0

api\_no = 0

api\_pb = {}

while True:

    response = requests.get(url + url\_category)

    data = response.text

    soup = BeautifulSoup(data, "html.parser")

    apis = soup.find\_all("tr")

    for api in apis:

        title\_td = api.find("td",{"class":"views-field views-field-pw-version-title"})

        if(title\_td):

            api\_name = title\_td.find("a").text

            api\_url = url + title\_td.find("a").get("href")

            description = api.find("td", {"class": "views-field-field-api-description"}).text

            category\_td = api.find("td",{"class":"views-field-field-article-primary-category"})

            category\_a = category\_td.find("a")

            if category\_a is not None:

                category = category\_a.text

            else:

                category = ""

            api\_no += 1

            api\_pb[api\_no] = [api\_name, api\_url, description, category]

            #print('API Name:', api\_name, '\nAPI URL:', api\_url, '\nDescription:', description, '\nCategory:', category, '\n---')

    url\_tag = soup.find('a',{'title':'Go to next page'})

    if url\_tag is not None:

        if (npage < 5):

            url\_category = url\_tag.get('href')

            print(url + url\_category)

            npage += 1

        else:

            break

    else:

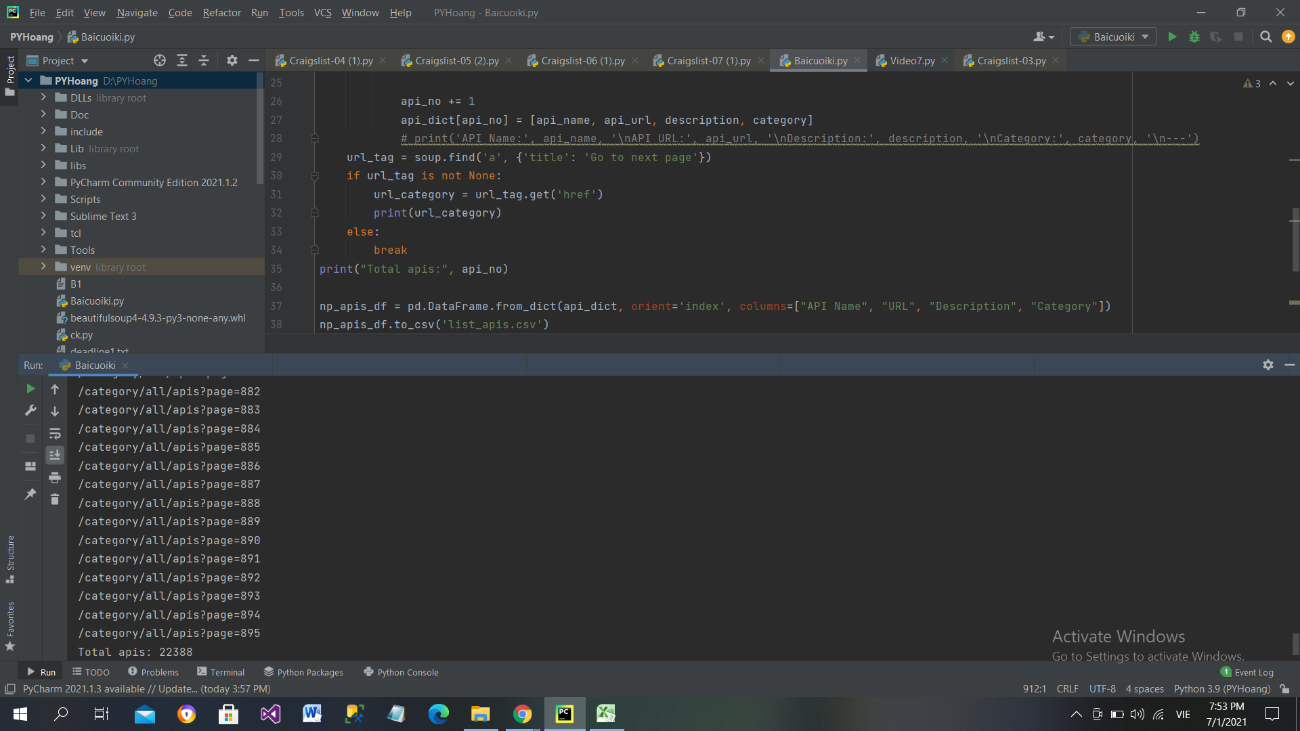
        break

print("Total apis:", api\_no)

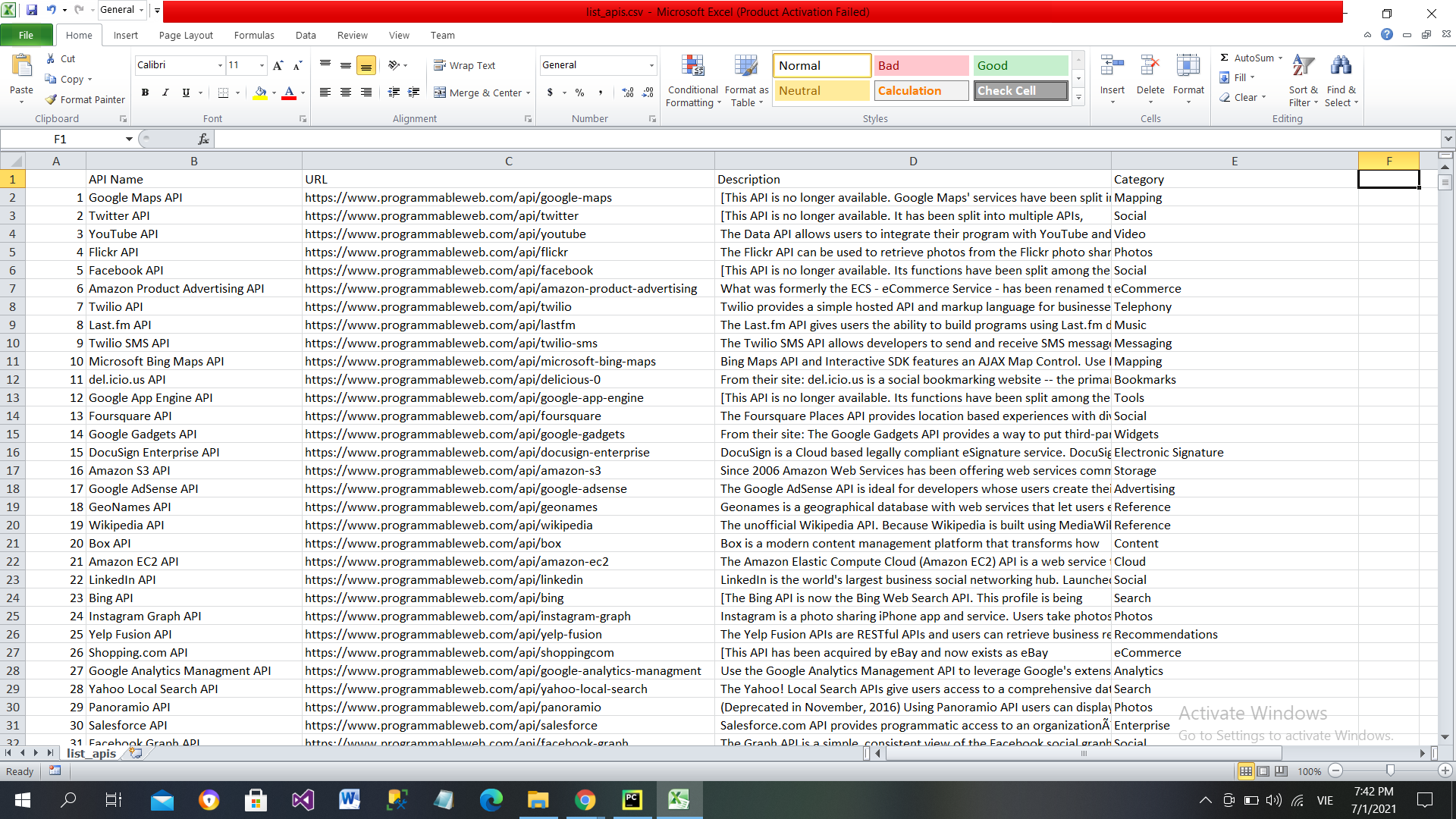
np\_apis\_df = pd.DataFrame.from\_dict(api\_pb, orient = 'index', columns = ["Name","URL","Description","Category"])

np\_apis\_df.to\_csv('np\_apis.csv')

Hình ảnh chứng minh:



Hình 3. Kết quả chạy code



Hình 3. Kết quả lưu về file csv

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Qua quá trình nghiên cứu, học tập cùng với sự hướng dẫn tận tình của GVHD: TS.Nguyễn Thành Thủy, đã giúp nhóm hoàn thành đề tài “***Crawl dữ liệu tự động trên web***”. Cùng với việc hoàn thành khóa học *Learn Web Scraping with Python from Scratch* đã giúp nhóm có được cái nhìn tổng quan nhất về việc trích xuất dữ liệu từ một trang web cũng như nắm được cách crawl dữ liệu bằng công cụ Beautifulsoup & Request. Nhóm cũng đã thực hành crawl thành công, lấy dữ liệu từ các trang web đơn giản. Tuy nhiên, hiện nay các trang web được thiết kế một cách phức tạp và bảo mật nên khó có thể lấy dữ liệu bằng các cách thông thường. Ví dụ như các trang web thương mại điện tử,…Để có thể trích xuất được dữ liệu từ các trang web này, thời gian tới nhóm sẽ vẫn tiếp tục tìm tòi, học thêm các khóa học và các công cụ mới, mạnh hơn, nhanh hơn.

Sau đề tài này, nhóm sẽ vẫn tiếp tục định hướng phát triển đề tài từ việc crawl dữ liệu trên một trang web sang việc crawl dữ liệu ở trên nhiều trang web, hoặc có thể xây dựng được một trang web thu thập dữ liệu từ nhiều trang web nhằm so sánh mức giá các sản phẩm,…Nhóm chúng em rất mong nhận được ý kiến đóng góp từ quý thầy, cô trong khoa để có thể hoàn thiện hơn đề tài của mình.

Nhóm xin chân thành cảm ơn!

# TÀI LIỆU THAM KHẢO

1. Udemy, *Learn Web Scraping with Python from Scratch:*

*link* *https://www.udemy.com/course/web-scraping-python-tutorial/learn/lecture/12925262#overview*

2. Spiderum, *Web Crawling và Web Scraping:*

*link https://spiderum.com/bai-dang/Web-Crawling-va-Web-Scraping-Su-tranh-cai-giua-an-cap-tai-lieu-hay-la-mot-cong-nghe-thu-thap-du-lieu-ruk*

3. *Một số khái niệm:*

*link https://vi.wikipedia.org*

*link https://quantrimang.com*

*4.* NiitHanoi*, Tại sao phải sử dụng Python cho AI và Machine learning:*

*link https://niithanoi.edu.vn/tai-sao-lai-su-dung-python-cho-ai-va-machine-learning.html*